Normalised Message Exchange in Building Industry

B. de VRIES*

ABSTRACT

A formal description of the message exchange process and a normalised data model are presented in order to provide a framework for exchanging unstandardised messages in the building process.

Key Words

information technology; electronic data interchange; process model; data model; message model

INTRODUCTION

The requirements for data-exchange between organisations differ from those for data-exchange between applications. For the latter several basically technical solutions have come forward varying from standard file formats to dynamic binding of objects. Within an organisation, applications will fit into process models and data models which are specific for that organisation.

This article contributes to the theory for a formal description of data-exchange in the building process between organisations. Whereas exchange of textual and graphical information has taken place for ages between human beings, it seems that automation requires a new approach. Even many of the current standards, codes and classifications are not consistent enough at a formal level. The design process in itself is also inconsistent. On the other hand computer programs require consistency at a high level. So there is a large gap between the information structure of a human being and that of a computer. The formal approach described in this paper tries to cover this problem, specifically for the information to be captured in electronic messages.

The Data-Exchange Process

Looking at the messages that are exchanged between building participants, there are two categories :

(a) Messages that follow a certain transaction sequence (sequential)

For instance, a request for product information to a supplier will normally always be followed by an answer with the requested information.

* *Eindhoven University of Technology, Faculty of Architecture and Building, The Netherlands.*

(b) Single purpose message without transaction sequence (parallel)
For instance the principal decides to change his demands and lets the architect know the consequences for the design.

For a predefined message sequence much more consistency checking is possible. But when we look at the data-exchange processes of different building projects, only parts of them have the same sequence structure. This is due to the ever changing way in which building companies work together.

Systems Theory

The systems theory is very helpful in structuring the message exchange process in the building industry. A system has the following characteristics :

- a system transforms input to output
- the output from a system can be input for another system
- a system can contain other systems.

For our purpose a system has many participants and where several activities can take place. For each activity input and output are defined as a set of building objects with attributes, independent of a specific building project.

Before the building process starts input and output have to be defined for all participants and for each activity during the building process. An activity (e.g. "The making of a preliminary design") can be performed by one or more participants.
In fact a formal project contract is required to define :

- who are the building participants
- what activities do they perform
- what is the information needed to execute the activity
- what is the result of an activity.

The project contract is central in the Message exchange process model described below, because it ensures that the required information is produced at any time by any participant. Of course, it does not give any guarantee about the quality of the information.

Message Exchange Process Model

The process of exchanging messages can be described as asynchronous movement of output from one system to input for another system (Figure 1). In our view each one of the participants in a certain building project acts as a system, such as the architect, the constructor, the principal and the engineer. The messages we consider are for instance: the global description of requirements, request for product pricing and information, etc.

Within a system there are [redacted]s and [redacted]s (Figure 2).
A building participant's organisation can be decomposed into:

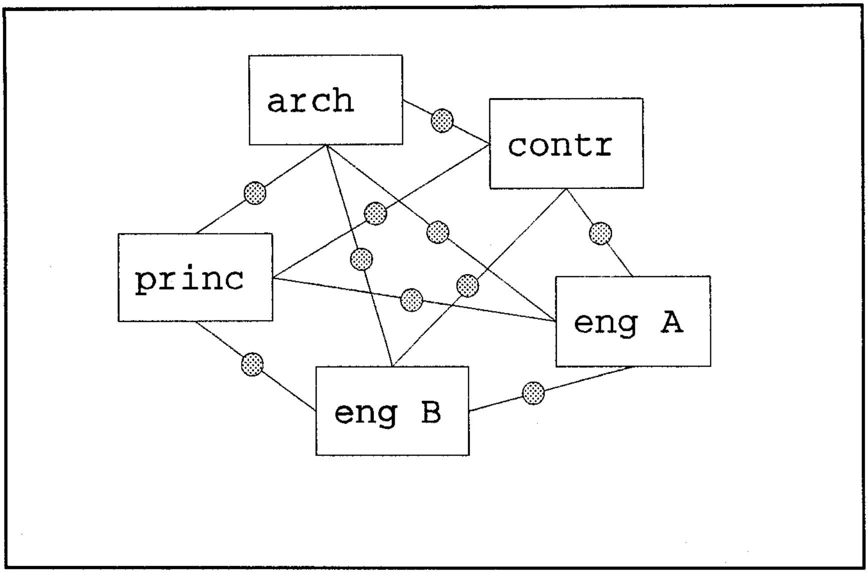

Figure 1. Building Participant Organisation

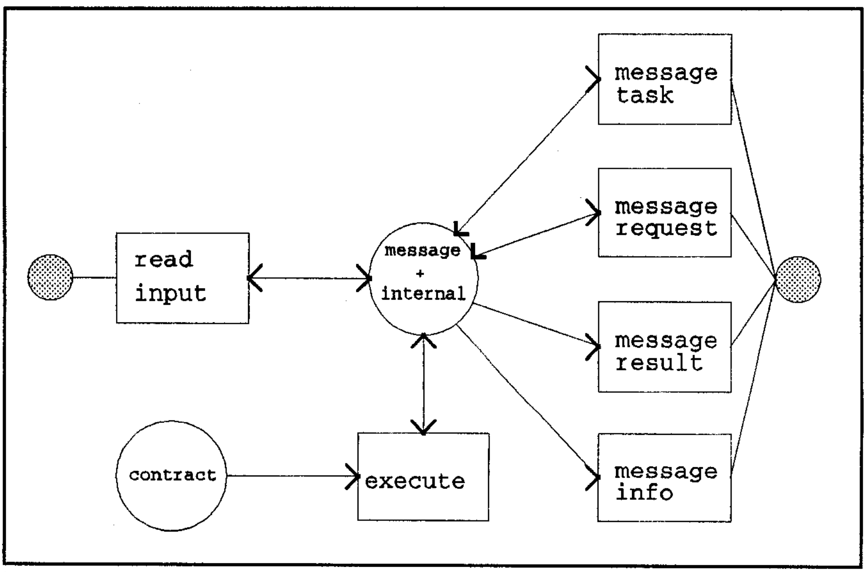

Figure 2. Message Exchange Process Model

1. Message-task system
To define a task to be fulfilled by another participant, objects are selected from the internal database to serve two purposes:
a) To submit the required input data
b) To describe the task to be done in terms of an Update, No_Update or None attribute for each attribute of a building object that belongs to the desired output. A ground floor (=object) with a Update attribute for the geometry (=aspect) means that a new design for the ground floor plan has to be made.
2. Message-request system
A request does not demand an action to be executed at another participant, but just a selection from his internal database. Similar to the message-task, input data are submitted and output data have the Update attribute.
3. Message-result system
A result is the output of an activity and a check is made whether the demands in the task that was fulfilled with the activity are met. A result can be a combined output of an activity that was executed by more than one participant. (E.g. the main contractor with its subcontractors)
4. Message-info system
An Info consists of a selection from the internal database. No activity is executed. A check is made whether the information is available or not.
5. Execute system
Before an activity is executed two controls are made :
a)The required input data must be available through either the received message or the available internal data
b)The activity must be capable of executing the required operations ([no]-update).
A participant has a definition of required input and produced output for every activity he can execute, independent of a certain building project. For the specific building project required input and produced output are defined in the contract store. So the controls must be made with both input/output specifications to ensure that all requirements are met and the building process can continue.
If an activity cannot be executed, the participant should
- collect more information from other participants
- produce the needed information with another activity in advance by himself or another participant
- let the complete requested output or part of it be produced by another participant.
6. Read-input system
Received messages are read and moved to the message store. If the newly received object data are more recent than the ones in the internal store, they

will be updated. Data not yet available will be appended.
A message-result is checked against the task that had to be fulfilled. The recipient can accept the result after which the task is completed, or reject the result.
A message-info is checked against the requested information. If the answer was not properly received then the needed information must be collected at another participant.

7. Message + Internal Store

This is the database where the following information is available:

- project dependent building information
- project independent building information
- project dependent information about tasks and requests to be fulfilled
- project independent input/output information for each activity.

All information is stored as a set of tupels of objects + aspects and if relevant operation + status.

8. Project Contract Store

For each project the contract store contains information about the activity a participant will take part in. For each activity input and output are stored as a set of tupels of object + aspects.

Message Types

When we look at the kind of messages that are exchanged nowadays they often get a more or less common name, but their contents vary a great deal. This is not so surprising because the messages often contain the output of a creative process. Standardisation of the process and data structure of unique designed products such as buildings will lead to a very rigged and complex data-exchange system.

Still a lot of messages are more or less independent of the specific building project, such as product and price information, payment of products, etc. This information is presently often filled in on proformas.

So a division can be made between:

- messages with a project dependent data structure
- standardised messages.

If a message is easy to standardise it is more profitable because more information can be put into the message. When data and semantics are exactly known at sender and recipient, then the data-exchange process can be fully automated.

If a message is not easy to standardise then the data structure itself is part of the semantics. The way the design is structured is specific for that design. Relations between the building objects are the expression of this structure.

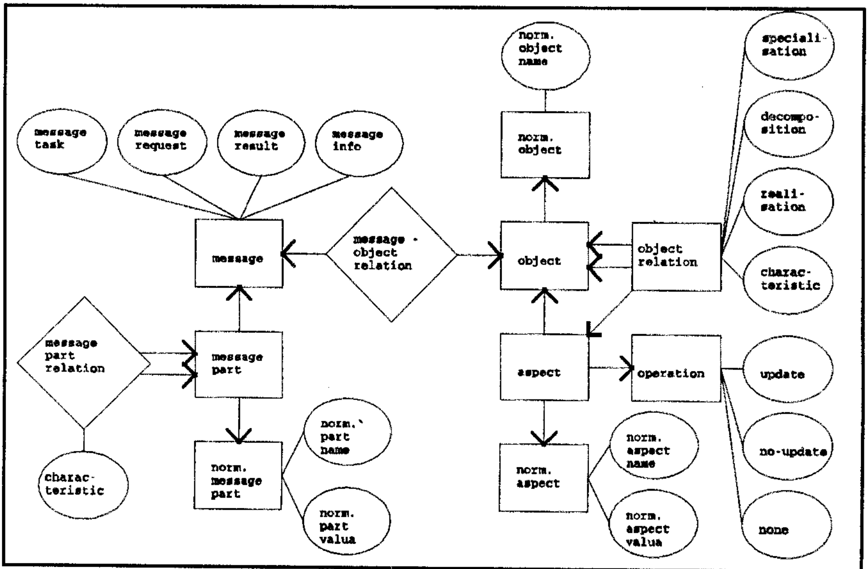

Figure 3. Normalised Message Model

Normalised Message Model

The normalised message model is the data model that is specifically designed to meet the demands of the electronic messages of the Message exchange process model and of the messages with a project dependent data structure (Figure 3). These assumptions lead to the following conclusions:

1. there should be a strong relationship with natural language to create the desired interpretation of the design;
2. redundancy of information is allowed;
3. objects, aspects and relations must have an unambiguous interpretation;
4. objects are atomic to make free ordering and selecting possible
5. description of the designed object "as-is until now"; no participant dependent knowledge is enclosed;
6. constraints between objects or aspects are not part of the message model because they are specific for each participant and may conflict with each other; and
7. integrity is not maintained by the message model for the same reason as mentioned in 6.

The main characteristics of the Normalised Message Model are:

a) Objects, aspects, messages, message-parts, operations and relations are normalised. This means that there is a common understanding about their meaning and that their number is limited. These lists are comparable with dictionaries with an explanation for each official known word in a certain language.

b) Objects are:

- physical elements (e.g. door)
- spaces (e.g. kitchen)
- work (e.g. painting)

Concatenated names reflecting to an object with a certain aspect such as "front door" are prohibited. In that case : object=door and aspect=location.

c) Aspects are:

- physical (e.g. geometry)
- physiological (e.g. noise level)
- psychological (e.g. style)
- economical (e.g. energy consumption)

An aspect cannot be replaced by an object. Nested object structures are constructed by relations.

d) There are only four types of messages: message-task, message-request, message-result and message-info. Their meaning has been explained before. Message parts consist of groups of attributes containing information about sender and recipient of the message.

e) Every aspect value can have an operation attribute. There are only two types of operations: Update and No-Update.

f) There are only four relation types: specialisation, decomposition, realisation and characterisation. Decomposition and realisation relate two objects according to a certain aspect. This means that they must have a common aspect. Specialisation and characterisation can relate according to any aspect. So though there is only a small variety of relation types it is possible to model a great variety of building data structures.

Because the data structure is not normalised no automated processing is possible. But what is more important is the ability to create a message containing the information the sender wants to. Moreover, the recipient must be able to retrieve the information in the way he can use it. Because the dictionary (object aspect) is well defined, only the construction (relation) needs interpretation. To do this formal (SQL) questions can be formulated in a rather simple and meaningful way. The output of the question will always be zero or have more aspect values.

To link the output to the desired input for an activity, there are sophisticated techniques coming forward such as Dynamic Data Exchange and Dynamic Object Linking. One should be aware that the link is only temporary, because it is unlikely that a new message will have the same data structure again. For equivalent messages with a low frequency there is the possibility of storing the linking; otherwise standardisation of the message should be considered.

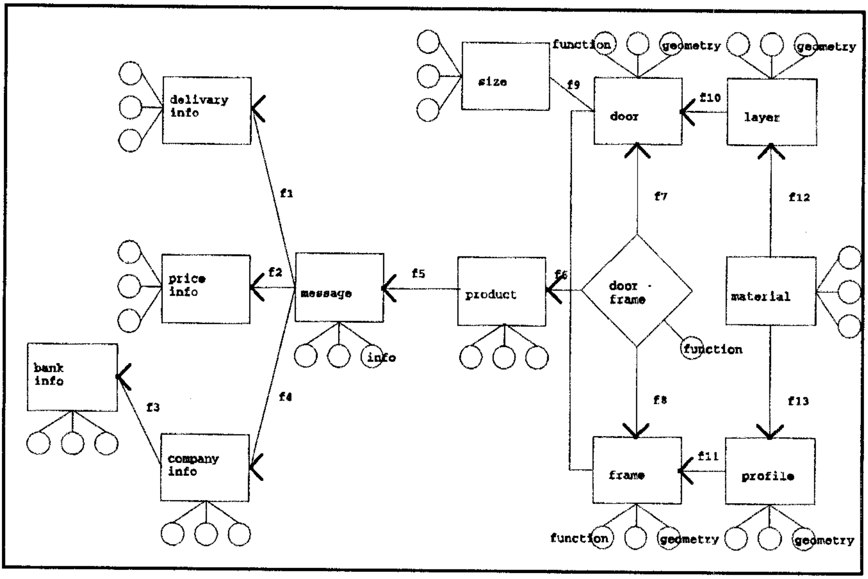

Figure 4. Example of a Normalised Message

Example of a Normalised Message

Figure 4 shows an example of a normalised message (not a standardised message). The message is a reply for a request for product and price information for a room door. The supplier returns information about doors and frames.

In this example the following relations are used:

f1,f2,f3,f4,f5 : characterisation
f6 : specialisation
f7,f8 : decomposition(function)
f9 : characterisation
f10,f11 : decomposition(geometry)
f12,f13 : characterisation

An aspect is written as a parameter of the relation type.

Knowing the normalised objects, aspect and relations that are used, the data model can be reconstructed from the message. At the same time, data can be stored in a database. Selections can be made from the database using textual and graphical representations. Many building objects have a graphical representation. To exchange shape information there must be a common understanding about the graphical format (Such as DXF,IGES,Step/Geometry). Only the shape of the object can be part of the data model (geometry), not derived values such as length and area. The recipient must be capable of generating these values from the geometry data. In fact, this is part of the data interpretation for specific usage that is done by a participant in the building process.

References

ACM Trans. on Database Systems.

Aerts, ATM (1991), Conceptueel Modelleren van Informatiesystemen. Academic Service.

Björk, B C (1993), A conceptual model of spaces, space boundaries and enclosing structures. Automation in construction, Elsevier, 193-214.

Chen, P (1976), The entity-relationship model: towards a unified view of data.

Eastman, C M and Bond, A H (1991), Product Information Models.

Eastman, C M (1992), A data model analysis of modularity and extensibility in building databases. Environment and Building.

Koetsveld, M J van (1992), Product Data Interchange. Kluwer.

Vlist, P van de (1992), EDI en de Industrie. Samson.

Waard, M (1992), Computer aided conformance checking. Doctoral dissertation, Delft University of Technology.

c. 1993, Management of Information Technology for Construction, K. Mathur et al (Eds), World Scientific Publishing Co., Singapore.