Heft 92

A. Hars, R. Heib, Chr. Kruse, J. Michely
A.-W. Scheer

Approach to Classification for Information Engineering - Methodology and Tool Specification

August 1992

Table of contents

1. INTRODUCTION

This paper presents research findings that evolved from the ESPRIT project EP 5499 CODE (Computer Supported Enterprise-Wide Data Engineering). The main focus of the CODE project is the development of tool-based methodologies to support comprehensive enterprise-wide information modelling activities such as the tool-based utilization of data reference models, the design of re-engineering methodologies and the development of simulation-based decision support systems.

Within this information modelling context, classification is seen as an important vehicle to facilitate and support the management (i.e. retrieval, maintenance and consolidation) of the numerous information objects within a company. Classification in itself may be utilized for various purposes, depending on the classification object type it is applied for. Within the CODE framework classification schemes are chiefly applied for

- the selection of an appropriate data reference models and
- the search and retrieval of information objects

In the subsequent chapters the concept of a classification methodology is outlined and used to exemplify a possible implementation approach for a classification tool.

The paper is structured as follows. In the first chapter fundamentals and different approaches to classification are discussed. This theoretical introduction addresses general aspects of classification. The subsequent chapters form the core of the classification methodology. They are structured in a way to serve as a guideline for developing a classification tool. As a means to concisely describe the tool structure and its functionality an ERM-based meta-structure of the classification tool and a hierarchical process diagram is developed. Both diagrams are documented with the help of the tool managerVIEW which is part of the project background material.
Finally, a sample classification scheme is presented which can be used to classify the information objects of an enterprise.

2. FUNDAMENTALS OF CLASSIFICATION

2.1 Basic terms

The term classification is closely related to the concept of documentation. Documentation is defined by the International Federation of Information and Documentation as the collection and storage, classification and selection, dissemination and utilization of all types of information.[1] A system to store, retrieve and distribute information is called documentation system.[2] The core component of each documentation system is its indexing language (documentary language) - a well thought through system of concepts (i.e. abstract, cognitive entities) and terms (i.e. single- or multiword expressions) used for the representation and/or for the arrangement of objects and/or their substitutes with the objective of making the items retrievable.[3] In documentation systems resp. indexing languages often the problem of detecting synonyms and homonyms is encountered. In the case of synonyms, several terms denote the same concept, e.g. a customer may be represented by c-# in one system and by cust_id in another with both terms designating the identical concept customer. A homonym is a term that corresponds to several distinct, semantically different concepts. Both synomyms and homonyms should properly handled by indexing languages.

The term classification was defined in 1964 by Elismore as "any method creating relations, generic or other, between individual semantic units, regardless of the degree of hierarchy contained in the systems and of whether those systems would be applied in connection with traditional or more or less mechanized methods of document searching."[4] In most cases the term "classification" is used to designate a list of descriptors. Descriptors are terms used to designate concepts for indexing and retrieval purposes. Throughout this paper the term classification system comprises the following features:

- a list of descriptors,
- grouping of descriptors into descriptor classes[5] and
- the specification of relationships between descriptors and descriptor classes.

2.2 Indexing languages and classification systems

The objective of this chapter is to give a comprehensive overview of current indexing languages. As already mentioned the core component of each indexing language is its list of descriptors. Current indexing approaches are often classified according to the structure of the descriptor list into

- indexing languages without any classification of descriptors and
- indexing languages grouping descriptors into classes.

2.2.1 Catchword-, Keyword-, Uniterm approaches and Thesaurus

Within the group of indexing languages without any classification of descriptors, the catchword-, keyword-, and uniterm-approaches as well as the thesaurus are the best known representatives. Catchwords are words or phrases (groups of words) which are taken out of the text of the document (and its heading) to describe its content. Keywords are words or phrases assigned to a document in order to describe its content. Keywords may not be included in the document (text or heading). In the uniterm approach, that has been developed by Mortimer Taube in 1950, compound terms (keywords) are not allowed, therefore, they are decomposed into their linguistic components, i.e words of a natural language that cannot be split into smaller units without loosing their meaning. These components are listed in alphabetical order in a catalogue and are used as descriptors. All uniterms are coordinated, i.e., there is no hierarchical relationship between them. Only in the search process the descriptors are combined. This procedure is called coordinated indexing. The major weaknesses of the conventional catchword, keyword and uniterm indexing are that the number of descriptors is not restricted, and that - due to the equality of importance of the descriptors and the lack of mechanisms to control the list of descriptors - there is the problem of synonyms and homonyms.

A thesaurus is an indexing language that differs from the above approaches by providing means to cope with the problem of synonyms and homonyms. A thesaurus contains[6]

- a structured system of concepts with indication of the relationships (hierarchical or other) between the concepts,[7] and
- for each concept all terms that designate that concept (synonyms).

Among the synonymous terms of a concept one so-called preferred term is selected to unequivocally designate that concept. Those terms in a thesaurus that are allowed to be used to index and retrieve objects are called descriptors. Terms not allowed for description are called non-descriptors, they are referenced to the descriptor to be used by the relation "USE".[8] Thus, the descriptors are compulsorily defined. A thesaurus is subdivided into a core component and a register. In the core part all descriptors and non-descriptors as well as their relationships are described. Definitions and further commenting explanations (called scope notes) serve to differentiate the terms and consequently their underlying concepts. The thesaurus core may be sorted alphabetically, thematically or by a combination of both. In case of a thematic arrangement the descriptors and non-descriptors are ordered alphabetically in the register, and vice-versa.

2.2.2 Monohierarchical and Faceted Classification Systems

Monohierarchical and faceted classification systems represent major approaches of indexing languages with grouped descriptors, i. e. classes of descriptors have been introduced by grouping descriptors according to specific properties. With regard to the arrangement of the classes, monohierarchical and polyhierarchical structures are discerned. In monohierarchical systems a specific class must not have more than one upper class but may have several subordinated classes. Hence, the class structure of monohierarchical systems is representable as a tree. In the case of polyhierarchical systems a specific class may have more than one upper class, therefore the relationships among the classes is depicted as a network. Another structural property of classification systems refers to the number of aspects the classification object is looked at. This feature is called dimension. In monodimensional systems a specific class is subdivided according to exactly one criterion on each hierarchical level, in polydimensional systems, on the other hand, more than one subdivision criterion is allowed on the same level of the hierarchy.

One particularly well-known monohierarchical systems is Dewey's Decimal Classification (DDC) [9]. It was developed by the American librarian Melvil Dewey in 1876 with the purpose to achieve a unified systematical arrangement of books for public libraries. Dewey classified the whole domain of human (scientific) knowledge into ten main divisions numbered by the digits 0-9.[10] Each division is again divided into ten sub-classes and encoded by a digit which is affixed to the digit of the directly superior group. This procedure can be repeated until the desired level of detail of structuring knowledge is achieved. This fact is the reason for the wide-spread use of the decimal classification approach. The original ordering system of Dewey is restricted to three levels with about 1000 hierarchical structured descriptors.[11] Dewey's decimal classification system has been enlarged (especially by Paul Otlet and Henri Lafontaine) to the "Universal Decimal Classification" respectively "International Decimal Classification" with the objective to capture also the content of the documents.[12] Additional sub-classes and as a result more descriptors have been introduced. Major advantages of decimal classification systems are:

- by the continued subdivision the decimal classification system can be enlarged downwards boundlessly,
- the notation is clearly organized, practically self-explanatory, internationally intelligible and easy to apply.

The major disadvantage of the decimal classification is the rigid monohierarchical division into at most ten classes on each level. This restriction aggravates the unambiguous classification of elements. The constraint that descriptors

are arranged in a monohierarchy and cannot be combined freely leads to problems, as in reality complex concepts have not only one but several broader concepts.

The concept of the faceted classification[13] was introduced by G. Cordonnier and S. R. Ranganathan in the 1930s[14]. The faceted approach takes into account that the facts which shall be classified are very complex in meaning and therefore cannot be completely and definitely described by one single descriptor representing only one aspect as it is done in classical monohierarchical systems. The strength of the faceted classification is the description of the classification object with regard to different logical aspects called facets or categories. Categories may be considered as "upper" facets, facets as sub-categories, but in most cases both terms are used synonymously. Facets may consist of elementary descriptors (so-called isolates or foci) or other facets. Thereby, a hierarchical structure of facets is defined. The number of facets, their meaning and order is fixed. This implies that each classification object is described using the same pattern. In the classification process from each facet that term, that best characterizes the object, is selected. The classification is described by a combination of descriptors, each descriptor represents another dimension of the object. The order of the descriptors is determined by the arrangement of the facets in the classification scheme. The most important advantage of the faceted approach is that it allows a more detailed description of classification objects than hierarchical classification systems do by the allocation of divers coordinated descriptors. Each descriptor reflects another different aspect of the object. The faceted approach serves as the conceptual basis for this paper.

2.3 Methodological aspects

The faceted classification approach chosen provides the possibility to represent manifold meanings of complex facts by describing the classification object from different perspectives. The consideration from varying points of view is necessary to reach an extensive and comprehensive characterization of the classification object.

The acceptance and utility of a classification tool depends upon the content of the classification scheme and its representation to the user. The content is determined by the descriptors resp. the descriptor classes while the presentation form is addressed by the (graphical) user interface and menu-selection technique.

With regard to the content of the classification scheme either

- a top-down approach,
- a bottom up approach or
- a mixture of both

may be applied to define the classification scheme. When a top-down approach is chosen, the process can be structured in the following subtasks:

1. Description of the classification subject area.
2. Collection of already existent classification descriptors that may be applicable for the 'new' classification scheme.
3. Determination of relevance of each descriptor class.
4. Check of correctness of descriptors and the grouping principles. This check is conducted with the help of the criteria described further below.
5. Necessary modifications of the descriptors and descriptor classes are completed.
6. Decision upon the graphical representation of the classification scheme, i.e. graphical arrangement of descriptors and descriptor classes
7. Determination of derivation rules.

Similarly, the bottom up process may also be divided into subtasks:

1. Selection of a representative set of information objects that are to be classified according to the classification scheme to be developed.
2. Description of those information objects by several people.
3. Analysis of the descriptions and reduction to simple terms, i.e. the descriptors.
4. Grouping of descriptors to descriptor classes according to the criteria described further below.
5. Verification of the correctness and relevance of the descriptors and descriptor classes.
6. Description of the classification subject area.
7. Decision upon the graphical representation of the classification scheme, i.e. graphical arrangement of descriptors and descriptor classes.
8. Determination of derivation rules.

Both distinct approaches may be combined and merged together resulting in a mixed approach.

2.3.1 Requirements for descriptors and descriptor classes

The task of identifying and defining descriptors is an intellectual process for which only guidelines may be formulated. Subsequently aspects such as the requirements for descriptors resp. descriptor classes, grouping principles for descriptor classes and the discussion of different types of descriptors are analyzed. Conceptually, the following requirements for descriptors may be formulated. Descriptors should be

- adjusted to the terminology of the application area.
- concise, single-word terms,
- intelligible to every user of the classification tool,
- distinguishable and disjoint within a class,
- unique within a classification scheme, and
- provide both interpersonal and intertemporal stability.

It is of great importance for the efficiency and accuracy of a classification scheme to incorporates 'special descriptors' which are applied in case the existing descriptors of one class are not considered to be appropriate to describe the classification object satisfactorily. The following three types of 'last-resort-descriptors' are conceivable. These descriptors could be

- 'not relevant',
- 'not yet determined' or
- 'anything else'.

The first two options violate the required completeness condition of the classification process, i.e. one meaningful descriptor of each descriptor class has to be assigned to the classification object before it may be saved. The other option 'anything else', however, leads to a complete classification and to closure with respect to the descriptor class. Similarly, descriptor classes should be

- complete and homogeneous with respect to the classification facet,
- comprise only few descriptors,
- have expressive names referring to the content of the class,
- not overlapping and
- provide both interpersonal and intertemporal stability.

2.3.2 Grouping principles for descriptor classes

The structure of the classification scheme is predominantly defined by the logical relationships between descriptors and descriptor classes. These logical relationships determine the grouping of individual descriptors into descriptor classes. Although the grouping of descriptors into descriptor classes is mainly a creative and intellectual process, some general

rules and hints for the selection of descriptors and their grouping into classes can be described. Different types of logical relationships may be discerned such as

- contrary or opposite terms,
- partitive relations,
- aggregation relations,
- generalization/specialization relation,
- time-related sequence relation (ranking relation) and
- application-oriented descriptors.

Classes formed according to the 'contrary/opposite' rule consist of exactly two mutually exclusive descriptors. Examples for this kind of relationship are: yes/no, internal/external, stationary/movable or static/dynamic.

The partitive relationship represents a combination of hierarchical and subset relationships. A superior concept is decomposed gradually into subordinated concepts. E.g. a year consists of halves-a-year, a half-a-year is made of quarters of a year, a quarter comprises months and so on. The essential characteristic of the partitive relationship is the stepwise decomposition of one concept into another concept of a hierarchically lower level. This lower-level concept in its turn provides the basis for the subsequent decomposition process.

The aggregation relationship also aims to represent subset relations. But contrary to the partitive relationship, it looks at only one concept. The instances allowed by the aggregation relationship are: single elements, groups of elements (more than one, but not all), or all elements (e.g. all products, a group of products, a single product). The generalized term is the name of the descriptor class that contains the specialized terms as descriptors.

The ranking relation can be seen as a special kind of the generalization/specialization relationship. Such a relation is given if there is a generally accepted ordering between concepts. An example for the ranking relationship is the distinction of a time period in previous period, this period, and following period.

In contrast to the generally applicable grouping principles discussed so far, the application-oriented descriptors are more specific in reflecting facets of the underlying application.

2.3.3 Types of descriptor classes

The descriptor classes defining the classification scheme may be discerned in

- original descriptor classes and
- derived descriptor classes.

Original descriptor classes are independent of the occurrence of descriptors in other classes. Each original descriptor class is represented by a column in the classification scheme.

Within the classification process, exactly one descriptor of each original descriptor class is logically related with an AND relation with other descriptors of original descriptor classes. Descriptor classes that are dependent on other descriptors are denoted as derived. They are located on the right part of the classification scheme. Derived descriptor classes may be discerned according to three cases of dependency in

- derived descriptor classes dependent on exactly one descriptor within one descriptor class (monohierarchical structure)
- derived descriptor classes dependent on more than one descriptor within the same descriptor class (polyhierarchical structure) and
- derived descriptor classes dependent on more than one descriptor in different (at least two) descriptor classes (derivation rule specification)

The hierarchical structure of the classification scheme is depicted in figure 1.

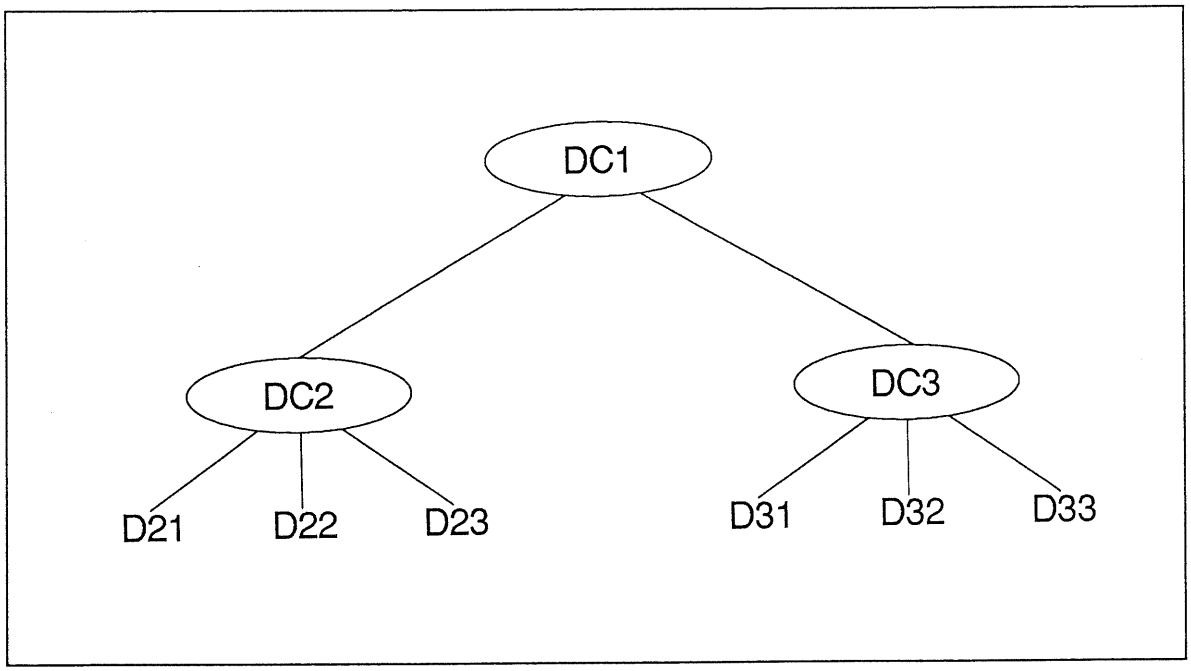

Fig. 1: Hierarchical structure of descriptor classes

In the first case sub-descriptor classes are subordinated to a specific term with the objective to further detail it. In the CODE terminology descriptors of such a complex type are named "sub-descriptor classes", the term "descriptor" is only used with respect to those elements that are not further broken down into other descriptors; they are the leaves of each branch in the tree given in figure 1. The root of the descriptor tree is termed original descriptor class. It is assigned no "upper" class, but at least two descriptors or sub-descriptor classes. Besides, the figure shows that within one descriptor class the descriptors are arranged monohierarchically.

In the second case (polyhierarchical dependency) a descriptor class may be dependent upon more than one descriptor within the same descriptor class. As being disjoint, the descriptors within one class are related to each other by a logical "exclusive-or"-relationship, i.e. within one class exactly one descriptor is marked during the classification process.
The third case refers to the dependency of a class on terms belonging to different (original or derived) descriptor classes. The main difference compared with the situation described in the second case is that the selected terms belong to distinct classes and are linked to each other by the Boolean operator "AND" (not by an exclusive "OR"). Rules specifying the derivation conditions have to be defined for accessing the derived descriptor class.

3. DATA STRUCTURE AND FUNCTIONALITY OF CLASSIFICATION SCHEME

This section describes the basic data and functional structure for the classification system and the classification methodology. The goal is to specify a general classification system which can be used for the classification of data elements as well as for the classification of information objects, data models and other items. Subsequently, we shall describe the basic functions and information objects.

3.1 Information object types

The information object types and the resulting conceptual data structure of the classification system are shown in figure 1. The notation used is the Scheer Extended Entity Relationship Model[15]. The model has been created using MSP's ManagerVIEW tool. The model can easily be transformed into relational, binary E-R or network data base structures. The data structure comprises the following information object:

Object types:

classification object
A classification object is each single item which is to be classified. It is always of a certain type, the classification object type. Examples for classification objects are (the type is in brackets): Machine-number (data element), order acceptance (function), orchid (flowers).

Classification object type
Sets of similar classification objects belong to a specific classification object type. If classification objects belong to the same classification object type, then they are classified using the same classification scheme of descriptors. Examples for classification object types are: data element, function, data model, information object.

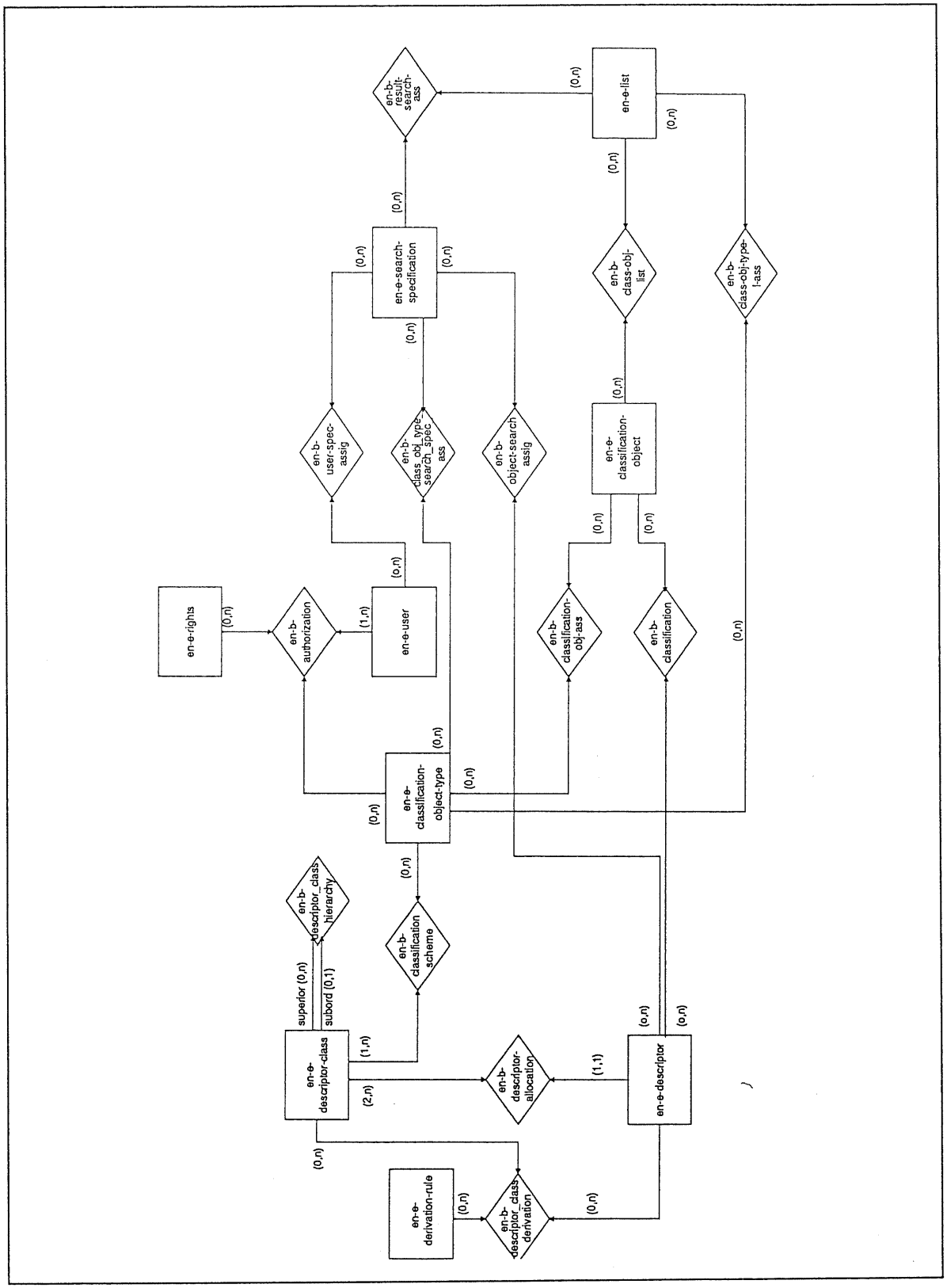

Fig. 2: Conceptual data structure for the classification system

Remark: The classification object type could also have been regarded as an initial descriptor class consisting of a descriptor for each classification object type. That the classi-

fication object type has been introduced, however, is results from special functions which it needs and that it often directly can be derived from already existing information.

Descriptor

Descriptors are standardized terms used to describe classification objects. Descriptors always belong to a descriptor class. For classification, these terms are allocated to classification objects. Out of each descriptor class, a classification object can only have allocated at maximum one descriptor. For search purposes, classification items are identified, that have the specified set of descriptors allocated to them. The terms displayed on the classification schemes are abbreviations, designed to be meaningful within the context they are displayed in. Hence a descriptor in the descriptor class "relation to persons" might be listed as "customer". The full descriptor term then signifies "related to the person customer". Examples for a descriptor are: number, code, related to person, related to products, update period: weekly, responsible: headquarters etc. Not every field which is listed under a descriptor class corresponds to a descriptor. Some represent descriptor classes which are hierarchically dependent upon superior descriptor classes. These sub-descriptor classes can not directly be allocated to a classification object. Instead, one of the descriptors which belongs to the sub-descriptor class has to be selected and allocated to the classification object (see descriptor hierarchy).

Descriptor class

A descriptor class consists of a fixed set of descriptors and/or sub-descriptor classes. This also applies to each sub-descriptor class. A descriptor class groups descriptors which represent one specific view of the classification object. The descriptors belonging to a descriptor class are related by logical relationships as Opposition, Generalization/Specialization, Subset etc.[16]

Remark: The dependency between descriptor class and sub-descriptor classes is stored within the Descriptor hierarchy.

Derivation rule

A rule specifies a relationship between descriptors and *a* descriptor class. Some descriptor classes are only relevant if another descriptor or a set of other descriptors have been selected as relevant for the classification object. Each rule specifies one such dependency. Example: The descriptor class "sum related to person" is only relevant if the descriptor "sum" which belongs to the class "mathematical operation" has been selected. The rule is specified as: if "sum" then descriptor class "sum related to person". See also: descriptor class derivation

Remark: A descriptor class derivation rule can consist of several descriptors which are defined as being connected by "and" and which lead to exactly one derived descriptor class.

List

The classification system generates different kinds of lists. A list consists of different elements. List elements are for example classification objects, descriptors, descriptor classes and lists themselves. A special case of list is the search result list. A result list is generated by a search process and is connected to a certain search specification.

Rights

Rights are the tool actions a user or a user group is allowed to activate within the tool. Examples are: System configuration, system administration, use tool to search classification object, classify data elements, modify the classification of a data element.

Search specification

The exact definition of search conditions is a prerequisite for an efficient information retrieval. Within the search specification the user defines what kind of information objects he wants to retrieve from the data dictionary.

User

A user is either a person who works with the classification system or a specification of a user group. Examples for users are: Jim Peters, Tom Jones, Application Department, Guests.

Relationship types:

Classification object-list assignment

Classification objects may be contained in lists made up for different purposes, e.g. a result list of a specific request or a listing of objects to be classified. The assignment of specific objects to concrete lists is shown in the classification object list.

Classification object type-classification object-assignment

A classification object is always of a certain type. The assignment of a classification object to its type is represented by Classification object type-classification object-assignment.

Classification object type-list assignment
This assignment allows to retrieve what classification object types belong to what list.

Classification object type-search specification assignment
This relationship type describes which type of classification object is concerned by a certain search specification.

Classification scheme
Complete classification structure consisting of the components descriptors, descriptor classes and derivation rules. It is assumed that this information may be derived from the descriptor class information

Descriptor allocation
Each descriptor has to be assigned to exactly one descriptor class and a descriptor class comprises at least two descriptors.

Descriptor class derivation
A derivation rule is intended to model logical relationships between one or more descriptors and derived descriptor classes. The relationship descriptor class derivation contains the information out of which descriptors one derivation rule consists and which descriptor class it is applied for.

Descriptor class hierarchy
A hierarchical relationship between descriptor classes and a sub-descriptor class expresses the fact that one descriptor class may consist of descriptors as well as sub-descriptor classes. The descriptor class hierarchy allows the introduction of 'recursive' descriptor classes.

Descriptor-search specification assignment
The selection of descriptors is the most important step to define a search specification. This relationship types contains the descriptors which belong to a certain search specification.

Search specification-list assignment
The result of a search specification is a list. This relationship between the search specification and its resulting list is described by the relationship type "search specification-list" assignment.

User-rights assignment
Each user can have different rights to use the classification system. The concrete rights a certain user has are defined by the relationship type "user-rights assignment".

User-search specification assignment
This relationship type describes the user who has defined a certain search specification.

3.2 Functional hierarchy

The hierarchy of functions as conceived for the classification tool is shown in figure 2. The functions are described in detail in subsequent chapters.

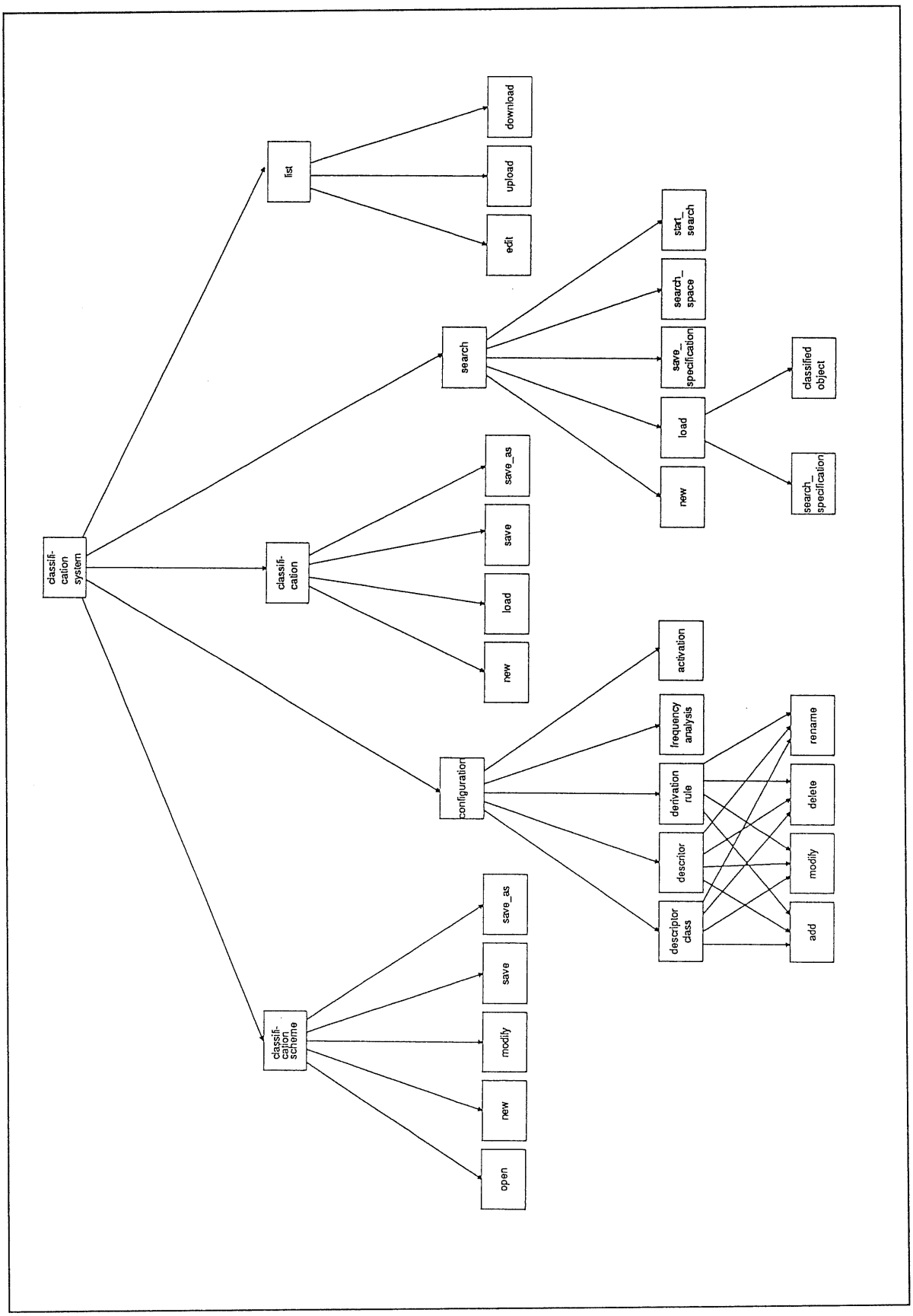

Fig. 3: Hierarchy of classification system functions

4. CONFIGURATION OF THE CLASSIFICATION TOOL

4.1 Layout and structure of a classification scheme

Beside the logical grouping of descriptors into classes as discussed in the preceding chapter, the display and arrangement of these classes in the classification scheme has to be decided upon. The ordering or graphical arrangement of descriptor classes may lead to different classification results, i.e. the question of optimality of a classification scheme arises. Although there are no proven criteria to measure the degree of optimality some guidelines for arranging descriptor classes may be formulated. Original descriptor classes are mandatory for all instances of a particular classification objects type. Therefore, it is advisable to arrange them on the left-hand side of the classification scheme. They are distinguished in colour/brightness from the derived descriptor classes, which are to be located at the utmost right part of the classification scheme. Application-oriented descriptor classes should be placed between the original descriptor classes and the derived descriptor classes.

The ordering of the descriptors within the classes can be done by the user with the help of the classification tool. Ordering principles for descriptors may be:

- ❑ the alphabetic order,
- ❑ the importance and frequency,
- ❑ increasing level of detail (abstraction) or
- ❑ decreasing level of detail (abstraction).

Since the classification tool is intended to be utilized by different users and various applications, such as data structure documentation, enterprise type classification, simulation component classification etc. the configuration functionality should allow the application-specific definition and tailoring of descriptor classes. Nevertheless, it is advisable to present a standardized 'classification screen' to the user. Such a uniform interface facilitates the classification process considerably.

4.2 Procedural description of configuration process

In the course of the configuration process the enterprise-specific classification environment is defined and tailored to the requirements of the classification object types. To introduce a new classification scheme the following subtasks have to be performed:

1. Re-set the classification scheme display to get the default classification grid. This is achieved by selecting the 'new' option on the classification scheme pull down menu.

2. Enter the configuration menu in the main menu bar and select any of the three constituting elements of the classification scheme: descriptor class, descriptor, derivation rule.
3. Adding descriptors resp. descriptor classes can be done by either typing in manually the descriptive information or by browsing through a list of already existing descriptors resp. descriptor classes. Before a descriptor may be added, the descriptor class it belongs to has to be introduced.
4. Descriptor classes and descriptors are displayed on the default classification grid in standardized boxes. The position of these boxes can be changed by clicking on the box and moving it on the screen
5. Derivation rules for derived descriptor classes may be added by selecting the derivation rule option and specifying the derivation condition by mouse clicking. Each derivation condition is saved to the meta-structure of the classification tool.
6. Upon finishing the definition of a classification scheme, the save as option of the classification scheme pull down menu is selected and the appropriate dictionary and host scheme entered.

The process of modifying an existing classification scheme that has already been used for classification purposes is similar to the described procedure for introducing a new scheme except that a semantically complex re-classification process of already classified objects is required. With respect to the required interactive tool functionalities no distinction is required.

4.3 Definition of classification scheme

The process of defining a classification scheme as outlined above includes tasks such as the introduction, editing, modification and deletion of new descriptors resp. descriptor classes. Beside, the graphical arrangement of the descriptors and descriptor classes has to be determined and derivation rules have to be defined. To construct a classification scheme for each of its components (i.e. descriptors, descriptor classes, derivation rules) the operations

- ❑ add,
- ❑ modify,
- ❑ delete and
- ❑ rename

may be executed. To activate these functions first the component upon which this function operates has to be selected and then the function itself is activated. Adding components to a new classification scheme can either be done by

- introducing genuinely novel components or
- by re-using existent components of other classification schema.

Introducing novel components requires the user either to type in the information or - in the case of a derivation condition - to specify it interactively on the screen. Descriptor classes and derivation rules already used by other classification applications may be re-used as long as no re-grouping of descriptor classes or modification of derivation rules is required. In case existing descriptors or descriptor classes are re-grouped or new derivation conditions for derived descriptors are formulated then a re-classification takes place. The deletion, modification and renaming of components is based upon interactive, mouse driven clicking. Only in the case of the derivation rule a window is popped up to indicate the derivation rule to be deleted or modified. The graphical arrangement of descriptors and descriptor classes is done interactively. In the following, the functions involved in the definition of a classification scheme are described in detail.

4.3.1 Add descriptor class

Descriptor classes have to be introduced before individual descriptors may be assigned to them. A descriptor class is created according to a grouping principle. It has class specific attributes such as header and number of descriptors per class. Descriptor classes may be derived according to derivation conditions. In the case of a monohierarchical derived descriptor class a class may be subordinate to a hierarchically superior descriptor. These descriptors are of complex type, i.e. they are further detailed by instances of a descriptor class on a hierarchically lower level. It is possible to browse through already existing descriptor classes. This allows the detection of semantically similar descriptor classes.

4.3.2 Add descriptor

Individual descriptors cannot be introduced unless the descriptor class they belong to has been introduced beforehand. The derivation status and the subordinate descriptor class name both indicate whether the descriptor is part of a derivation condition for derived descriptor classes. Also, descriptors may be selected from a list that can be browsed. A descriptor is added by first clicking on the descriptor option and then selecting the add option. Then a window template for describing the descriptor pops up. Moving the descriptor box with the mouse cursor to the class it is to be assigned to leads to an automatic assignment to and a data update of that class description.

4.3.3 Add derivation rule

A derived descriptor class can be interpreted as a conditional descriptor class that can only be accessed by choosing an appropriate combination of descriptors. A derivation rule is assumed to consist at least of one descriptor. One descriptor may be used more than once in different derivation rules. Several descriptors of one descriptor class may be used for specifying alternative derivation rules for a particular derived descriptor class, i.e. an OR relation between descriptors of one class with respect to a derived descriptor class is allowed. The logical relations of descriptors determining a derivation rule are expressed by boolean expressions. They may be represented by graphical flowcharts. An example of such a logical chart is depicted in figure 4.

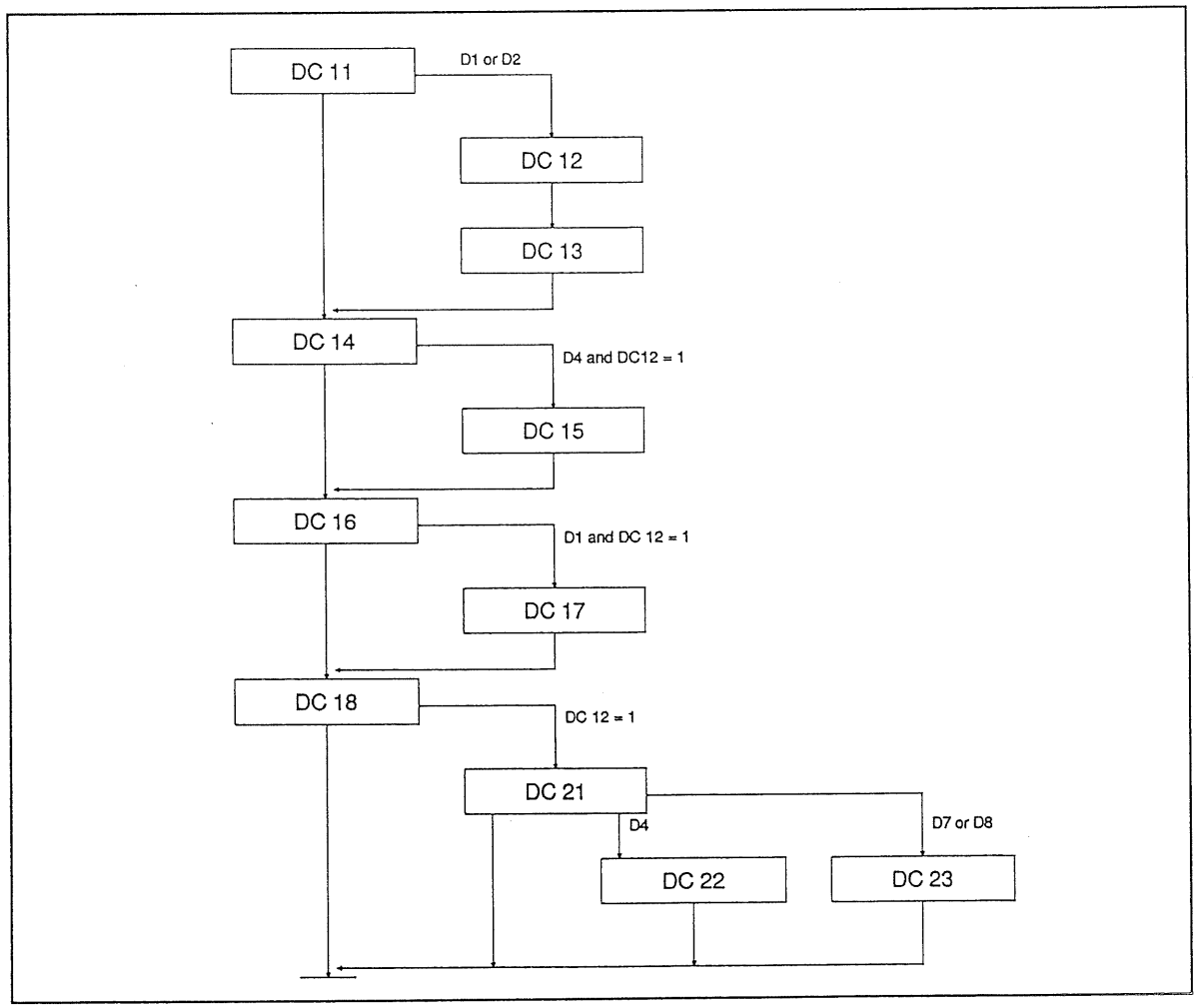

Fig. 4: Logical flowchart for descriptor class interrelation

A derivation rule is defined interactively with the mouse button. The descriptors of one derivation rule - they are related by a logical AND - are selected one after the other. On the termination of the derivation rule, the header of the derived class is clicked upon.

4.3.4 Deletion of descriptor classes and descriptors

The deletion option for descriptor classes and descriptors work identically by clicking first on the respective component type and then selecting the delete option. The deletion of either a descriptor class or a descriptor is restricted if they are part of a derivation rule that is still valid. To remedy this deletion restriction the concerning derivation rule has to be removed first. The delete option is then re-entered.

4.3.5 Modification of descriptor classes and descriptors

The modification option allows to move around descriptors resp. descriptor classes on the screen and supports the graphical arrangement of the classification schema. Descriptors may be assigned to other descriptor classes. A mouse click is used to grab a descriptor resp. descriptor class.

4.3.6 Deletion of a derivation rule

To activate the deletion of a derivation rule first the derivation rule option is to chosen (marked by a tick) followed by the delete option. The pop-up window allows a selection of the derivation rule to be deleted. By removing the derivation rule from the list box the derivation rule is deleted from the meta-structure. The pop-up window also allows a search on the elements of derivation rules. For example, if a descriptor, that is part of different derivation rules, is to be deleted all concerned derivation rules may be highlighted in the derivation rule list box. Similarly, if a descriptor class is to be deleted all related derivation rules may be searched for.

4.3.7 Modification of a derivation rule

The modification of a derivation rule is done interactively by using the mouse button. When the option is activated, i.e. first selecting the derivation rule in the upper menu part and then clicking on the modify option. The derivation rule to be modified is selected from the menu box with each concerned descriptor highlighted in the classification scheme. The modification is done by highlighting resp. de-highlighting the appropriate descriptors. The

modified derivation rule is then saved to the meta-structure. The renaming of a derivation rule is considered to be a modification limited to the name.

4.3.8 Renaming of descriptor classes and descriptors

Renaming an object of the classification scheme is done interactively by first selecting the component type to rename and then selecting the rename option. After the activation of the rename option a mouse click on the component to be renamed pops up a window that allows to type in the new name of the component.

4.4 Re-definition of the classification scheme (re-classification)

The modification of a classification scheme in use is denoted as re-definition. The subsequent discussion of the re-definition process is primarily intended to illustrate some aspects and possible problems that might be encountered. Major ideas are summarized and discussed. The complexity of the actual implementation of a re-definition functionality can only be outlined at this stage of the paper. When a re-definition of the classification scheme is performed the benefits and side-effects have to be carefully considered and evaluated. A re-definition of a used classification scheme is 'harmful' for all information objects classified so far and leads, unless accompanied by a (procedurally complex) re-classification process, to inconsistent descriptions of the information objects.
The process of re-defining an existing classification scheme - i.e after information objects have already been classified - is very similar to the first-time introduction of a new classification scheme. The necessity for a re-definition may occur if a new facet of an already existing classification has to be added or during the use of a classification system. The logical rules for changing the classification scheme are inherently more complex as compared to the descriptor definition.
The re-definition of the classification scheme as conceived throughout this paper is divided into two phases: First, all the modifications to the classification scheme have to be completed yielding a logically consistent new structure. This new scheme is then - in a second step - activated, i. e. the classification strings for the classified information objects are updated by a re-classification process similar to the general classification process.

Re-defining the classification scheme includes the following task:

1. Select the 'modify' option in the classification scheme to change the modus of the configuration menu to 're-configuration'.

2. Enter the configuration menu in the main menu bar and select any of the options. The options 'frequency analysis' and 'activate' are now available indicating that the re-configuration modus has been selected.
3. Conducting a frequency analysis to get information about the relative utilization of each component (descriptors, derivation rules) in the classification scheme. The frequency analysis is optional in that it is no prerequisite for changing the classification scheme. Also, it may be applied without changing the structure at all.
4. The modification of the classification scheme comprises the basic options of the configuration menu such as
   - ❑ re-naming of descriptors, descriptor classes and derivation rules
   - ❑ adding descriptors, descriptor classes and derivation rules
   - ❑ deleting descriptors, descriptor classes and derivation rules
   - ❑ re-grouping descriptors into new classes

   Each of these modifications affects objects that have already been classified according to the classification scheme that is currently being modified. Hence, the old classification strings will not be consistent with the new ones. As a consequence one classification object classified before and after the re-definition of the classification scheme may be classified differently.
5. The modifications of the classification scheme have to be activated. This activation initiates a complete re-classification of the objects that are concerned by the re-definition of the classification scheme. To conduct a re-classification both the classification and the search module have to be accessible.
6. Upon finishing both the re-definition of a classification scheme and the re-classification, the 'save as' option of the classification scheme pull down menu is selected and the appropriate dictionary and host scheme entered.

Following, the functions involved in the reclassification process are described in detail:

4.4.1 Frequency analysis

The frequency analysis option is introduced to analyze the classification scheme in terms of the relative utilization of each descriptors resp. derivation rule. With its help the information about the relative frequency each descriptor is displayed. This information guides the re-classification process. Descriptors that are rarely used may be unnecessary or irrelevant and may hence be discarded. If, on the other hand, a descriptor is heavily utilized this might indicate its too general definition or too a broad scope. In the first step, it has to be determined whether single descriptors will be analyzed or derivation rules. Subsequently, the threshold level has to be defined. This threshold sets a minimum or maximum number which the relative frequency of a descriptor resp. derivation condition may not exceed

resp. may not be below. After the definition of the analysis scope, a list is created containing all the descriptors the relative frequency of which is within the defined range. This list is mandatory in that it determines exactly what descriptors are subject to re-grouping in succeeding re-classification steps. The administrator has the opportunity of adding and voiding some of the descriptors suggested to him before they become mandatory for the re-classification. Nevertheless, it is possible to start a re-classification process without having conducted a frequency analysis beforehand.

4.4.2 Scheme modification

The modification of the classification scheme strongly resembles the scheme definition. Identical windows are applied for adding elements and the user interaction for deleting, re-arranging and renaming components are similar. The main difference lies in the complex rules that have to be obeyed before any modification of the existing classification scheme is conducted and comes into effect. A detailed description of the procedural implementation of these rules is beyond the scope of this paper. Nevertheless, some of these rules will be described subsequently as a starting point for a further discussion.

4.4.3 Edit derivation rule

Editing a derivation rule may be either adding a new derivation rule and deleting or modifying an existing one. Each of these cases implies a slightly different re-classification to take place in case the 'activate' option is chosen subsequently. After the addition of a new derivation rule, each object that fulfills the 'access path' will be assigned an additional descriptor of the derived descriptor class. Furthermore, the status of the descriptors of the access path has to be updated. Similarly, the deletion and modification of a derivation rule result in an update of the corresponding entries after the re-classification has been conducted.

4.4.4 Rename scheme component

To change the name of a descriptor resp. descriptor class the user clicks on either a descriptor or a descriptor class header and changes its name. It has to be ensured that no name appears more than once within a given classification scheme.

4.4.5 Activation

After all the modifications of the classification scheme have been defined, they have to be activated. This is performed by selecting the 'activate' option. The activation process has two alternative modes that determine the effects of the modifications introduced. The administrator can decide that the changes of the classification scheme are either applied to all information objects that have already been classified or that the new classification scheme will only be applied in succeeding classification processes. While the first choice requires complex re-classifications the latter leads to inconsistent classification results depending on the date of classification. Hence, the intertemporal stability of the classification scheme is not guaranteed.
The first option is preferable in terms of a consistent state of the classified objects. Compared to a complete 'first-time classification' updating already classified information objects can more easily be realized. If, for instance, a new descriptor is added to a class, the update process is limited to this class. Then, in a limited search process, all the information elements classified by descriptors of that class have to be retrieved. Unless certain rules for re-assigning the new descriptor (and replacing the old ones) have been formulated, each information object has to be re-classified in a scaled-down re-classification process as compared to the complete classification as described. When a re-grouping of descriptors takes place, this process has to be conducted both for the descriptor class the descriptor is removed from and for the descriptor class the descriptor is moved to. When a new descriptor class on the highest classification level (i.e. AND relation) is introduced, each information object that has been classified before has to be retrieved to add one descriptor of the new class to it. It is conceivable to facilitate this process by introducing a class 'not yet determined' to the new descriptor class and assign this descriptor value by default to all the former information objects. The renaming of descriptor classes can be conducted automatically.

4.4.6 Concluding remarks

Considering the re-definition and subsequent re-classification process as a whole, it is impossible to develop general rules for an automated re-classification. How a re-classification is actually conducted depends heavily on the relation of the descriptors to each other within a descriptor class and the operation that is conducted. The degree of the procedural complexity of a re-classification is also determined by the 'quality' of the existing classification scheme and the objective pursued with a re-classification. If a bad structure is to be transformed into a consistent structure of high quality this can hardly be achieved by a re-classification. In this case it is advisable to define a completely new classification scheme and conduct the classification from scratch. The benefit of incorporating

a re-classification option in the classification menu lies in its support of an evolutionary approach to classification.

After having discussed the introduction and modification of a classification scheme the classification process itself is analysed in detail subsequently.

5. CLASSIFICATION PROCESS

5.1 Procedural description of classification process

The primary objective of the classification is to describe the semantics of a classification object by the allocation of descriptors to allow a later retrieval. The consideration from varying points of view is necessary to reach a complete, definite, and comprehensive characterization of the classification object. Each such perspective is reflected by a specific descriptor class and its descriptors. In the course of the classification process a classification object is described by the allocation of descriptors. Objects of the same type have to be classified using the same classification scheme to ensure their comparability.
The classification process itself covers several consecutive activities:

- ❑ select classification object type,
- ❑ select classification object,
- ❑ classify selected object,
- ❑ verify classification,
- ❑ save object classification.

Subsequently, the functions of the classification process are described in detail.

5.2 Selection of classification object type

At the beginning, the initial screen of the classification tool is exposed to the user. The classification functionality is activated by clicking the "classification scheme" item in the menu bar and selecting the option "open" in the pull-down menu. The "open" option leads to another (pop-up) window listing the classification object types. The user selects that type he intends to classify by marking it using the mouse. The classification scheme for the selected type appears on the screen. It contains two types of descriptor classes, that are distinct in colour. Original classes are bright, derived classes dark. It is possible to choose descriptors only in bright classes. The mode (classification or search), however, is not yet determined.

5.3 Selection of classification object

To start the classification application the mode has to be set to "Classification" by clicking the "classification" item in the menu bar. The "Classification" pull-down menu appears. Then, the user is asked to indicate the object he intends to classify. It does not matter whether this object is new or already stored in the dictionary or work space, but it must be an instance of the type selected before.

The distinction in existing and non-existing objects influences the selection mode of the classification object. Both kinds of objects can be typed in manually using the keyboard if the user knows the object identifier by heart. The user selects the option "new" in the pull-down menu and enters the object identifier which is displayed in the object identifier field at the top of the classification scheme.
Classification objects that are already available in the dictionary or work space can also be selected from a list. This selection mode is favourable if the object identifier is unknown or not exactly known. The user marks the option "load" in the "classification" pull-down menu. A dialogue box is displayed on the screen. The following activities must be carried out:

- Specify list type.
  The user marks the relevant list type (dictionary or work space), and all lists of the specified type are displayed in a window.
- Select list.
  The user clicks that list name he is looking for, the classification objects of this list are shown in a window.
- Select classification object from list.
  The user marks that object he wants to classify. The identifier of the selected object is automatically entered into the object identifier field at the top of the classification scheme.

5.4 Classification of selected object

After the initial steps of specifying the object type respectively the classification object the classification process proper may start. Descriptors are assigned to the classification object. The sequence is left to the user. But, it is advisable to begin the classification process with the first column of the scheme and to continue consecutively from left to right. The classification rule says that within each descriptor class regarded to be relevant for the classification object the user has to select that term that best describes the object. Within each descriptor class exactly one descriptor must be chosen. The user marks the relevant descriptor by clicking on it once with the mouse. The colour of the descriptor field changes to distinguish the selected descriptor from others. The selection is undone by a double-click.

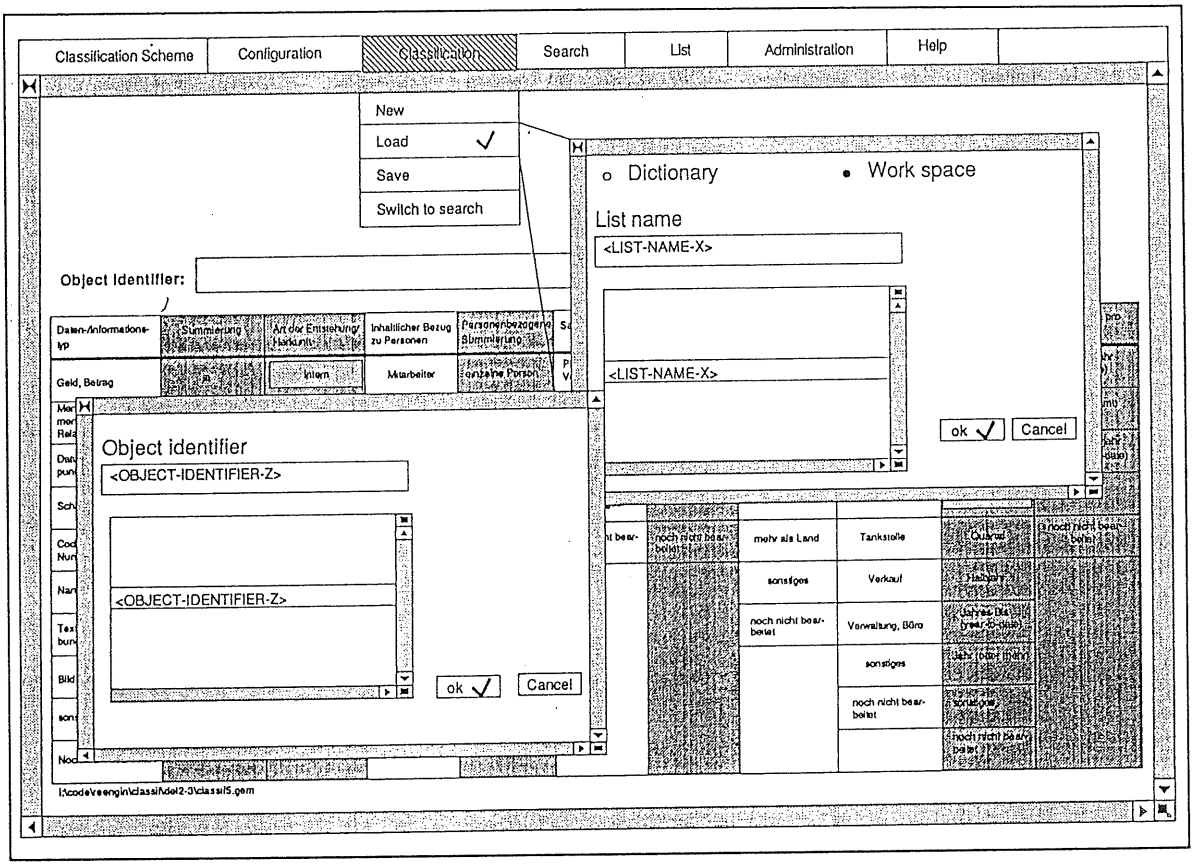

Fig. 5: Window template "Load classification object from list"

In case of marking a sub-descriptor class initially hidden from the user a window listing the respective descriptors appears on the screen. After the selection of a descriptor the window disappears. This procedure continues until there is no more sub-descriptor class. In order to show what descriptor has been chosen it is entered into the uppermost descriptor field. In case a derived class becomes relevant for the description of the classification object because its triggering descriptor(s) has (have) been marked before, the derived class becomes bright indicating the user that he must also select a descriptor within this originally dark column.
The classification process is finished if out of each bright column one descriptor is allocated to the classification object. At the end of the classification process the classification object is described by a list of descriptors.

5.5 Verification of the classification

Before the result of the assignment process is stored in the work space, it is advisable to verify the classification with respect to formal and semantical aspects. The verification covers both

- the completeness and
- the correctness

of the classification.
Completeness refers to descriptor classes (the number of descriptors), correctness relates to descriptors (the meaning of descriptors).

A classification is complete if the object is described by as much descriptors as there are relevant descriptor columns for this object. The requirement for completeness represents a formal demand. Therefore, the completeness check can be done automatically or by the user. As there is a distinction in colour between mandatory and inadmissible descriptor classes as well as between selected and not-marked descriptor fields it is easy for the user to ascertain the completeness of the classification. He only has to know that in each bright column one descriptor must be selected. The fading-in of originally derived classes is accomplished by the tool. Correctness refers to semantical aspects. Hence, it requires intellectual input and cannot be examined automatically. A classification is correct if the chosen descriptors describe the classification object in a right way. The selection of other terms would lead to a wrong classification. A selection of a descriptor can be undone by clicking the marked field twice with the mouse. Another term must be marked in the corresponding descriptor column.

Furthermore, it is absolutely indispensable for ensuring a valid classification that the user of the classification tool comprehends the meaning of the descriptor classes and the descriptors respectively. Both should be provided with expressive textual descriptions in the configuration process. Besides, knowledge about the classification object and its type is of great importance for the correct allocation of descriptors.

5.6 Saving the object classification

The finishing activity of the classification application is storing the correct object classification in a work space that may be accessed and reused by other tools or other functions of the classification tool (e.g. the search and retrieval module). The user marks "save" in the "classification" menu.
Depending on the input mode of the classification object identifier the "save"-function looks different:

- In case an object identifier has been typed in manually (new option) there is no work space specified. I.e., if "save" is selected the user is asked in a dialogue box to indicate the work space (list type) and the list (list name) the object classification should be appended to.

- In case an object identifier is entered by the "load" function the object classification is automatically saved to that work space and list it has been loaded from.

After having saved the object classification the classification process of this object is finished. If there are other objects that should be classified, the process restarts with

- the "open" option in the "classification scheme" menu if the classification object belongs to another object type or
- "new" or "load" in the "classification" menu if the object type does not change.

5.7 Switch to search

Before saving the object classification to the data dictionary it must be checked for synonyms and homonyms to guarantee the consistency and the lack of redundancy in the dictionary. This check is performed by the search module of the classification tool. The tool provides the possibility to call the search module without leaving the classification module by "switch to search". Thereby, the mode is set from classification to search.

If a classified object shall be checked for synonyms and homonyms immediately after the classification the user calls the search functionality by "switch to search". The object classification remains on the screen as basis for the search specification and may be modified interactively.

6. SEARCH

6.1 Procedural description of search process

The systematic search and localization of information objects stored in an information system can be defined as information retrieval. A special case of the information retrieval is the search of synonyms. The search of synonyms can be part of the information engineering process supported by data dictionaries or repositories. Thereby, all software engineering projects within the enterprise are forced to update their results to a central data dictionary or repository. A new data information object is only accepted if no synonyms and no homonyms for that object exist within the data dictionary. The synonym search compares the classification of the new object to the classifications of all data dictionary objects. In the case of an existing synonym the new application has to use the already existing object in order to avoid redundancies. In the case of an existing homonym the application has to determine a new name and a new identifier for the data element.
This paper does not explicitly differentiate between the search of synonyms and the information retrieval because the search of synonyms always includes an information retrieval.
The information retrieval process comprises the following main functions:

- Specify search
- Save search specification
- Define search space
- Start search
- Analyze results

At the beginning of the retrieval process the information objects which should be searched are specified. In the case of a synonym search, the user loads a classified object from the workspace of the classification module to the search module. The user can also load a former search specification as starting point for the search or he can specify the search conditions from scratch by manually selecting the relevant descriptors. Before the search is started, the search specification can be stored and the search space has to be defined. Then, the search can be started and the results of the search process are stored. The subsequent analysis phase is necessary to decide whether the relevant information objects have been found or the search process has to be started again. Following, the functions belonging to the search process are described in detail.

6.2 Specify search

The exact definition of the search conditions is a prerequisite for a successful information retrieval. The user must define what kind of information object he wants to retrieve from the information system.
The search specification entails two main functions:

- Select descriptors
- Load classification

Both functions can be used separately or combined. Concerning the starting of the search specification four cases can be discerned:

- the specification is started by manually selecting the descriptors of the classification scheme
- the specification is started by loading a former search specification
- the specification is started from the classification module
- the specification is started by loading a classified information object

The last two approaches are used to support a synonym search.

6.3 Select descriptors

The selection of the descriptor classes and descriptors is the basis for the search specification. It is possible to define logical connections between selected descriptor classes on the same level by using Boolean logic operations ("AND"/"OR"/"NOT"). A logical "AND" between two descriptor classes means that an information object will only be retrieved from the data dictionary if the search conditions concerning both descriptor classes are fulfilled. [DC 1 AND DC 2] A logical "OR" between two classes means that a data element will be retrieved if at least one of the search conditions concerning the two descriptor classes is fulfilled. [DC 1 OR DC2] "AND" and "OR" operators may also be combined for logical connections between different classes. [DC 1 AND (DC2 OR DC3)]
But it is doubtful whether logical connections other than "AND" are necessary to describe search conditions. A restriction to the "AND" connection between descriptor classes can significantly simplify the search process. Therefore the classification system defined by this paper will allow only "AND" connections between descriptor classes. The "OR" connection is only allowed for connections between descriptors belonging to the same descriptor class.
The user starts the specification by selecting descriptor class elements of the classification scheme. The classification scheme has to be determined before according to the classification object type which has to be searched. The selected elements can be descriptors or descriptor classes. If the selected element is a descriptor this means that the search condition

is only true for an information object containing this element within its classification. For the search specification, it is possible to mark more than one descriptor per class. In this case there is a logical "OR" between the marked descriptors of this class. A logical "AND" between descriptors of one class is not allowed because the classification of an information object can contain only one descriptor per class. A logical "NOT" within a descriptor class specifies that a search condition is true if an information object does not contain the selected descriptors or descriptor classes. A logical "NOT" can be transformed into a logical "OR" between all descriptors of this class which are not marked by "NOT". If the selected element is a descriptor class this class has to be further specified on its subclasses until the lowest search level is reached (which is always a descriptor) or until the search specification is stopped. If the search specification stops at a descriptor class this implies that all descriptors of the subclasses belonging to the selected class fulfil the search condition. (logical OR)
The specification of the search conditions does not require the selection of all descriptor classes. Descriptor classes which are not selected are not used for the search process.

6.4 Load classification

The search specification can be started by using a classification which is already existing. For example a classification of a certain information object can be used to start a synonym search for that object. Or a new search process can be started by using a former search specification which is modified according to new user requirements.

6.4.1 Load classified object

When a classified object is loaded the classification screen is marked according to the classification of this object. The classification can be modified or directly used to start a search process. Loading a classified information object as starting point for the search specification is mainly relevant for the search of synonyms.
During the search of synonyms, a given information object is compared to other objects in order to analyze whether there already exist information objects which differ in terms but denote the same concept. This analyses is realized by comparing the classification of the given data element (i.e. the descriptors assigned to it) to the classifications of all other information objects. The classification of the relevant information object can be inserted manually or it can be loaded if this classification is already available. For example the information objects can be stored in the classification workspace. As a next step they are loaded from the classification workspace to the information retrieval module, checked for synonyms and eventually inserted into the data dictionary.

The user has to determine the name of the classified object and the list it belongs to. This is implemented by the following pop-up window, see figure 7.

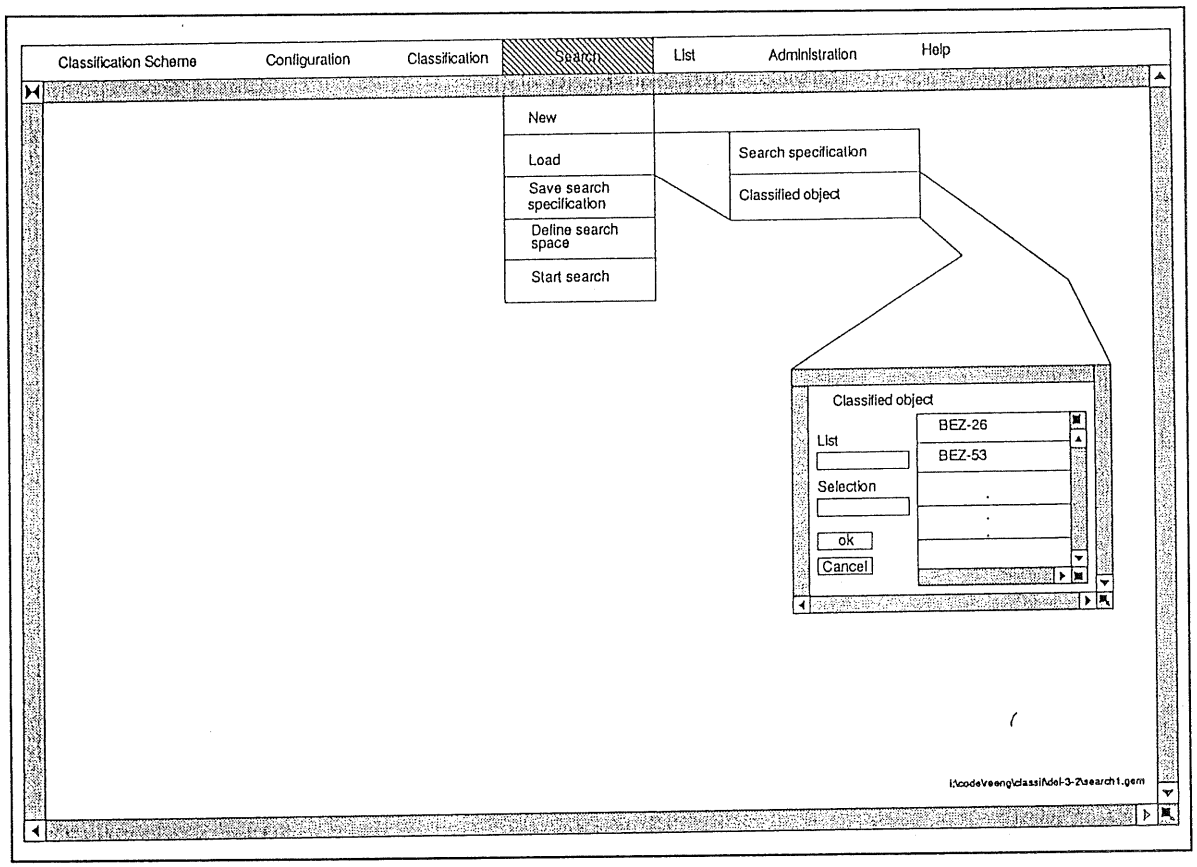

Fig. 6: Pop-up window to load classified objects from classification workspace

6.4.2 Load search specification

It is possible to use former search specifications to start a new search specification. The search specification loaded can be modified according to the new requirements of the user. The user has to select the name of the search specification and the list it belongs to. The search specification which is loaded can be manually modified by marking descriptors of the classification scheme.

The "load search specification" function is implemented by the following pop-up window, see figure 8.

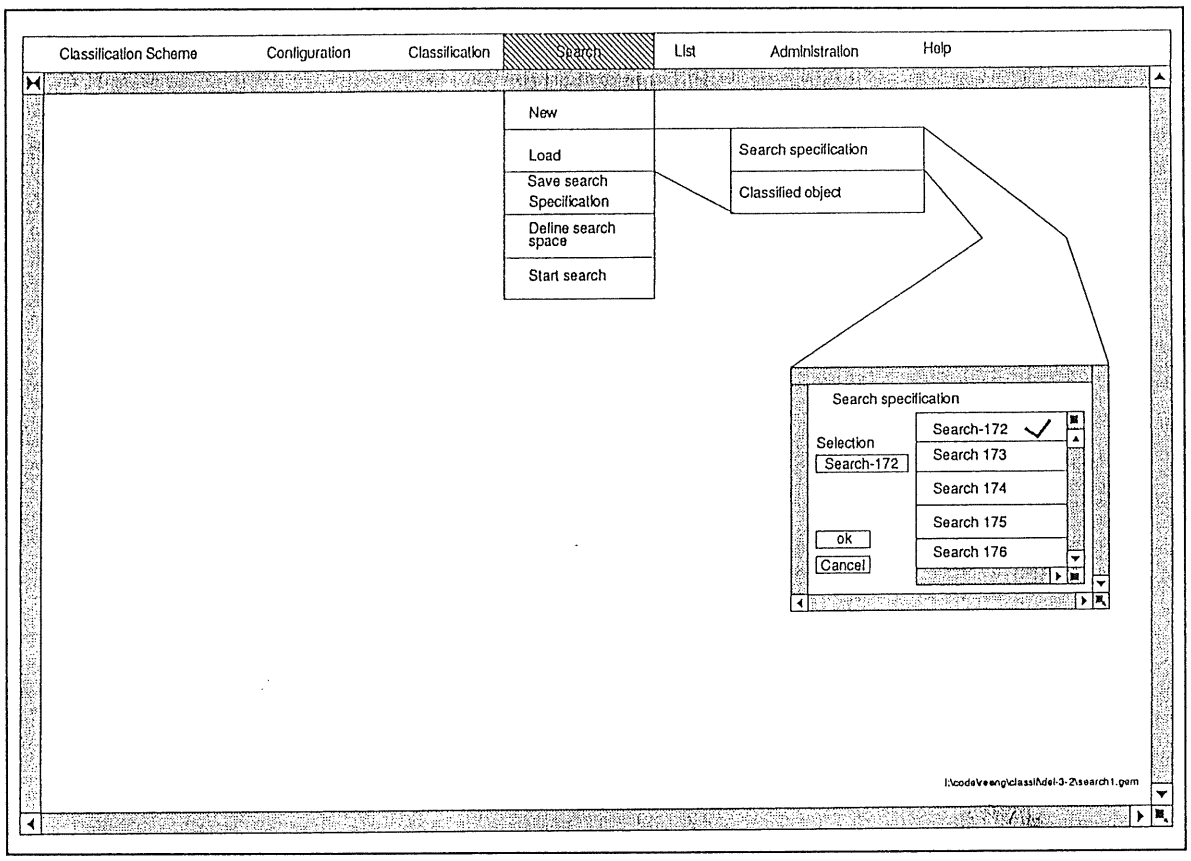

Fig. 7: Pop-up window to load former search specifications

6.5 Store search specification

A search specifications can be stored. It can be used to start the specification of a new search processes. The same search specification can be used to analyze changes of the data dictionary by comparing result lists which are generated by the same specification at different points of time.

6.6 Define search space

Before the search process is started the search space has to be defined. The search can be performed on the whole data dictionary or on certain segments of the data dictionary. The efficiency of the search process can be considerably increased by the preselection of a set of elements on which the search has to be carried out. For example, the search can base on result lists generated by former search processes. It is also possible to base the preselection of the search space on the functionality of the data dictionary. For example the data dictionary functionality can be used to generate so-called kept-data lists which can be used as search space.

6.7 Start search

In order to avoid inefficient search procedures stop conditions can be defined before starting the search. For example the size of the result list can be limited. I.e. if the number of elements yielded by the search exceeds a defined number the search process will be stopped.
After the specification of the search conditions the search process is started. The system searches information objects which meet the requirements defined in the phase before.
The result of the search process is a list with information objects (the list can of course be empty).

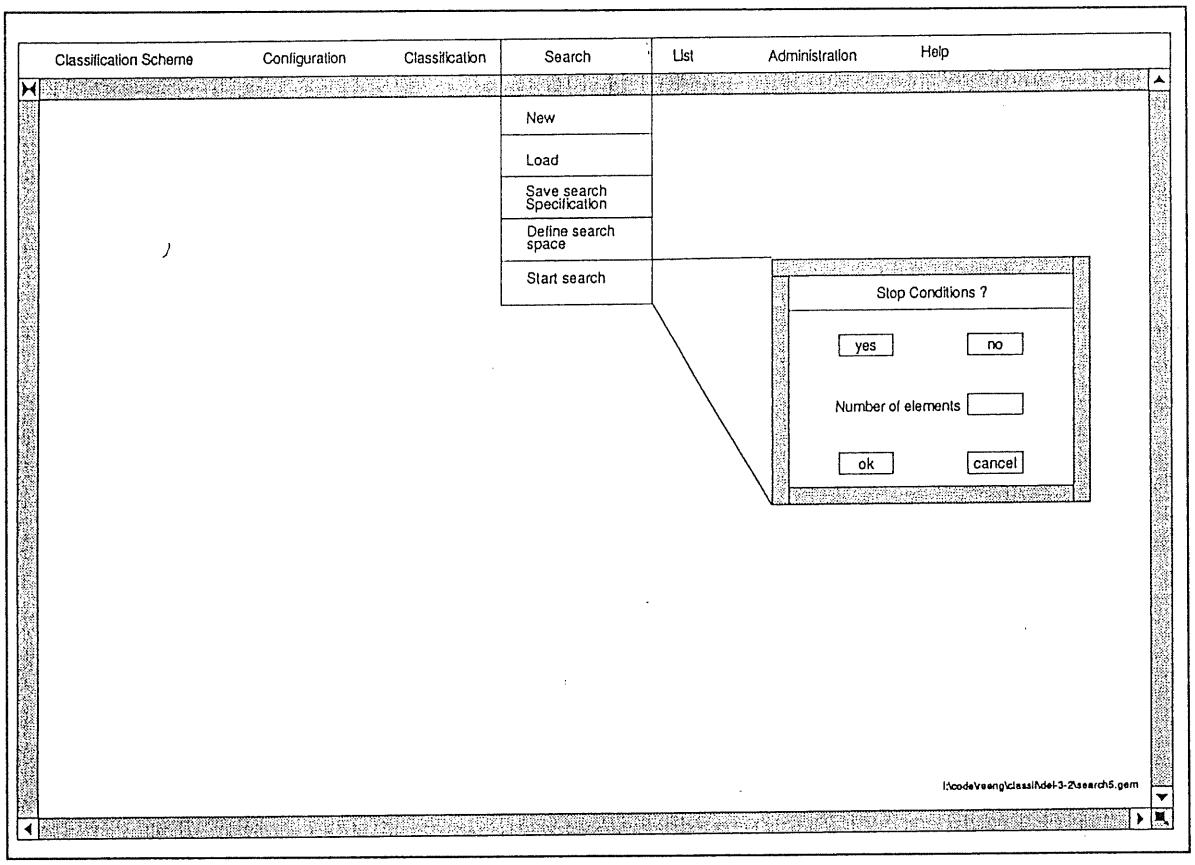

Fig. 8: Pop-up window for the definition of stop conditions

6.8 Analyze results

The main objective of the search process is to retrieve information objects respectively to find synonyms. But the results of the search function are not always the results the user expected.

There are certain measures to evaluate the quality of the results yielded by the information retrieval systems:[17]

y1 = number of information objects which have been correctly selected.
y2 = number of information objects which have not been selected but which are relevant for the search
z1 = number of information objects which have been selected but which are not relevant for the search
z2 = number of information objects which have not been selected and which are not relevant for the search.

Precision (relevance ratio) = [selected relevant information objects] : [all selected information objects]
P = [y1] : [y1 + z1]

Output ratio = [selected relevant information objects] : [all relevant information objects]
O = [y1] : [y1 + y2]

Fallout ratio = [selected irrelevant information objects] : [all irrelevant information objects]
F = [Z1] : [Z1 + Z2]

The analysis function can not only support the analysis of the results of the information retrieval system. It can also be used to analyze and evaluate the structure of the classification system i.e. the descriptor classes and descriptors and their inter-relationships. E. g. the fact that certain descriptors are selected most of the time whereas other descriptors are not used at all can indicate the necessity of a redefinition of the classification system. Because of the problems arising from a redefinition of the classification system this analysis function should mainly be performed during the test phase of the classification system (see chapter: configuration of the classification tool).

The result list can be structured in different ways: a list can contain only the names of the selected elements or it can also contain the descriptors assigned to the selected elements. A list with the names and the descriptors of the selected elements can be necessary if the search specification allows to retrieve information objects which do not have exactly identical descriptors. The lay-out of the lists determines the efficiency of the analyze function.

The descriptors can be represented by abbreviations or keys which refer to the descriptor classes they belong to. For example "1.1" means the first descriptor of the first descriptor class or "1.2.3" means the third descriptor of the class number two which is subordinate to class 1.

The disadvantage of the representation by keys is that the user can not recognize the meaning of the descriptor by reading the key. Therefore, the classification system should allow result lists with the names of the descriptors.
The analysis of the descriptors supports the decision whether an element has been correctly retrieved or not i.e. information objects of the result list are analyzed concerning their relevance. Additionally, the user has to think about whether there could be more relevant information objects which are not part of the result list. The analysis is a highly interactive process because only the user can decide about the relevance of information objects of the result list. The analysis function can also involve a manipulation of one or several result lists. E.g. elements of the result list can be dropped or the union or intersection of two or more result lists can be generated.

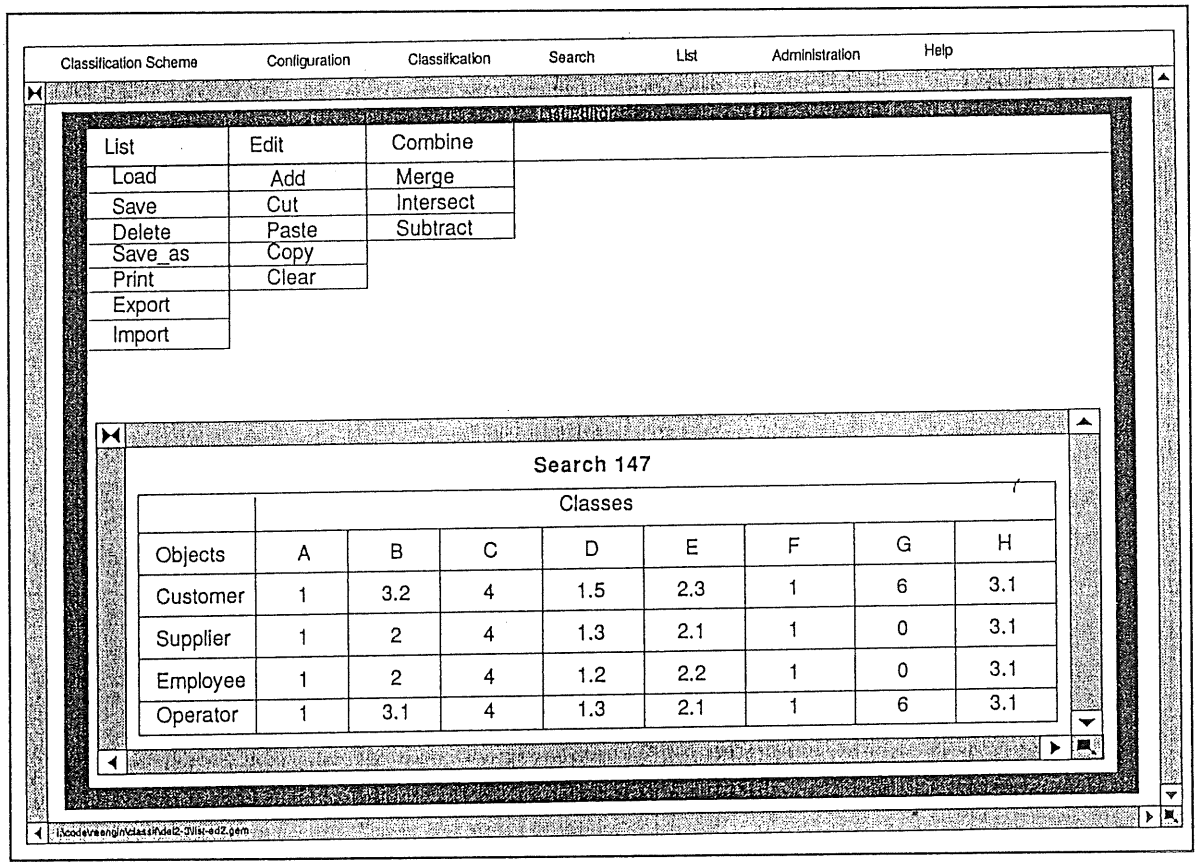

Fig. 9: Result list with descriptor keys

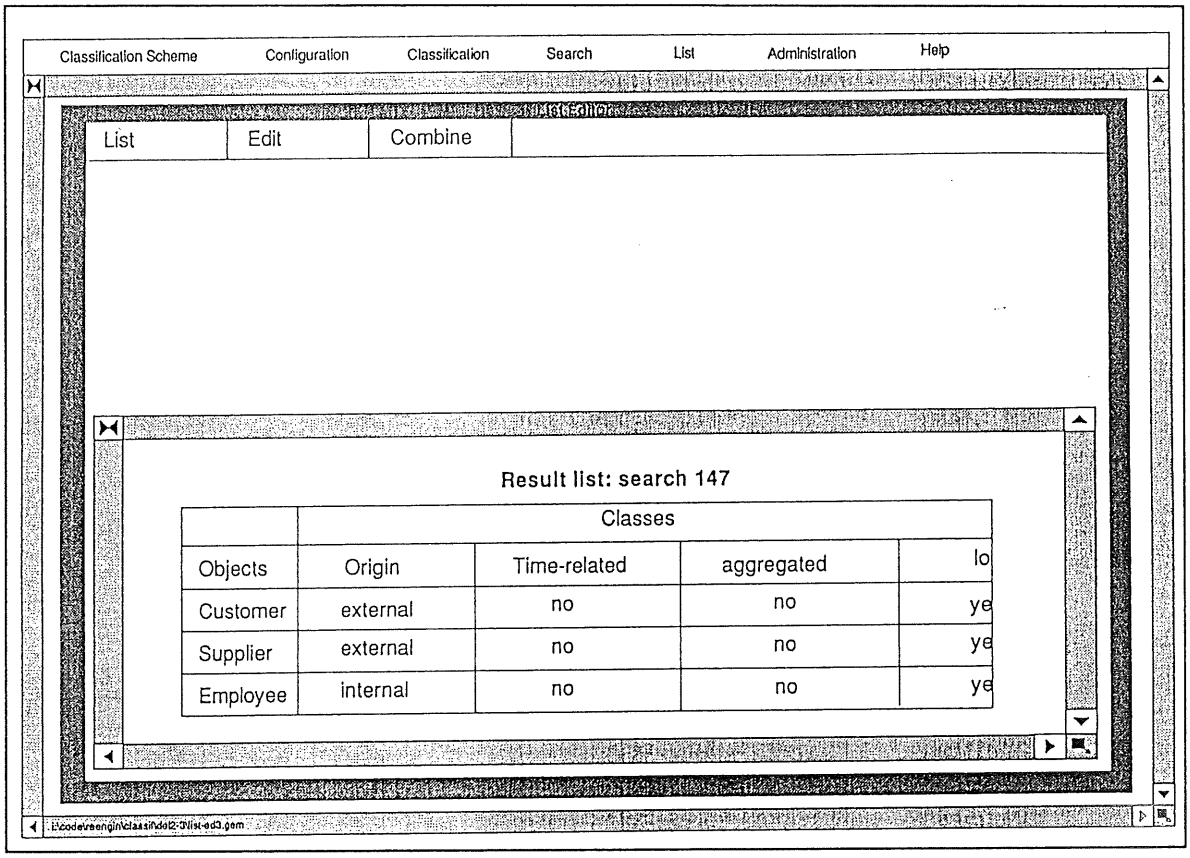

Fig. 10: Result list with descriptor names

The result list is the input for the analysis phase. During the analysis phase the result lists can be stored, modified and compared to other result lists.

7. SAMPLE CLASSIFICATION SCHEME FOR INFORMATION OBJECTS

This chapter presents a sample classification scheme for information objects. The classes and descriptors of the scheme are derived from theoretical considerations and experiences from one of the CODE testbed partners. This classification scheme can be utilized to describe the elements of a data reference model and the customized data model.

In the following, each class and its descriptors of the classification scheme are shortly defined. The classification scheme is depicted by figures 11 and 12.

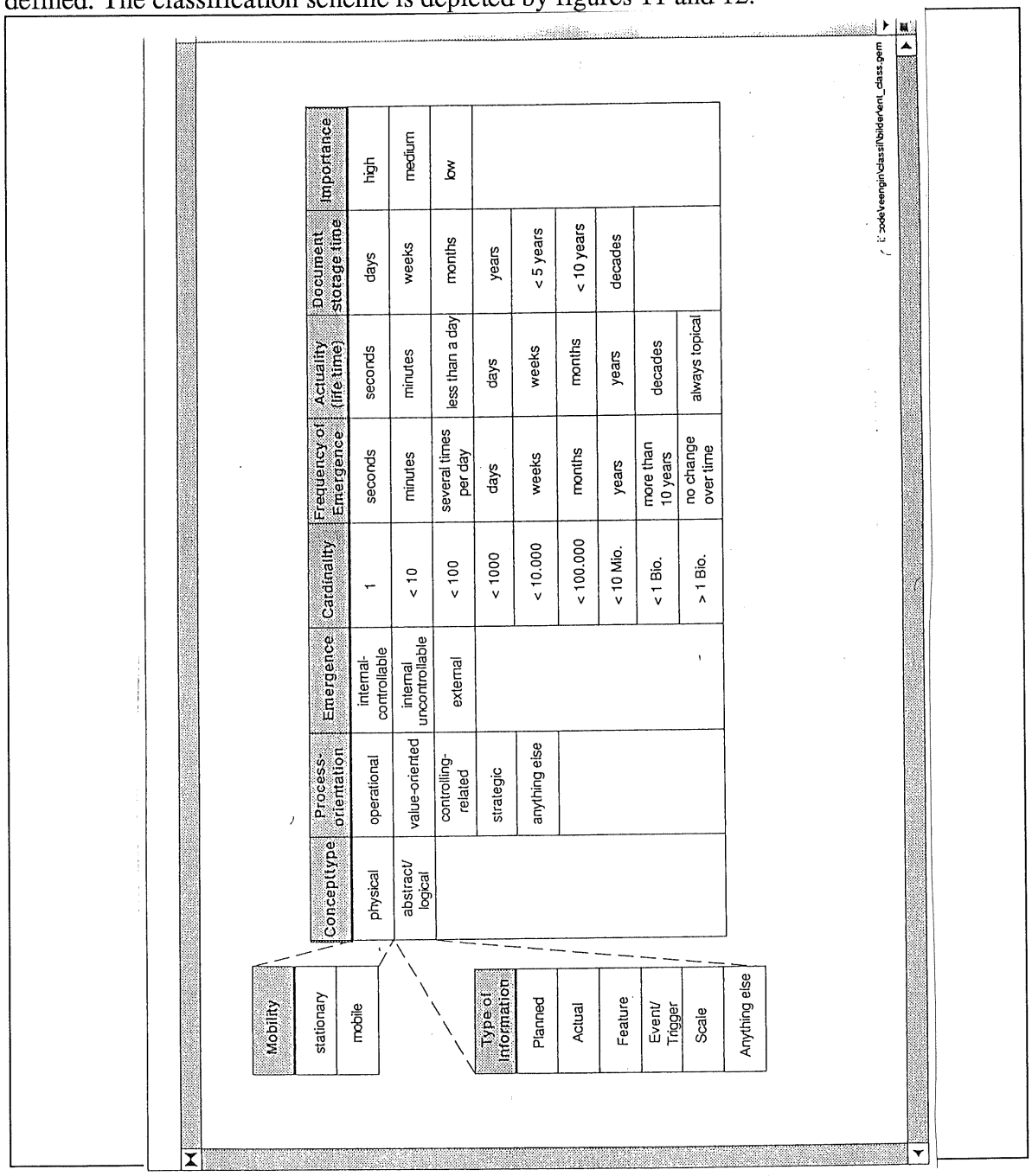

Fig. 11: Sample classification scheme for information objects (part1)

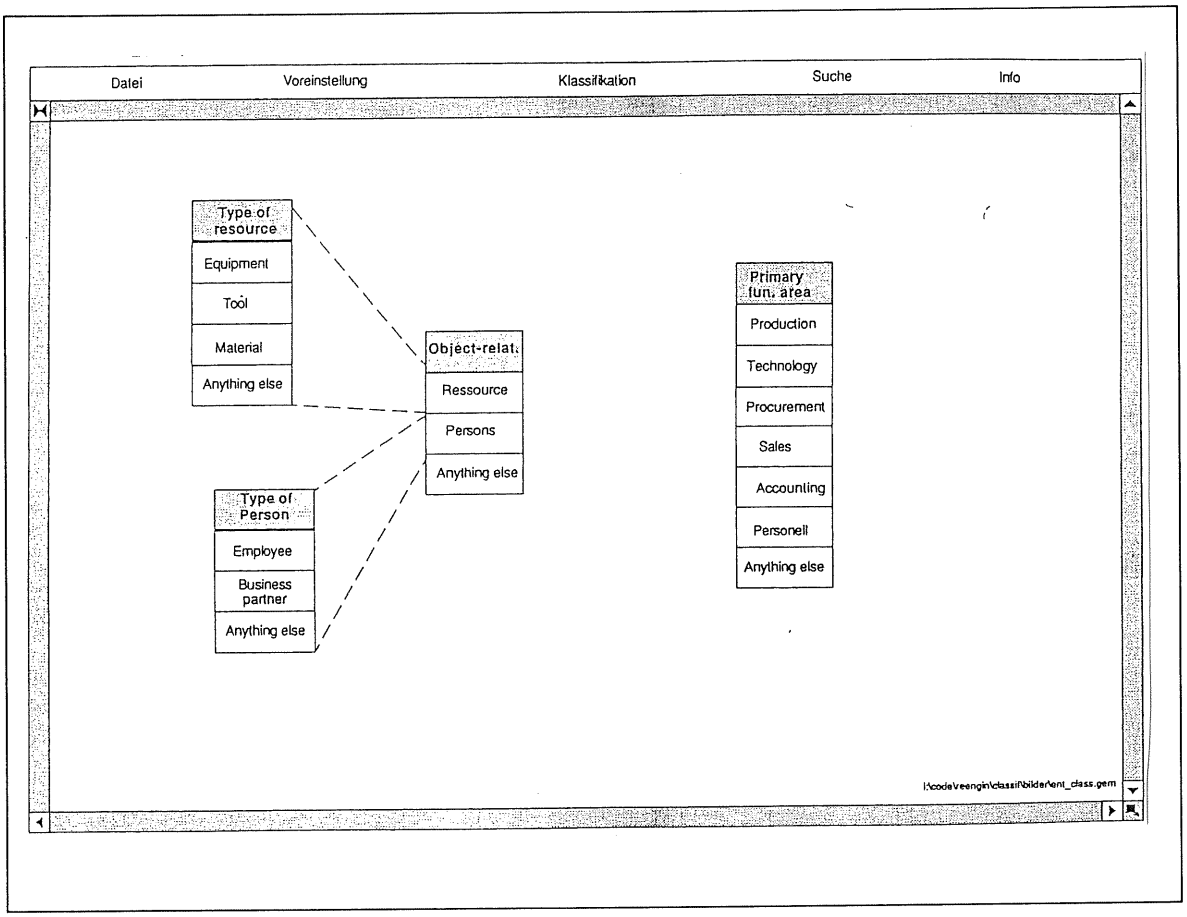

Fig. 12: Sample classification scheme for information objects (part 2)

7.1 Concept type:

The concept type is defined as the appearence of the enterprise object the information object refers to. The class "concept type" comprises the descriptors "physical" and "abstract/logical".

- ❑ Physical
  All enterprise objects which have a material existence are defined as physical. Physical objects can be additionally classified by the class "mobility" which decribes the fact that an object can and will be moved or is always located at the same place.
  - -- Stationary
    Stationary objects are always located at the same place i.e. the location of the object does not change.

-- Mobile

Mobile objects can be moved within the enterprise i.e. their location can change during they stay in the enterprise.

- Astract/Logical

  All enterprise objects which have no material existence are defined as abstract/logical. They can be additionally described by the class "type of information" which comprises the following descriptors:

  -- Planned

  Information objects which refer to future values of certain measuring indices.

  -- Actual

  Information objects which refer to realized values of certain measuring indices.

  -- Feature

  Information objects which describe characteristics of an enterprise object.

  -- Event/Trigger

  Information objects which refer to the appearance of a certain state of the information system are defined as event. An event can be a result or an input of a function.

  -- Scale

  Information objects which are used as a means to measure the value of certain variables are defined as scale.

  -- Anything else

7.2 Process orientation

The class "process orientation" incorporates the information objects into the different information levels of the enterprise.

- Operational

  Information objects which describe elements of the production process are defined as operational.

- Value-oriented

  Information objects which refer to the value of enterprise elements are defined as value-oriented. They basically belong to the accounting system of the enterprise.

- ❑ Controlling-related
  Information objects which refer to the analysis and planning of costs and benefits of enterprise elements are defined as controlling-related. They basically belong to the controlling system of the enterprise.
- ❑ Strategic
  Information objects which refer to the long-range planning of the enterprise are defined as strategic.
- ❑ Anything else

7.3 Emergence

The class "emergence" describes the origin of an information object. For internal objects i.e. information objects which emerge inside the enterprise it is stated whether they are controllable by the enterprise or not.

- ❑ Internal-controllable
  An information object which first emerges inside the enterprise and which can be controlled by the enterprise.
- ❑ Internal-uncontrollable
  An information object which first emerges inside the enterprise but can not be controlled by the enterprise.
- ❑ External
  An information object which emerges outside the enterprise✎

7.4 Cardinality

The cardinality describes the amount of instances of one information object within the enterprise. The descriptors are self-explanatory.

- ❑ 1
- ❑ < 10
- ❑ < 100
- ❑ < 1000
- ❑ < 10.000
- ❑ < 100.000
- ❑ < 10 Mio.
- ❑ < 1 Bio.
- ❑ > 1 Bio.

7.4 Frequency of Emergence

The frequence of emergence describes the average period of time between two succeeding instantiations of an information object. The descriptors are self-explanatory.

- ❑ seconds
- ❑ minutes
- ❑ several times per day
- ❑ days
- ❑ weeks
- ❑ months
- ❑ years
- ❑ more than 10 years
- ❑ no change over time

7.5 Actuality (life time)

The actuality determines the period of time during an information object is relevant (active) for the enterprise. The descriptors are self-explanatory.

- ❑ seconds
- ❑ minutes
- ❑ less than a day
- ❑ days
- ❑ weeks
- ❑ months
- ❑ years
- ❑ decades
- ❑ always topical

7.6 Document storage time

This class determines the period of time during the documentation of the information object has to be stored within the enterprise. The descriptors are self-explanatory.

- ❑ days
- ❑ weeks
- ❑ months
- ❑ years
- ❑ < 5 years
- ❑ < 10 years

- decades

7.7 Importance

The class "importance" describes the relevance of an information object for the enterprise.

- high
- medium
- low

7.8 Object-related

The class "object-related" describes the type of enterprise object an information object refers to.

- Resource

  The class "resource" comprises all productive factors of the enterprise except the human beeings.

  - Equipment

    The descriptor equipment comprises all machines and machine groups of the enterprise.

  - Tool

    Tools are used to support the usage of the enterpriseøs machines.

  - Material

    Materials are the necessary input for the production of the enterprise products.

  - Anything else

- Persons

  - employee

    An information object which refers to a member of the enterprise.

  - business partner

    An information object which refers to an external partner of the enterprise e.g. a customer or a supplier.

  - anything else

- Anything else

7.9 Primary functional area

This class describes the primary functional area the information object refers to. The descriptors are self-explanatory.

- ❑ Production
- ❑ Technology
- ❑ Procurement
- ❑ Sales
- ❑ Accounting
- ❑ Personell
- ❑ Anything else

References:

1 Buder, M.; Rehfeld, W.; Seeger, T. (Eds.): Grundlagen der praktischen Information und Dokumentation. Ein Handbuch zur Einführung in die fachliche Informationsarbeit. 3rd edition. Vol. 1. München 1990, p. 1.

2 Soergel, D.: Klassifikationssysteme und Thesauri. Eine Anleitung zur Herstellung von Klassifikationssystemen und Thesauri im Bereich der Dokumentation. Frankfurt/Main 1969, p. 27.

3 Soergel, D.: Indexing Languages and Thesauri: Construction and Maintenance. Los Angeles, California 1974, p. 27.

4 Atherton, P. (Ed.): Classification Research: Proceedings of the Second International Study Conference, Elsinore, 14. - 18. 09.1974. Copenhagen, Munksgaard 1965, p. 544.

5 Soergel, D.: Indexing Languages and Thesauri: Construction and Maintenance. Los Angeles, California 1974, p. 35.

6 Soergel, D.: Indexing Languages and Thesauri: Construction and Maintenance. Los Angeles, California 1974, p. 4.31.

7 The most important relationships between terms are standardized in DIN 1463 and ISO 2788. Examples are: BT (broader term), NT (narrower term), RT (related term), USE, UF (used for).

8 Gebhardt, F.: Dokumemtationssysteme. Berlin et al. 1981, p. 46.

9 Dewey, M: Decimal Classification and Relative Index. 19th edition. Albany, New York 1979.

10 Reusch, P.J.A.: Informationssysteme, Dokumentationssprachen, Data Dictionaries. Eine Einführung. Mannheim et al. 1980, p. 49.

11 Laisiepen, K.: Klassifikation. in: Laisiepen, K.; Lutterbeck, E; Meyer-Uhlenried, (Eds.): Grundlagen der praktischen Information und Kommunikation,. Eine Einführung. 2nd edition. München et al. 1980, pp. 299-350, p. 312.

12 Laisiepen, K.: Klassifikation. in: Laisiepen, K.; Lutterbeck, E; Meyer-Uhlenried, (Eds.): Grundlagen der praktischen Information und Kommunikation,. Eine Einführung. 2nd edition. München et al. 1980, pp. 299-350, p. 312-326.

13 Vickery, B.C.: Faceted Classification. A Guide to Construction and Use of a Special Kind of Schemes. London 1968. Vickery, B.C.: Faceted Classification Schemes. New Brunswick 1966.

14 Ranganathan, S.R.: Prolegomena to Library Classification. Bombay, India 1967.

15 see Hars, A.; Heib, R.; Kruse, Chr.; Michely, J.; Scheer, A.-W.: Concepts of current data modelling methodologies - A survey -. In: Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 84, Saarbrücken 1991.

16 see the section on configuration of the descriptor tables.

17 Mresse M.: Information Retrieval - Eine Einführung. Stuttgart 1984.

Die Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi) im Institut für empirische Wirtschaftsforschung an der Universität des Saarlandes erscheinen in unregelmäßiger Folge.

* Die Hefte 1 - 31 werden nicht mehr verlegt.

Heft 32: A.-W. Scheer: Einfluß neuer Informationstechnologien auf Methoden und Konzepte der Unternehmensplanung, März 1982, Vortrag anläßlich des Anwendergespräches "Unternehmensplanung und Steuerung in den 80er Jahren in Hamburg vom 24. - 25.11.1981

Heft 33: A.-W. Scheer: Disposition- und Bestellwesen als Baustein zu integrierten Warenwirtschaftssystemen, März 1982, Vortrag anläßlich des gdi-Seminars "Integrierte Warenwirtschafts-Systeme" in Zürich vom 10. - 12. Dezember 1981

Heft 34: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS - Ein Ansatz zur Entwicklung prüfungsgerechter Software-Systeme, Mai 1982

Heft 35: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Konzept einer computergestützten Prüfungsumgebung, Juli 1982

Heft 36: A.-W. Scheer: Rationalisierungserfolge durch Einsatz der EDV - Ziel und Wirklichkeit, August 1982, Vortrag anläßlich der 3. Saarbrücker Arbeitstagung "Rationalisierung" in Saarbrücken vom 04. - 06. 10.1982

Heft 37: A.-W. Scheer: DV-gestützte Planungs- und Informationssysteme im Produktionsbereich, September 1982

Heft 38: A.-W. Scheer: Interaktive Methodenbanken: Benutzerfreundliche Datenanalyse in der Marktforschung, Mai 1983

Heft 39: A.-W. Scheer: Personal Computing - EDV-Einsatz in Fachabteilungen, Juni 1983

Heft 40: A.-W. Scheer: Strategische Entscheidungen bei der Gestaltung EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom 26. - 28.09.1983

Heft 41: H. Krcmar: Schnittstellenprobleme EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom 26. - 28.09.1983

Heft 42: A.-W. Scheer: Factory of the Future, Vorträge im Fachausschuß "Informatik in Produktion und Materialwirtschaft" der Gesellschaft für Informatik e. V., Dezember 1983

Heft 43: A.-W. Scheer: Einführungsstrategie für ein betriebliches Personal-Computer-Konzept, März 1984

Heft 44: A.-W. Scheer: Schnittstellen zwischen betriebswirtschaftlicher und technische Datenverarbeitung in der Fabrik der Zukunft, Juli 1984

Heft 45: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Ein Werkzeug zur Messung der Qualität von Software-Systemen, August 1984

Heft 46: H. Krcmar: Die Gestaltung von Computer am-Arbeitsplatz-Systemen - ablauforientierte Planung durch Simulation, August 1984

Heft 47: A.-W. Scheer: Integration des Personal Computers in EDV-Systeme zur Kostenrechnung, August 1984

Die Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi) im Institut für empirische Wirtschaftsforschung an der Universität des Saarlandes erscheinen in unregelmäßiger Folge.

* Die Hefte 1 - 31 werden nicht mehr verlegt.

Heft 32: A.-W. Scheer: Einfluß neuer Informationstechnologien auf Methoden und Konzepte der Unternehmensplanung, März 1982, Vortrag anläßlich des Anwendergespräches "Unternehmensplanung und Steuerung in den 80er Jahren in Hamburg vom 24. - 25.11.1981

Heft 33: A.-W. Scheer: Dispositon- und Bestellwesen als Baustein zu integrierten Warenwirtschaftssystemen, März 1982, Vortrag anläßlich des gdi-Seminars "Integrierte Warenwirtschafts-Systeme" in Zürich vom 10. - 12. Dezember 1981

Heft 34: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS - Ein Ansatz zur Entwicklung prüfungsgerechter Software-Systeme, Mai 1982

Heft 35: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Konzept einer computergestützten Prüfungsumgebung, Juli 1982

Heft 36: A.-W. Scheer: Rationalisierungserfolge durch Einsatz der EDV - Ziel und Wirklichkeit, August 1982, Vortrag anläßlich der 3. Saarbrücker Arbeitstagung "Rationalisierung" in Saarbrücken vom 04. - 06. 10.1982

Heft 37: A.-W. Scheer: DV-gestützte Planungs- und Informationssysteme im Produktionsbereich, September 1982

Heft 38: A.-W. Scheer: Interaktive Methodenbanken: Benutzerfreundliche Datenanalyse in der Marktforschung, Mai 1983

Heft 39: A.-W. Scheer: Personal Computing - EDV-Einsatz in Fachabteilungen, Juni 1983

Heft 40: A.-W. Scheer: Strategische Entscheidungen bei der Gestaltung EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom 26. - 28.09.1983

Heft 41: H. Krcmar: Schnittstellenprobleme EDV-gestützter Systeme des Rechnungswesens, August 1983, Vortrag anläßlich der 4. Saarbrücker Arbeitstagung "Rechnungswesen und EDV" in Saarbrücken vom 26. - 28.09.1983

Heft 42: A.-W. Scheer: Factory of the Future, Vorträge im Fachausschuß "Informatik in Produktion und Materialwirtschaft" der Gesellschaft für Informatik e. V., Dezember 1983

Heft 43: A.-W. Scheer: Einführungsstrategie für ein betriebliches Personal-Computer-Konzept, März 1984

Heft 44: A.-W. Scheer: Schnittstellen zwischen betriebswirtschaftlicher und technische Datenverarbeitung in der Fabrik der Zukunft, Juli 1984

Heft 45: J. Ahlers, W. Emmerich, H. Krcmar, A. Pocsay, A.-W. Scheer, D. Siebert: EPSOS-D, Ein Werkzeug zur Messung der Qualität von Software-Systemen, August 1984

Heft 46: H. Krcmar: Die Gestaltung von Computer am-Arbeitsplatz-Systemen - ablauforientierte Planung durch Simulation, August 1984

Heft 47: A.-W. Scheer: Integration des Personal Computers in EDV-Systeme zur Kosten-

rechnung, August 1984

Heft 48: A.-W. Scheer: Kriterien für die Aufgabenverteilung in Mikro-Mainframe Anwendungssystemen, April 1985

Heft 49: A.-W. Scheer: Wirtschaftlichkeitsfaktoren EDV-orientierter betriebswirtschaftlicher Problemlösungen, Juni 1985

Heft 50: A.-W. Scheer: Konstruktionsbegleitende Kalkulation in CIM-Systemen, August 1985

Heft 51: A.-W. Scheer: Strategie zur Entwicklung eines CIM-Konzeptes - Organisatorische Entscheidungen bei der CIM-Implementierung, Mai 1986

Heft 52: P. Loos, T. Ruffing: Verteilte Produktionsplanung und -steuerung unter Einsatz von Mikrocomputern, Juni 1986

Heft 53: A.-W. Scheer: Neue Architektur für EDV-Systeme zur Produktionsplanung und -steuerung, Juli 1986

Heft 54: U. Leismann, E. Sick: Konzeption eines Bildschirmtext-gestützten Warenwirtschaftssystems zur Kommunikation in verzweigten Handelsunternehmungen, August 1986

Heft 55: D. Steinmann: Expertensysteme (ES) in der Produktionsplanung und -steuerung (PPS) unter CIM-Aspekten, November 1987, Vortrag anläßlich der Fachtagung "Expertensysteme in der Produktion" am 16. und 17.11.1987 in München

Heft 56: A.-W. Scheer: Enterprise wide Data Model (EDM) as a Basis for Integrated Information Systems, Juli 1988

Heft 57: A.-W. Scheer: Present Trends of the CIM Implementation (A qualitative Survey) Juli 1988

Heft 58: A.-W. Scheer: CIM in den USA - Stand der Forschung, Entwicklung und Anwendung, November 1988

Heft 59: R. Herterich, M. Zell: Interaktive Fertigungssteuerung teilautonomer Bereiche, November 1988

Heft 60: A.-W. Scheer, W. Kraemer: Konzeption und Realisierung eines Expertenunterstützungssystems im Controlling, Januar 1989

Heft 61: A.-W. Scheer, G. Keller, R. Bartels: Organisatorische Konsequenzen des Einsatzes von Computer Aided Design (CAD) im Rahmen von CIM, Januar 1989

Heft 62: M. Zell, A.-W. Scheer: Simulation als Entscheidungsunterstützungsinstrument in CIM, September 1989

Heft 63: A.-W. Scheer: Unternehmens-Datenbanken - Der Weg zu bereichsübergreifenden Datenstrukturen, September 1989

Heft 64: C. Berkau, W. Kraemer, A.-W. Scheer: Strategische CIM-Konzeption durch Eigenentwicklung von CIM-Modulen und Einsatz von Standardsoftware, Dezember 1989

Heft 65: A. Hars, A.-W. Scheer: Entwicklungsstand von Leitständen[1], Dezember 1989

Heft 66: W. Jost, G. Keller, A.-W. Scheer: CIMAN - Konzeption eines DV-Tools zur

Gestaltung einer CIM-orientierten Unternehmensarchitektur, März 1990

Heft 67: A.-W. Scheer: Modellierung betriebswirtschaftlicher Informationssysteme (Teil 1: Logisches Informationsmodell), März 1990

Heft 68: W. Kraemer: Einsatzmöglichkeiten von Expertensystemen in betriebswirtschaftlichen Anwendungsgebieten, März 1990

Heft 69: A.-W. Scheer, R. Bartels, G. Keller: Konzeption zur personalorientierten CIM-Einführung, April 1990

Heft 70: St. Spang, K. Ibach: Zum Entwicklungsstand von Marketing-Informationssystemen in der Bundesrepublik Deutschland, September 1990

Heft 71: D. Aue, M. Baresch, G. Keller: **URMEL**, Ein **U**nte**R**nehmens**Mod**e**L**lierungsansatz, Oktober 1990

Heft 72: M. Zell: Datenmanagement simulationsgestützter Entscheidungsprozesse am Beispiel der Fertigungssteuerung, November 1990

Heft 73: A.-W. Scheer, M. Bock, R. Bock: Expertensystem zur konstruktionsbegleitenden Kalkulation, November 1990

Heft 74: R. Bartels, A.-W. Scheer: Ein Gruppenkonzept zur CIM-Einführung, Januar 1991

Heft 75: M. Nüttgens, St. Eichacker, A.-W. Scheer: CIM-Qualifizierungskonzept für Klein- und Mittelunternehmen (KMU), Januar 1991

Heft 76: Ch. Houy, J. Klein: Die Vernetzungsstrategie des Instituts für Wirtschaftsinformatik - Migration vom PC-Netzwerk zum Wide Area Network (noch nicht veröffentlicht)

Heft 77: W. Kraemer: Ausgewählte Aspekte zum Stand der EDV-Unterstützung für das Kostenmanagement: Modellierung benutzerindividueller Auswertungssichten in einem wissensbasierten Controlling-Leitstand, Mai 1991

Heft 78: H. Heß: Vergleich von Methoden zum objektorientierten Design von Softwaresystemen, August 1991

Heft 79: A.-W. Scheer: Konsequenzen für die Betriebswirtschaftslehre aus der Entwicklung der Informations- und Kommunikationstechnologien, Mai 1991

Heft 80: G. Keller, J. Kirsch, M. Nüttgens, A.-W. Scheer: Informationsmodellierung in der Fertigungssteuerung, August 1991

Heft 81 A.-W. Scheer: Papierlose Beratung - Werkzeugunterstützung bei der DV-Beratung, August 1991

Heft 82: C. Berkau: VOKAL (System zur Vorgangskettendarstellung und -analyse) - Struktur der Modellierungsmethode - Juni 1991 (wird nicht verlegt)

Heft 83: A. Hars, R. Heib, Ch. Kruse, J. Michely, A.-W. Scheer: Concepts of Current Data Modelling Methodologies - Theoretical Foundations - 1991

Heft 84: A. Hars, R. Heib, Ch. Kruse, J. Michely, A.-W. Scheer: Concepts of Current Data Modelling Methodologies - A Survey - 1991

Heft 85: W. Hoffmann, M. Nüttgens, A.-W. Scheer, St. Scholz: Das Integrationskonzept am CIM-TTZ Saarbrücken (Teil 1: Produktionsplanung), Oktober 1991

Heft 86: A.-W. Scheer: Koordinierte Planungsinseln: Ein neuer Lösungsansatz für die Produktionsplanung, November 1991

Heft 87: M. Nüttgens, G. Keller, A.-W. Scheer, S. Stehle: Konzeption hyperbasierter Informationssysteme, Dezember 1991

Heft 88: W. Hoffmann, B. Maldener, M. Nüttgens, A.-W. Scheer: Das Integrationskonzept am CIM-TTZ Saarbrücken (Teil 2: Produktionssteuerung), Januar 1992

Heft 89: G. Keller, M. Nüttgens, A.-W. Scheer: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßkosten (EPK)", Januar 1992

Heft 90: C. Berkau, A.-W. Scheer: VOKAL (System zur Vorgangskettendarstellung und -Analyse), Teil 2: VKD-Modellierung mit VOKAL

Heft 91: C. Berkau: Konzept eines controllingbasierten Prozeßmanagers als intelligentes Multi-Agent-System, Januar 1992

Heft 92: A. Hars, R. Heib, Chr. Kruse, J. Michely, A.-W. Scheer: Approach to Classification for Information Engineering - Methodology and Tool Specification, August 1992

Heft 93: M. Nüttgens, A.-W. Scheer, M. Schwab: Integrierte Entsorgungssicherung als Bestandteil des betrieblichen Informationsmanagements, August 1992

Heft 94: Chr. Kruse, A.-W. Scheer: Modellierung und Analyse dynamischen Systemverhaltens, Oktober 1992