The use of data models for assessing standard logistics software

ELSEVIER

Computers in Industry 25 (1994) 211-216

COMPUTERS IN INDUSTRY

Short Note

The use of data models for assessing standard logistics software

J.C.J. de Heij [a,b]

[a] *Coopers & Lybrand Management Consultants, Churchilllaan 11, 3527 GV Utrecht, Netherlands*
[b] *Eindhoven University of Technology, Graduate School of Industrial Engineering and Management Science, P.O. Box 513, 5600 MB Eindhoven, Netherlands*

Abstract

The traditional method of selecting standard (logistics) software is very much based on the use of large functional checklists. This paper first describes the limitations of the traditional method and how these limitations are normally resolved. Then the advantages of the use of data models in the area of software selection are argued. Based on these advantages a method is described to compare the data models of different software packages by using a Reference Data Model (RDM). Apparently the results of assessing software systems very much depend on the quality of this RDM. The design of this RDM is therefore a major part of the research. Up to now the use of this data-oriented approach appears very promising, provided that it is seen as an addition to the current method of software selection.

Keywords: Data models; Software selection; Logistics software; Reference models; Standard software packages

1. Introduction

An increasing number of companies have changed their minds about inventing the 'software wheel' themselves, instead they shop around in the large market of standard software packages. The traditional method of selecting such packages is based on an assessment of all requirements. The approach uses a sizeable checklist, "the longer, the better". This checklist forms part of the request for proposal to various suppliers (the more, the better?!). Afterwards, selection is no more than weighing and adding the answers. The package with the highest score is the best choice. It sounds logical to say that package selection could be done more effectively. Nevertheless, the same pitfall is always imminent. One is creative enough to devise ever more requirements but at the same time, afraid to scrap so much as a single item in case it could be important.

Together with the Eindhoven University of Technology, Coopers and Lybrand Management Consultants is currently studying the role that data models can play in evaluating standard software. The study is focused on standard software for production control. The proposition being tested is whether the use of data models is a useful *addition* to the common (process-oriented) methods of selection. With the data-oriented approach a number of limitations of the existing methodologies can be overcome.

Of course, standard packages cannot be selected on the basis of a data model only. Software evaluation is only one aspect of the selection process. Obviously, other important aspects should be taken into account during the evaluation. There are functional requirements which cannot be traced in the data model and one single data element may hide a multitude of functionality. Not to forget that demands made upon the supplier, the contract, and the price are also important aspects of the selection process. This article focuses

SSDI 0166-3615(94)00046-8

on the use of data models to evaluate the functional features of software.

2. Common process-oriented approach

An understanding of the business processes and the way in which they are to be controlled should be the basis for change, including areas such as information technology and automation. Before taking standard packages into consideration, it should be clear what the required business situation will look like, e.g., in terms of financial and logistic control. Detailing this control concept will result in a description of the preferred way of working. The final result will be a specification of requirements which should be met by the improved business processes.

By describing the processes and activities in more detail, the final specification will be formulated in terms which are closely related to the user and the organisation. Examples of requirements in a production environment are "subcontracting of production operations should be supported" or "procurement contracts and delivery schedules should be supported". Where standard software is an option, various suppliers are approached with—if possible—an exhaustive list of these sorts of requirements.

In practice, this approach yields a limited result. The lists with requirements have a natural inclination of growing and growing. Checklists with 500 criteria are by no means an exception. Afraid to forget something one tries to add as many items as possible. This leads to too much detail and loss of the main issue. In the worst case the main issue—what is really important—has not even been considered.

A possibly even more tricky problem is the consistent formulation of questions to the suppliers. Often these questions can be interpreted in various ways, which of course makes the supplier choose the interpretation which suits him best.

To support the process-oriented approach, software comparisons are published which consist of a 'shopping list' of questions, including the supplier's answers (the yeas and nays). Consequently these comparisons are never exhaustive (far from it), they are unable to pronounce upon significance or insignificance (because they are independent of a specific business situation) and turn out to be subject to various explanations. This is confirmed by suppliers who admit to taking advantage of this 'opportunity'.

Apart from the advantages and disadvantages of a process-oriented approach one should not forget that a detailed inventory phase also has a second—often hidden—objective, namely to create awareness, within the user organisation, of the opportunities to improve. Hence, involving the user organisation and creating commitment are vital issues in the change process which take time.

3. Why is the common process approach satisfactory in practice?

In spite of the problems mentioned, packages are selected and implemented successfully. For one reason this success can be attributed to the increasing applicability of the packages. Since software packages offer more and more functions (thus also increasing in size) the chance of choosing the wrong package diminishes. Requirements which cannot yet be met are collected by the supplier and added to a future release. If one cannot get what one wants, creative tricks and 'work-arounds' are devised by improper use of the software (which does not have to be a bad thing). Unfortunately, the surplus of functionality which is bought is not visible but lurks behind the higher procurement and maintenance costs, higher demands on hardware, longer response times, lengthy implementation projects and awkward communication with the main users who get too much information and are faced with a system that is complex in its use.

Due to a lack of knowledge of standard software, companies often make use of external professional support in these complex processes of change. Experience with the business, the software package market, software selection and implementation processes are of great advantage. An experienced professional will be able to gain insight into the major criteria quickly, both regarding control of the business processes and the information systems required. Using experience as a filter the major requirements—in terms of control and information provision—become clear. Often the resulting list will not be more than two typed pages.

Based on only a few discriminating requirements, called knock-out criteria, a short list of three or four eligible packages can be defined. To do so, sufficient

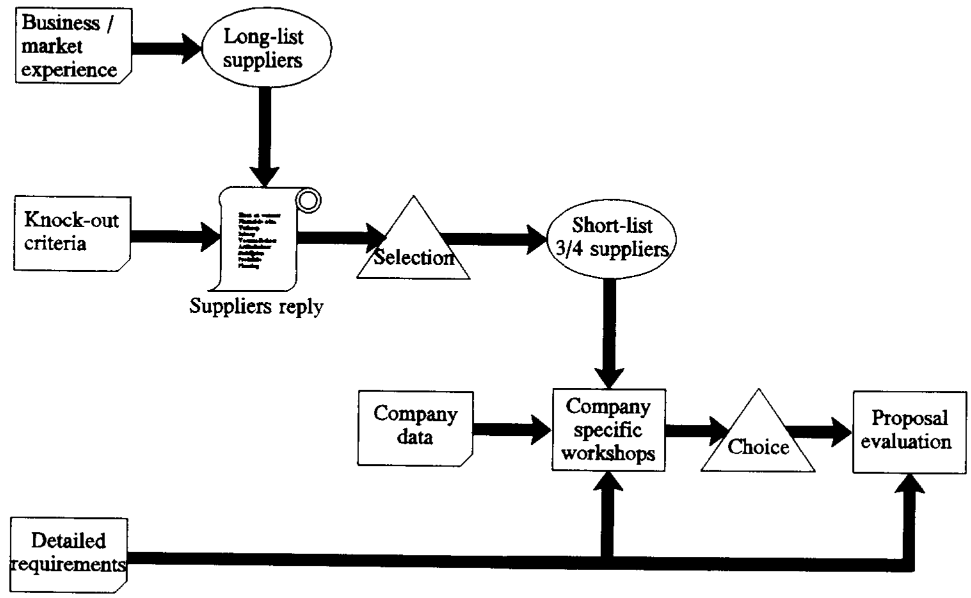


Fig. 1. Selection based on knock-out criteria.

and detailed insight into the available supply of packages is a must. Then the packages on the short list are extensively discussed in a company-specific workshop in which future (key) users and suppliers cooperate. Coopers and Lybrand has successfully applied this two-phase approach based on knock-out criteria many times.

Two critical elements remain: on the one hand the ability of the expert to distinguish between what is important and what not, and on the other hand his knowledge of the packages available on the marketplace to be evaluated on the basis of the knock-out criteria. Even an experienced consultant knows only a (rather) limited number of software packages in sufficient detail and suppliers are only too quick to claim that they can do anything. So it may be obvious that there is a clear need for a compact, transferable and unbiased recording of package knowledge.

4. Power of the data model

Much functionality of logistic packages—perhaps some 80 per cent—is pure registration. Obvious examples are the registration of sales, production and procurement orders and maintaining stocks. In the data base of these packages one also finds aspects that concern planning and control. Examples are the relationship between procurement and sales orders in the case of customer-specific purchasing (back to back deliveries). With regard to planning one could think of planning bills and production schedules at item and item group level.

Hence, the functional power of a package is also defined by its recording possibilities; the data model shows this. The data model is the core of the software, especially in those systems that support the operational business processes. The data model and the data dictionary are the fundament when developing information systems by using modern fourth-generation tools. This proposition is confirmed by suppliers who are very reluctant in releasing the data model of their packages. They fear that the competition will take advantage of this documentation in any further development.

Because the data model is regarded as the core of many information systems (this is true anyway for the present logistic packages), any structural change in the data model leads to an actual change in the surrounding functionality (software). Thus the data model helps to get to the bottom of what is important in terms of software packages.

The following may serve as an example. Many functional requirements such as monitoring suppliers performance are related to reporting functions. These requirements are not very relevant when using modern development tools. What is important, is whether the data required to make such reports is available. In this example there should be an accessible history of purchases and receipts, including data such as required, agreed and realised delivery time, price, quality and other procurement arrangements. Hence, this is in favour of assessing the data model rather than the reporting facilities already present in the package.

Other important advantages are the data model's density and objectiveness. Since the documentation of a package may sometimes cover several metres, it seems to be possible to describe, in sufficient detail, the logical data model of the same package in less than 25 pages.

The last aspect concerns the stability of the data model compared to the processes. Administrative processes are regularly subject to change, e.g., changes in the paper flow, alterations to the division of tasks and responsibilities, and the ever changing demands for management information. However, primary data are much less subject to change. For example, the introduction of lot tracking and tracing will comprise a radical change for a company, while changing from invoicing after goods receipt to invoicing in advance can be carried out relatively easy.

5. Availability of data models

Some practical problems need to be overcome before package data models of the required quality will be available. The quality of the technical specifications of many packages is mediocre. A diagram of the data model is often lacking. Most suppliers limit themselves to a record description without providing an understanding of the interrelationships, definitions and explanations. Still, also as a result of our study, a number of suppliers are now actively developing this sort of documentation with our support.

It turns out that in older packages the logical data model is a long way off the technical implementation especially in systems that have not been developed on the basis of a data model. A lot of tricks have been applied, such as files and data elements with more than one meaning and hidden relations. Unravelling and interpreting then takes place within the software. All in the name of response times and maximum functionality!

Even if it is available, the documentation is not easy to obtain. The confidential nature of the information is the reason for many suppliers not to release it, not even to presnt customers. To use the documentation one will have to sign a declaration of secrecy with the supplier.

6. The use of data models for package selection

The concepts mentioned above have prompted some companies to define a data model prior to selecting a package. Of course, these data models have been described in the corporate terminology. However the trouble starts when evaluating the packages. Apart from the fact that only few suppliers can or will provide the documentation concerned, it appears to be very difficult to compare one's own data model with that of a package. The reason for this is the difference in terminology and definitions. This problem is of the same magnitude as the comparison of data models of different packages, which also have their own terminology.

To deal with this problem we have developed a Reference Data Model (RDM). The RDM looks like a data model and it defines the terminology to be used, the level of detail and the scope. A RDM is a common framework in which both (logistic) business situations and standard software packages can be assessed.

In order to make a sensible comparison the (desired) business situation must be described in terms of the RDM. Instead of designing a data model from scratch, a characterisation is made directly in terms of the RDM. The result is a compact description of the many requirements to be met by the information system in order to support the business processes.

Next, data models are required of the packages to be evaluated. If the supplier gives this documentation, then their data model can also be projected onto the RDM. In the course of our study we have already carried out this activity for some much-sold standard packages. If the supplier cannot provide the data model, then he must carry out the projection directly himself. This request to the supplier will constitute an essential part of the assessment process. Although it is more difficult for a supplier to accomplish this as-

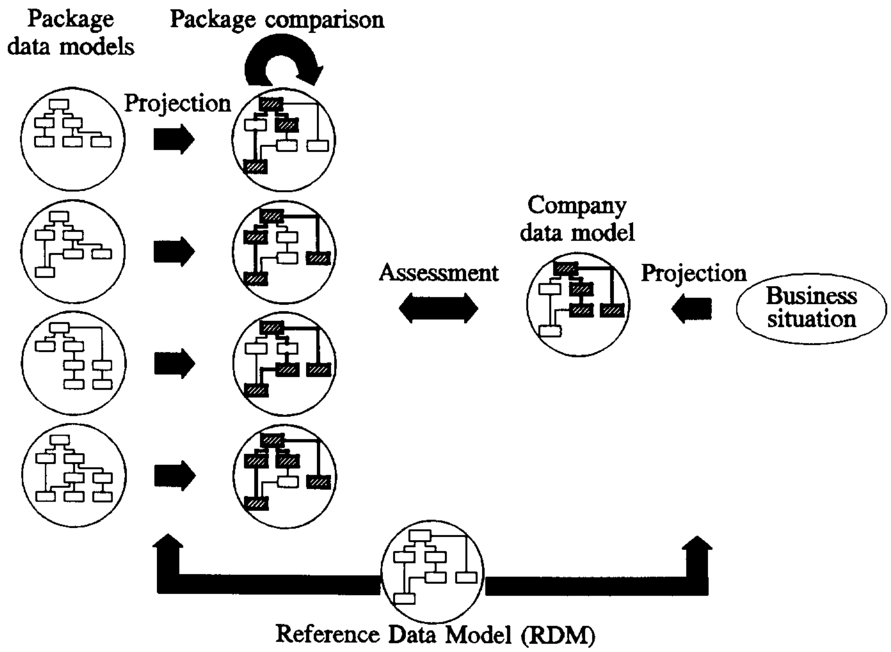


Fig. 2. Method for evaluating data models.

signment than to answer long lists with questions, it does lead to a sufficiently objective description of the package.

The projections always involve the usual discussions about the functionality of the package without it becoming apparent from the data model. Nevertheless, suppliers are remarkably enthusiastic, the reason being that this structured approach leads to the core of the chosen solutions and thus to what is functionally possible. It is this very discussion that provides insight into what the package is really worth. Hence the result is a projection of the package to be assessed based on the RDM.

Evaluation of the package can now be done 'simply' by comparing the projections. After all, the best choice will be the package projection which shows the most overlap with the required model. Naturally, by doing so one looks only at one (important) aspect of the package. The elements of the data model (entities, relations and data fields) which are not covered give insight into the necessary modifications and their implications. After all, changes in the data model affect the core of the system. Comparison also shows the surplus of one's purchase, and surplus functionality costs a lot of money (purchase, maintenance, bespoke tailoring, user instructions). Fig. 2 shows a diagram of the evaluation approach.

7. Features of the Reference Data Model

The core of the method described above is the Reference Data Model (RDM). Although basically any data model may serve as a reference, a good RDM has a number of special features. When developing a RDM the major objective is to gain insight into the differences between the data models to be projected. Decisions will have to be made as to the level of abstraction so that the differences between the software packages become visible. Many packages record various types of items in one file. For example, stock items, non-stock items, variant items, specials, services, drawings and even capacity units can all be recorded in one file. In order to assess what the packages can and cannot do, such a file must be specialised to a level of more than one entity.

In view of the breadth of functionality (purchasing, sales, stock, production and planning) most of the RDM consists of a balanced choice concerning abstraction, detail and scope. Without this valuation one will be swamped by the details; leading logistic packages may contain several thousands of entities. The objective remains the pursuit of insight into the applicability and not the detailed completeness of the package.

A RDM is also created by 'overlaying' the data models of various packages, resulting in a 'lowest common multiple'. Various control models, either capacity- or material-oriented, are thus combined into one model. Hence, much redundancy is created and fundamentally differing principles are combined into this single RDM. This is legitimate when projecting packages or production situations but the RDM is not a data model on which to base the development of a new software system. In fact, the proposition is that the RDM therefore is not a logical data model at all because of this combination of conflicting philosophies and concepts.

In the development of the RDM use is made of the experience of Coopers and Lybrand and the Eindhoven University of Technology together with the detailed knowledge embedded in the large number of available packages. Consequently, the RDM is also a point of view on subjects such as purchasing or bills of material, routings and recipes whose realisation has not yet been found in standard packages. A typical example is the exchange of a rolling forecast with the supplier. This concerns a purchasing forecast for the coming months in the course of which the uncertainty increases in quantity and specification with the horizon. This forecast only is considered precise and final for the immediate future as opposed to the old-fashioned one-off purchase order.

8. Study

This (postgraduate) research is a collaboration between experts from the Eindhoven University of Technology and Coopers and Lybrand Management Consultants. The study has three objectives:

(1) Further development and detailing of the Reference Data Model (RDM) with regular evaluation by comparing it with various packages and practical use;

(2) Increasing, recording and applying package knowledge in a structured way;

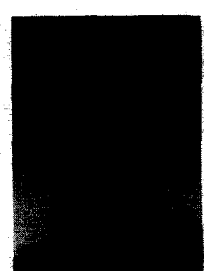
(3) Further development and testing of the proposed method for software assessment. Communicating this method and the results to users and non-IT people will be an important issue.

9. Conclusions

We have received many positive reactions from both suppliers and colleagues. "Everybody has been talking about the importance of data models for years" but structural research has never been done and reference models have never actually been applied. The main reason for this is the effort it takes to obtain the necessary information from the suppliers, as well as the mere idea and the effort required to develop an initial RDM which can serve as a growth model. Everyone agrees that the data model quickly provides insight into the possibilities and restrictions of a software package.

Some resistance to the approach has been encountered. Many people are used to looking at systems only from the outside and have neither the knowledge nor the experience to reason in terms of data models. As is the case with buying a car, most future owners look at the outward appearance. If an expert buys a car he usually opens the bonnet as well. By actually assessing the interior—the core of a software system—data models become a tool for experts and not for users. Nevertheless, how many users know the software market well enough to carry out a selection themselves?

The main condition remains a large involvement of the user organisation. Consequently, the business specification within the RDM should be built up step-by-step with the user organisation. In practice this will lead to the company's understanding of the activity and will form the basis for a meaningful and effective discussion concerning the required future situation.



Jos C.J. de Heij is a senior consultant in the Manufacturing and Logistics group of Coopers and Lybrand Management Consultants. He has been working there as a professional for more than five years and has extensive experience in the 'traditional' way of selecting and implementing standard logistics software in a wide variety of industries. Within the scope of his postgraduate research, which is the subject of this article, he is affiliated with the Eindhoven University of Technology.