Using "Universal Data Models" to Jump-Start Your Data Modeling Effort

Len Silverston
Kent Graziano
Quest Database Consulting, Inc.

Abstract

On many of our data modeling consulting engagements, clients often ask us the same question: "Where can we find a book or paper showing a standard way to model this structure? Surely we are not the first company to model company and address information!"

In general, one third of a data model (corporate or logical) consists of common constructs that are applicable to most organizations and the other two thirds of the model are either industry or enterprise specific. This means that most data modeling efforts are at some point re-creating data modeling constructs that have already been built many times before in other organizations. Doesn't it make sense then to have a source from which you can get a head start on your data model so you are not "re-inventing the wheel" each time you are asked to develop a new system?

This paper will illustrate some examples of common or "Universal Data Models" related to one subject area and explain how they can be used as a starting point for most data modeling efforts. By using these constructs, both time and money can be saved in systems development efforts. Note that there are many other "Universal Data Models" currently available for other subject data areas and applications in <u>The Data Model Resource Book</u> (see reference at the end of this paper).

The need for "Universal Data Models"

Data modeling has been an art that first gained recognition since Dr. Peter Chen's 1976 article which illustrated his new-found approach called "Entity-Relationship Modeling". Since then it has become the standard approach used towards designing databases. By properly modeling an organization's data, the database designer can eliminate data redundancies which are a key source for inaccurate information and ineffective systems.

Currently, data modeling is a well known and accepted method for designing effective databases. Therefore, there is a great need to provide standard templates to enterprises (the term enterprise is used to describe the organizations for whom the models and systems are being developed) so they can refine and customize their data models instead of starting from scratch.

Although many standards exist for data modeling, there is an great need to take data modeling to the next step: providing accessibility to libraries of common data model examples in a convenient format. These libraries of models should be able to be used across many different organizations and industries. These "universal data models" can help save tremendous amounts of time and money in the systems development process.

A Holistic Approach to Systems Development

One of the largest challenges to building effective systems is integration. Systems are often built separately since there are particular needs at different times within each enterprise. Enterprises have needs to build many systems: sales contact management systems, sales order systems, project management systems, accounting systems, budgeting systems, purchase order systems and human resources systems, to name a few.

When systems are built separately, there are separate pools of information for each system. Many of these systems will use common information such as information about organizations, people, geographic locations or products. This means that each separate effort will build and use their own source of information. A huge problem with this approach is that it is almost impossible to maintain accurate up-to-date information since the same type of information is stored redundantly across many systems. In large organizations, it is not uncommon to see information about customers, employees, organizations, products and locations stored in dozens of separate systems. How is it possible to know which source of information is most current or accurate?

Another way to approach systems development, is from a perspective that an enterprise's systems are connected and, in fact, may be viewed as one interconnected system. From this perspective, there are tremendous benefits to building an enterprise wide framework so that systems can work together more effectively. Part of this enterprise wide framework should include a corporate data model which can assist the enterprise in maintaining one of its most valued assets: information. Since each system or application may use similar information about people, organizations, products and geographic locations, a shared information architecture can be invaluable.

The IS (information systems) industry has recognized the need for integrated designs and this is why many corporate data modeling and corporate data warehouse modeling efforts have taken place. Unfortunately, the IS track record for building and implementing corporate data models has been very poor. Enterprises have realized that it takes a tremendous amount of time and resources to build these models.

Enter CASE (Computer Aided Systems Engineering) tools. These tools claimed tremendous productivity and time savings when used for corporate wide modeling efforts. While these tools help document the models, unfortunately they do not reduce the time to develop good corporate models. Many enterprises have stopped building corporate data models because of their time constraints. Enterprises are looking at the track record of corporate data modeling and CASE efforts and choosing other alternatives.

Enter "data warehousing". Finally, an approach to provide executives with the management information they need, without all the time and expense of corporate data modeling. Enterprises are now extracting the various pieces of information they need directly from their operational systems in order to build decision support systems.

The only problem with this approach is that *the same problem exists*! First of all, the information in the data warehouse may be extracted from several different inconsistent sources. If there are multiple places that customer information is being held, which system represents the most accurate source of information?

According to data warehousing principles, the transformation routines are responsible for consolidating and "cleansing" the data. However, if different departments have different needs for various pieces of data, then each department may build their own extracts from the operational systems. One department may transform the information using one algorithm while a different department may use another algorithm. For example, if two departments are extracting sales analysis information, one department may use the order entry system as its source and another department may use the invoicing system as its source. A high level manager may view information from both data warehouses and see inconsistent results, thus questioning the credibility of *any* of the information. This type of scenario actually compounds the initial problem of many data sources by creating even more "slices of data".

Not to say that data warehousing is the wrong approach. It is an ingenious approach which can be used extremely effectively not only to create decision support systems but to build a migration path to an integrated environment. The data warehouse transformation process helps to identify where there are data inconsistencies and data redundancies in the operational environment. However, it is imperative to use this information to migrate to new integrated data structures.

The answer is still to build integrated data structures in order to provide good, accurate information. It is also necessary to understand the nature of the data in order to build effective systems. Instead of saying that corporate data modeling or CASE is the wrong approach because it just takes too long, the IS community needs to find a way to make it work effectively.

By building common re-usable data structures, the IS community can produce quicker results and move towards integrated structures in both the transaction processing and data warehouse environments.

What is the intent of this paper and these models?

The approach behind this paper is dramatically different. Most data modeling books and papers focus on the techniques and methodologies behind data modeling. This paper assumes that the reader knows how to model data. Data modeling has been around long enough that most information systems professionals are familiar with this concept and will be able to understand this paper. Therefore, this paper makes no efforts to teach data modeling principles, except by example. Data modelers can use this paper (and/or the related book), and their previous experience, to build upon and refine the following data model examples in order to develop more customized data models. Essentially, it is providing the modeler with fundamental tools and building blocks which can be re-used. The modeler can thereby be more productive by starting with standard data models instead of building data models from scratch.

These models are intended to be a *starting point* for developing logical data models for an enterprise. Each enterprise will have their own detailed requirements and the models will need to be modified and customized in order to be implemented for a specific enterprise. In addition, the models in this paper can be used to validate an enterprise's existing data models.

Note that the models presented in this paper are logical data models and not physical database designs. Therefore these model are normalized and may require some denormalization when designing the physical database. This paper does not discuss methodologies for physical database design. Consistent with this point, the logical data models do not include any derived attributes since derived attributes do not add anything to the information requirements of a business. They merely serve to enhance performance of the physical database.

The logical data models in this paper represent possible data requirements for enterprises. The models do not include business processing rules that may accompany data models. The data models generally provide all the information that business rules would need, however the reader is advised in many cases that additional business rules need to be developed to supplement their data models. Examples of the need for business rules are provided throughout this paper.

The following data models were designed to be of benefit to many different industries and enterprises. These models were picked specifically because *they represent very common data constructs that appear (or should appear) in most organizations*. Within these models, whenever there was a data modeling decision which may have been dependent on a specific enterprise, the most flexible data modeling option was chosen in order to accommodate many different enterprises.

Samples of Common Models for People and Organizations

The most frequent business information need is to be able to ask questions about people and organizations and to be able to rely on accurate information. For instance, what are the attributes or characteristics of the people and organizations that are involved in the course of conducting business? What relationships exist between various people, between various organizations and between people and organizations? What are the addresses of people and organizations and how can they be contacted?

Almost all business applications track information about people and organizations, recording information about customers, suppliers, subsidiaries, departments, employees and contractors redundantly in many different systems. For this reason, it is very difficult to keep key information such as client contact data consistent and accurate. Examples of applications which store information about people and organization include sales, marketing, purchasing, order entry, invoicing, project management and accounting. The following sections discuss some standard data constructs for both organizations and people as well as their related data.

Organizations

Most data models maintain organizational information in various entities. For instance, there may be a customer entity, a vendor entity, and a department entity. Each application within an enterprise has its own needs, and therefore the data modeler will often base the model upon the needs of a particular application. When building an order entry application, the customer information is crucial; therefore the data modeler will show a separate entity for customer. When building a purchasing application, the supplier information is critical and hence there will normally be a supplier entity. For a human resources system, the data modeler might show an entity called a department within which the employees work.

The problem is that an organization may play many roles, depending on the particular circumstance. For instance, in larger companies, internal organizations sell to each other. The property management division may be a supplier to the product sales division. The property management division may also be a customer of the product sales division. In this case, there would normally be both a customer and supplier record, with redundant data, for each of these divisions. Not only could there be a customer and supplier record, but there could be many additional records for the organization depending on how many roles the organization plays within the enterprise.

When an organization's information changes, such as a change in address, the information might be updated in only one of the many systems where organization information is stored. This, of course, results in inconsistent information within the enterprise. It may also result in major frustration on the part of managers, customers, suppliers and anyone who might want to get out a correct mailing list!

The solution to this redundancy problem is to model an entity called 'ORGANIZATION' which stores information about a group of people with a common purpose such as a corporation, department, division, government agency, or non-profit organization. Basic organizational information such as its name and federal tax ID is stored once within this entity, reducing redundancy of information and eliminating possible update discrepancies.

Figure 1 shows the data model for organization information. The 'ORGANIZATION' entity is sub-typed into 'INTERNAL ORGANIZATION' and 'EXTERNAL ORGANIZATION'. An 'INTERNAL ORGANIZATION' is one that is part of the enterprise for whom the data model is being developed and an 'EXTERNAL ORGANIZATION' is not part of that enterprise.

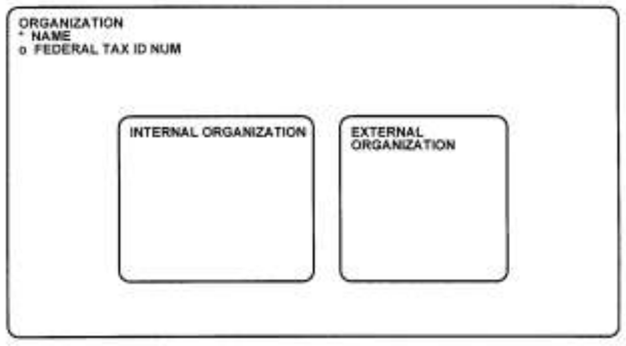

Figure 1 Organization.

This model reduces redundancy since the organization name is stored only once, as opposed to storing this information redundantly in a customer entity, supplier entity, department entity or any other entity storing organization information. Note that organizations include not only businesses but other groups of individuals such as departments. For example the accounting and information systems departments would be included as organizations.

People

Just as most data models show separate entities for various types of organizations, most data models also show separate entities for various types of people such as employees, contractors, supplier contacts and customer contacts. The problem with keeping this information in separate entities is that people may also have different jobs and roles which change over time. Most systems will record redundant information about a person since they store a record each time the person's role changes.

For example, John Smith was a good customer of ABC Corporation. John then decided to perform contract labor for ABC Corporation. ABC Corporation liked his work so much that they then hired him as an employee. For most systems, there would be a separate record for John Smith as a customer contact, then as a contractor and then as an employee. However much of John Smith's information has remained the same such as his name, sex, birth date, other demographics and skills. Because John Smith's information is stored in several locations, many systems would have trouble keeping his information accurate and consistent.

Another problem is that the same person may have many different roles ***at the same time***. For instance, ABC Corporation is a large company with many divisions. Shirley Jones is an employee and manager of the transportation division. She is also considered a customer for the supplies division. At the same time, she is the supplier for the publishing division who needs her services to transport catalogues. Shirley is therefore an *employee* of one division, a *customer contact* of another division, and a *supplier contact* of another division. Rather than have three separate records for Shirley with redundant information, there should only be one record for Shirley.

To address this issue, Figure 2 shows a 'PERSON' entity which stores a particular person's information independent of their jobs or roles. Attributes of the 'PERSON' entity could include sex, birth date, height, weight, and any characteristics which describe the person.

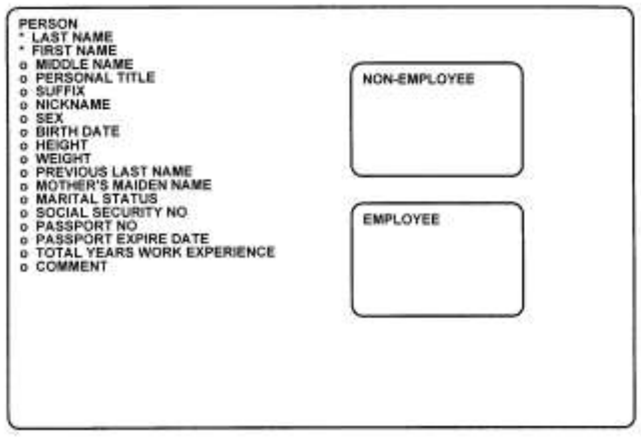

Figure 2 Person.

Just as the 'ORGANIZATION' entity is sub-typed, the 'PERSON' entity is sub-typed into 'EMPLOYEE' and 'NON-EMPLOYEE'. An 'EMPLOYEE' is a person that is employed by the enterprise for whom the data model is being developed and a 'NON-EMPLOYEE' is not employed by that enterprise (employed means that the person is an employee whose federal tax is withheld from their pay check by the enterprise).

This model has again helped reduce redundancy since the person's base information is only maintained once, even though the person may play many different roles. The "Party Relationships" section later in this paper will describe how to model the various roles each person and organization can play.

Party Definition

Organizations and people are similar in many respects. Organizations and people both have common characteristics which describe them such as their credit rating, address, phone number, fax number, or E-mail address. Organizations and people can also serve in similar roles as parties to contracts, as buyers, as sellers, as responsible parties or as members of other organizations. For example, membership organizations (like a database users group) keep similar information on their corporate members and their individual members. Contracts can usually specify an organization or a person as a contracted party. The customer for a sales order may be either an organization or a person.

If person and organization were modeled as separate entities, the data model would be more complex. Each contract, sales order, membership, or transaction that involved either a person or an organization would need *two* relationships: one to the person and one to the company. Furthermore, these relationships are mutually exclusive and thus form an *exclusive arc*. For instance, a sales order could *either* be placed by a person *or* an organization but a single sales order *can not be placed by both* a person and an organization at the same time.

Therefore, Figure 3 shows a super-entity named 'PARTY' which has as it's two sub-types, 'PERSON' and 'ORGANIZATION'. This 'PARTY' entity will enable storage of some of the common characteristics and relationships which people and organizations share.

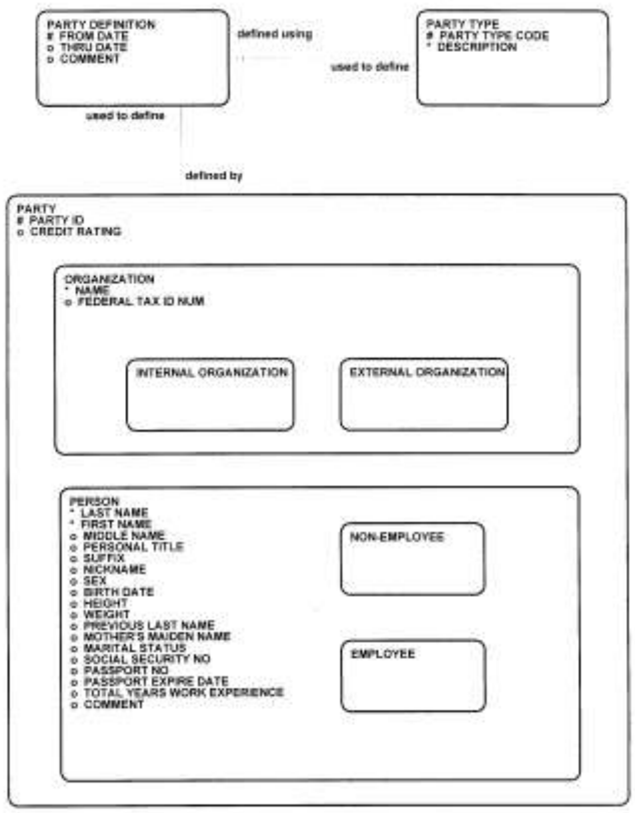

Figure 3 Party definition.

Parties are classified into various categories using the entity 'PARTY DEFINITION' which stores each category into which parties may belong. The possible values for categories are maintained in the 'PARTY TYPE' entity. For example, a type of party may be "minority owned business", "8A business", "woman owned business" "government institute" or "manufacturer". The categorizations of parties can be used to determine if there are any special business considerations for parties, special pricing arrangements, or special terms based upon the type of party. It is also a mechanism for classifying businesses into types of industries for market segmentation. A ‘from date’ and ‘thru date’ are included so history can also be tracked since it is possible for the definition to change over time (e.g., businesses may “graduate” from the 8A program).

Table 1 shows several party occurrences. This single entity allows other data models to refer to either a person or organization as a *party* to a transaction.

Table 1 Party Definition Data

Party ID	Party Type*	Name	Last Name	First Name
100	Minority owned business	ABC Corporation		
200	Subsidiary	ABC Subsidiary		
300	Department	Accounting		
400	Department	Information Systems		
5000	Shareholder		Smith	John
6000			Jones	Shirley
7000	Minority		Cunningham	Barry
8000			Johnson	Harry

* ***The value of this attribute in the entity described is actually a numeric ID. Instead, a description is provided for ease of understanding***.

Party Relationship

As noted previously, a person or organization may play any number of roles such as a customer, supplier, employer or subsidiary. Each role that a party plays only makes sense in relation to another party. If the ACME Company is a customer, are they a customer of ABC subsidiary? Or are they a customer of the parent company, ABC Corporation? Maybe they are a customer of the widgets division or maybe they are a customer of the gadgets division.

Instead of modeling just the roles of the party, there is a need to model the relationship between parties. For example, there is a need to know not only that ACME Company is a customer but that ACME Company is a customer of the ABC subsidiary. By default, this fact also implies that the ABC subsidiary is a supplier of the ACME Company.

A relationship is comprised of two parties and their respective roles. For example, customer/supplier, parent/subsidiary and division/department are possible organization relationships. The ‘PARTY RELATIONSHIP’ entity shown in Figure 4 allows parties to be related to other parties and maintains the respective roles in the relationship. The ‘PARTY RELATIONSHIP’ entity has attributes of ‘from date’ and ‘thru date’ in order to show the valid time frames of the relationship.

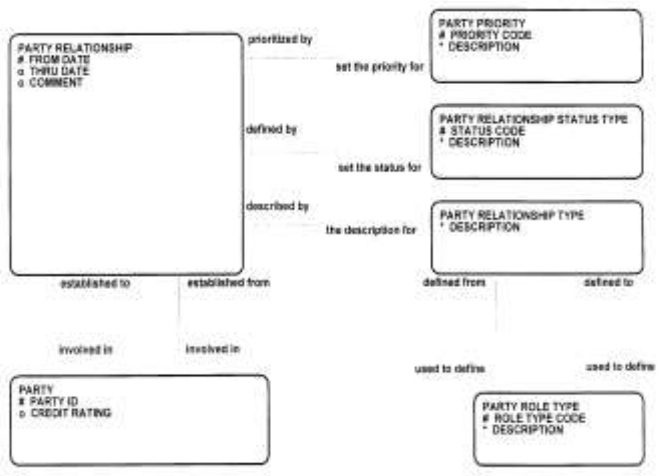

Figure 4 Party relationship.

The ‘PARTY RELATIONSHIP TYPE’ entity in Figure 4 consists of a pair of roles which are used to define the nature of a ‘PARTY RELATIONSHIP’. Customer/Supplier is a valid pair of roles, while the combination of Customer/Sales Agent roles would not be valid because these roles do not compliment each other (Authorizor/Sales Agent would make more sense). The ‘description’ attribute describes the nature of a specific relationship. For example, a customer/supplier relationship description may be “where the customer has purchased or is planning on purchasing items or services from the supplier”.

The ‘PARTY ROLE TYPE’ entity is a list of possible roles that can be played by the parties within a ‘PARTY RELATIONSHIP TYPE’. The two relationships from ‘PARTY ROLE TYPE’ to ‘PARTY RELATIONSHIP TYPE’ define the nature of the relationship. To form a customer/supplier relationship there would be two relationships to “customer” and “supplier” instances in the ‘PARTY ROLE TYPE’ entity.

The ‘PARTY RELATIONSHIP STATUS’ entity defines the current state of the relationship. Examples include “active”, “inactive” or “pursuing more involvement”. The ‘PARTY PRIORITY’ entity establishes the relative importance of the relationship to the enterprise. Examples may include “very high”, “high”, “medium” and “low”. Alternatively, an enterprise may choose to use “first”, second”, “third”, and so forth, to prioritize the importance of various relationships.

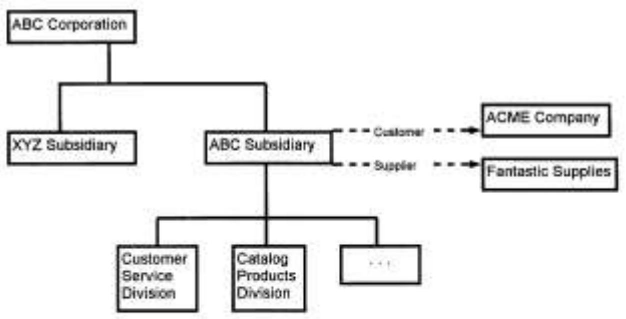

Figure 5 Party relationship hierarchy example.

Figure 5 illustrates the relationships for the organization, ABC Subsidiary. Table 2 shows the data which is stored in the ‘PARTY RELATIONSHIP’ entity to represent these relationships. The internal organizations are the ABC Corporation, ABC Subsidiary, and ABC’s Customer Service Division. The first row shows that ABC subsidiary is a subsidiary of the parent corporation, ABC Corporation. The second row shows that the Customer Service Division is a division of the ABC Subsidiary. The third row shows that ACME Company is a customer of ABC Subsidiary. Notice that the fifth row shows that ABC Subsidiary is a customer of Fantastic Supplies, or in other words, Fantastic Supplies is a supplier for ABC Subsidiary. If Fantastic Supplies was a supplier for all of ABC Corporation, there would be a relationship to the parent company, ABC Corporation instead of to the subsidiary.

Table 2 Organization to Organization Party Relationships

From Party*	To Party*	From Relationship*	To Relationship*	From Date	Through Date
ABC Subsidiary	ABC Corporation	Subsidiary	Parent Corporation	3/4/88	
Customer Service Division	ABC Subsidiary	Division	Corporation	1/2/92	
ACME Company	ABC Subsidiary	Customer	Supplier	1/1/94	
Sellers Assistance Corporation	ABC Subsidiary	Sales Agent	Authorizing Corporation	6/1/95	12/31/95
ABC Subsidiary	Fantastic Supplies	Customer	Supplier	4/5/93	

Just as organizations have relationships with other organizations, people have relationships with other people. Examples of person to person relationships include people’s reporting structures, people’s mentors, people’s family structures and people’s previous managers. Table 3 shows person to person relationship examples. These relationships are stored in the same entity (‘PARTY RELATIONSHIP’) as organization to organization relationships, however Table 3 breaks out the person to person relationships for ease of understanding.

In Table 3, John Smith reported to Harry Johnson in 1995 and currently reports to Jim Biggs. John Smith has as a mentor, Barry Goldstein. Judy Smith is John Smith's daughter. Joe Schmidt is the customer representative who Nancy Barry calls upon to sell her company's products. John Smith is also Barry Cunningham's customer (contact).

Table 3 Person to Person Party Relationships

From Party*	To Party*	From Relationship*	To Relationship*	From Date	Through Date
John Smith	Harry Johnson	Reports to	Manager	1/1/95	12/31/95
John Smith	Jim Biggs	Reports to	Manager	1/1/96	
John Smith	Barry Goldstein	Apprentice	Mentor	9/2/95	
Judy Smith	John Smith	Child	Parent	4/5/92	
Joe Schmidt	Nancy Barry	Customer Contact	Supplier Contact	3/15/93	
John Smith	Barry Cunningham	Customer Contact	Supplier Contact		

Finally, a person may play any number of roles within an organization. The person may be a employee of an organization, a supplier contact, a customer contact, and so on. Table 4 shows examples of people's roles within organizations. For Example Nancy Barry, John Smith and William Jones are all employees of ABC Subsidiary. William Jones is not only an employee of ABC Subsidiary but also contracts to Hughes Cargo. Barry Cunningham is a supplier representative for Fantastic Supplier and therefore people can contact him to purchase items from Fantastic Supplies. Joe Schmidt is the customer representative for ACME Company and therefore represents their interests as a customer.

Table 4 Person to Organization Party Relationships

From Party*	To Party*	From Relationship*	To Relationship*	From Date	Through Date
Nancy Barry	ABC Subsidiary	Employee	Employer	7/19/82	
John Smith	ABC Subsidiary	Employee	Employer	12/31/89	12/01/92
William Jones	ABC Subsidiary	Employee	Employer	5/07/90	
William Jones	Hughes Cargo	Contractor	Contracting Firm	1/31/95	12/31/95
Barry Cunningham	Fantastic Supplies	Supplier Representative	Supplier	2/31/95	
Joe Schmidt	ACME Company	Customer Representative	Customer	8/30/95	

Address Definition

Figure 6 shows the data model for address related information. The 'GEOGRAPHIC BOUNDARY' entity stores the 'COUNTY', 'CITY', 'STATE' and 'COUNTRY' of an 'ADDRESS'. The 'ADDRESS' entity maintains all addresses used in the enterprise in a central place. The 'PARTY ADDRESS' entity shows which 'ADDRESS' is related to which 'PARTY'. The 'PARTY ADDRESS ROLE' entity defines the roles that a 'PARTY ADDRESS' may have and is primarily used to validate that the address is used for its intended purpose. 'PARTY ADDRESS ROLE TYPE' maintains the possible values of the roles which addresses may play.

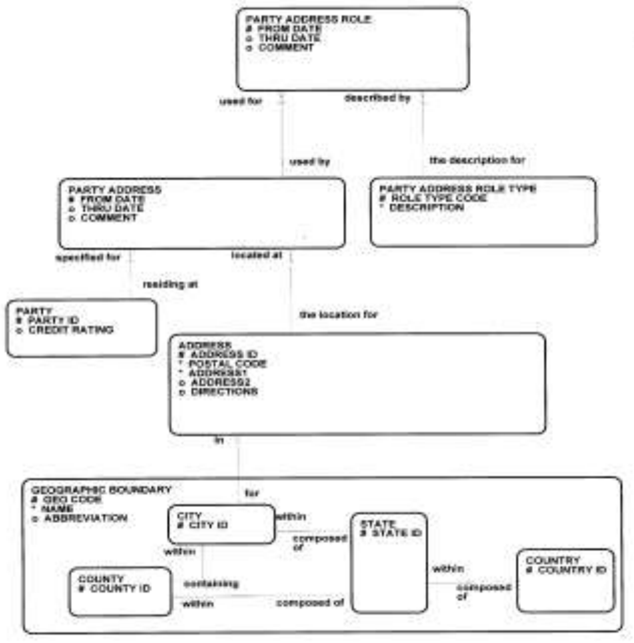

Figure 6 Address definition.

Address

Each 'ADDRESS' has geographic boundaries in which it resides. These could include counties, cities, states, territories, provinces (Canada), prefectures (Japan), regions, countries and they will vary by country. *As an example*, the model in Figure 6 includes the subtypes 'COUNTY', 'CITY', 'STATE', 'COUNTRY' and the super-type 'GEOGRAPHIC BOUNDARY' with appropriate relationships between them.

The 'ADDRESS' entity stores attributes to identify the specific location within the geographic boundary. The 'address1' and 'address2' attributes provide a mechanism for two text lines of an address. There may be a need for more address line attributes depending on the needs of the enterprise. The 'postal code' identifies the mailing code that is used for delivery. The 'directions attribute' provides instructions on what roads to travel on and what turns to take in order to arrive at that address.

Party Address

An organization may have many addresses or locations. For instance, a retailer might have several outlets at different addresses. In this instance, there is only one 'ORGANIZATION' or 'PARTY' but many locations or addresses. Additionally, the same address might be used by many organizations. For instance, many departments might share the same address. Another example is that the subsidiary and parent company might share the same address. Also, different organizations might share the same address if they are in a shared office facility. Therefore, there is a many to many relationship between 'ORGANIZATION' and 'ADDRESS'.

A particular address may have many people residing there such as when many employees work at the same facility. And, of course, people generally have many addresses; their home address, work address, vacation address, and so on. So there is also a many-to-many relationship between 'PERSON' and 'ADDRESS'. Instead of two separate relationships for people and organizations, the model shows a many-to-many relationship between 'PARTY' and 'ADDRESS'. The many-to-many relationship is resolved via an *intersection entity* (sometimes referred to as an *associative* or *cross reference* entity) named 'PARTY ADDRESS' in Figure 6. Notice that 'PARTY ADDRESS' has a from and thru date which allows the ability to track the address history of parties. With this model, addresses are only stored once, thus eliminating redundant data problems, and can be reused many times in relationship to many parties.

Address Role

For each party located at an address, or 'PARTY ADDRESS', there may be many purposes or roles for the address. The address might be a mailing address, a headquarters address, a service address, and so on. Most systems have a separate record for the mailing address, headquarters address and service address, even though the address information may be exactly the same. Therefore, the data model in Figure 6 shows that each 'PARTY ADDRESS' must have one or more 'PARTY ADDRESS ROLES'. The 'PARTY ADDRESS ROLE' stores the roles an address may play. A list of possible values is available in the 'PARTY ADDRESS ROLE TYPE' entity.

Another way this could be modeled is to include the role in the 'PARTY ADDRESS' entity and have additional cross reference records for each of the address' roles. For example, if the same party's address served as a mailing, headquarters and service address, it would be stored as three instances in the 'PARTY ADDRESS' entity. Each instance would have the same party and address ID but would have a different role. The disadvantage of this model is that the 'PARTY ADDRESS' entity has significance on its own. For instance, each 'PARTY ADDRESS' may have telephone and fax numbers associated with it. For this reason, our model shows separate ‘PARTY ADDRESS’ and ‘PARTY ADDRESS ROLE’ entities.

Table 5 illustrates that ABC Corporation’s address can be used as the corporate headquarters, central mailing address and legal office. The ‘PARTY ADDRESS ROLE’ entity provides for the storage of a party address only once with many roles for that party’s address.

Table 5 Party Address Role

Party*	Address ID	Address Role*
ABC Corporation	2300	Corporate Headquarters
ABC Corporation	2300	Central Mailing Address
ABC Corporation	2300	Legal Office
ABC Subsidiary	2400	Sales Office
ABC Subsidiary	2400	Warehouse

Contact Mechanism Definition

In many data models, phone numbers are shown as attributes of the organization or person. There are also usually fields for fax numbers, modem numbers, pager numbers, cellular numbers, and electronic mail addresses. This often leads to limitations in the systems built. For instance if someone has two business phone numbers and there is only one business phone number field for a person, where is the other business phone number entered? In this new world where there are many methods for contacting parties, more flexible data structures are needed.

The ‘CONTACT MECHANISM’ entity in Figure 7 stores access numbers for parties. Each ‘CONTACT MECHANISM’ may be the way to contact either a particular ‘PARTY’ or ‘PARTY ADDRESS’. The intersection entity ‘’PARTY CONTACT MECHANISM’ shows which contact mechanisms are related to which parties or addresses. The model also shows valid roles available via the relationship from ‘PARTY CONTACT MECHANISM’ to ‘PARTY CONTACT MECHANISM ROLE’. The ‘CONTACT MECHANISM TYPE’ and ‘CONTACT MECHANISM ROLE TYPE’ are entities which maintain allowable values.

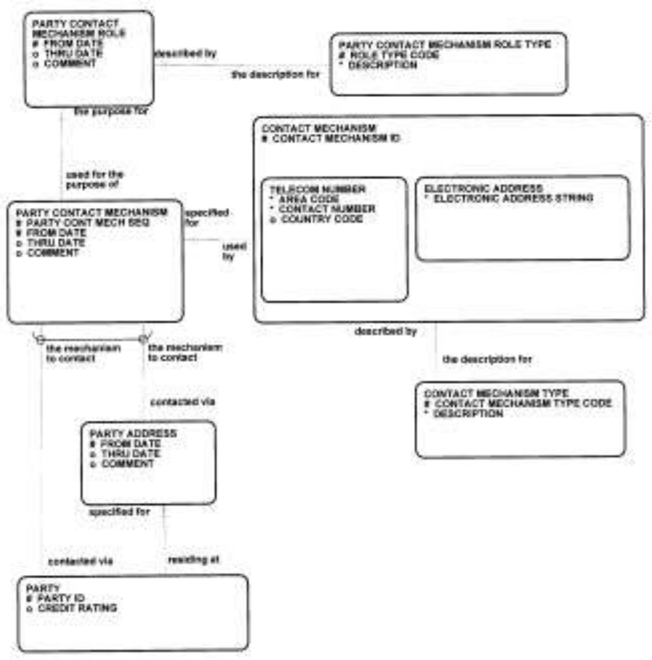

Figure 7 Contact mechanism definition.

Contact Mechanism

‘CONTACT MECHANISM’s are sub-typed to include ‘TELECOM NUMBER’ and ELECTRONIC ADDRESS’. ‘TELECOM NUMBER’ includes any access via telecommunications lines such as phones, faxes, modems, pagers, and cellular numbers. ‘ELECTRONIC ADDRESS’ includes any access via services like the Internet or other electronic mail services.

The ‘CONTACT MECHANISM TYPE’ entity shows all the possible values for types of contact mechanism. Examples include office phone, home phone, office fax, modem, cellular, Internet Address, and other electronic addresses. With technology growing so quickly, it is very likely that there will be many ways to get in touch with someone. The data structure in Figure 7 provides an easy method for adding any new contact mechanisms by simply inserting and using new ‘CONTACT MECHANISM TYPE’s.

Contact Mechanism Relationships to Party and Party Address

A contact mechanism could be tied to particular physical locations (namely ‘PARTY ADDRESS’) such as the telephone number for a retailer’s store location or it might be tied to a particular ‘PARTY’ such as a person’s cellular telephone number. There is a many-to-many relationship from ‘CONTACT MECHANISM’ to both ‘PARTY ADDRESS’ and ‘PARTY’. For example, a contact mechanism may be used to contact more than one party such as a joint telephone number for a family. The contact mechanism may also be for more than one party address such as a roaming telephone number that covers more than one location. Therefore, the ‘PARTY CONTACT MECHANISM’ is used as the intersection entity and represents the combination of a ‘CONTACT MECHANISM’ used by either a ‘PARTY’ or ‘PARTY ADDRESS’.

Table 6 gives examples of party contact mechanisms. The first two entries show the phone and fax numbers which are tied to the ‘PARTY ADDRESS’ for ABC Corporation at 100 Main Street. The third and fourth rows show that John Smith’s number at 100 Main Street is (212) 234-9856 but he has another office phone at 345 Hamlet Place. These are both tied to a ‘PARTY ADDRESS’. John Smith also has a cellular number which is tied directly to his ‘PARTY’ instance. Barry Goldstein has an office phone which is tied to one ‘PARTY ADDRESS’ (his work address) and a home phone which is tied to a different ‘PARTY ADDRESS’ (his home address). He also has an Internet address which is tied directly to his ‘PARTY’ instance.

Table 6 Party Contact Mechanism

Party*	Party Address*	Contact Mechanism *	Contact Mechanism Type*
ABC Corporation	100 Main Street	(212) 234 0958	Office Phone
ABC Corporation	100 Main Street	(212) 334 5896	Office Fax Number
John Smith	100 Main Street	(212) 234 9856	Office Phone
John Smith	345 Hamlet Place	(212) 748 5893	Office Phone
John Smith		(212) 384 4387	Cellular
Barry Goldstein	100 Main Street	(212) 234 0045	Office Phone
Barry Goldstein	2985 Cordova Road	(203) 356 3984	Home Phone
Barry Goldstein		bgoldstein@abc.com	Internet Address

Contact Mechanism Role

Furthermore, just as addresses are intended for specific purposes, so are party contact mechanisms. A single contact mechanism may have more than one purpose. For example, business people sometimes have a single number for both their phone and fax needs. Therefore, the ‘PARTY CONTACT MECHANISM ROLE’ defines the designated purposes for each ‘PARTY CONTACT MECHANISM’. The valid roles are described in ‘PARTY CONTACT MECHANISM ROLE TYPE’.

An example of a party contact mechanism role is that a telephone number may be playing roles as the “primary business contact number” and as the “general information number”. Other possible party contact mechanism roles include “customer service number” or “invoicing questions line”. In the complex world of today, since there are usually many contact mechanisms, it is very useful to identify the purposes of each contact mechanism. Since the purposes of various contact mechanisms change over time, the ‘from date’ and ‘thru date’ identify when the purposes are valid.

Summary

This paper provides some examples of “Universal Data Models” relating to people and organizations. These represent only a small sample of the many reusable data models which can save tremendous amounts of time and money when applied to systems development efforts. These models were designed to be a very practical ***resource*** to allow data modelers, database designers, and other systems professionals to be more productive by building upon the logical data models presented.

These “Universal Data Models” can be used to:

- provide a starting point in developing a logical data model
- add a new section of a data model to an enterprise’s existing data model

- validate an enterprise’s existing logical data models and provide ideas for additions or modifications
- help build a corporate data model that demonstrates the interrelationships between information in various applications
- help systems developers to understand the nature of various pieces of data and possible options and examples

While care should be taken when modifying these models, as stated before, one purpose of these models is to provide a starting point for data modelers to work from. If the models are used for this purpose, modifications to the models *should be expected* and encouraged in order to meet the information requirements of each enterprise.

We hope that this is just the beginning of efforts towards “Universal Data Models” and that the Information Systems (IS) industry will continue to develop re-usable models. If the IS industry can develop more common sharable models over time, systems development professionals will be able to shorten systems development cycles and produce higher quality information systems at a reduced cost for the user community at large.

References

Much of the material in this paper was excerpted by permission from the John Wiley and Sons publication The Data Model Resource Book: A Library of Logical Data Models and Data Warehouse Designs by Len Silverston, W.H. Inmon, and Kent Graziano. The book contains many other common data models (and data warehouse designs) regarding products, orders, shipping, professional services, invoicing, work order management, budgeting, accounting, sales analysis, and human resources, to name a few.

There is a CD-ROM which may be ordered to supplement the book which includes a physical database design of the book’s logical data models (including those in this paper) using a very straightforward conversion process. No denormalization has taken place and each super-types with its sub-types are generally implemented as a single table. This should provide a good starting point for the physical database design of these models. The CD-ROM contains SQL DDL generated from Designer/2000. In addition, Quest has all these models (and more) available in a Designer/2000 repository.

Contact Information

Len Silverston, President~~, internet: lsilvers@qdbc.com~~

Kent B. Graziano, Jr., Director of Data Warehouse Services,- ~~internet: kgrazian@qdbc.com~~

Quest Database Consulting, Inc.

5600 S. Quebec, Suite 310D

Greenwood Village, CO 80111

Phone: 303-771-2246 Fax: 303-771-1866

Web Site: www.qdbc.com