MASTER IN COMPUTER SCIENCE

Conception, spécification and prototyping of a contract writing support system

Hoorens, Jan

Award date:
1983

Awarding institution:
University of Namur

Link to publication

INSTITUT D'INFORMATIQUE - F.N.D.P. NAMUR

CONCEPTION SPECIFICATION AND PROTOTYPING OF A CONTRACT WRITING SUPPORT SYSTEM

JAN HOORENS

DISSERTATION FOR THE OBTAINMENT OF THE DEGREE OF

"LICENCIE ET MAITRE EN INFORMATIQUE"

ACADEMIC YEAR 1982-1983.

PROMOTER

PROF. PH. VAN BASTELAER

INTRODUCTION

This paper presents a support system for contract writers. The paper is subdivided in two parts. The first part is intitled ' Towards a solution ' and relates the problem to the proposed solution. The second part, intitled ' Towards a realisation ', relates the proposed solution to EDP technology.

Part 1 starts by describing the problem of contract writing, analysing and comparing the existing solution types. Then we explore two research directions that lead to discovery of requirements for a better solution type. The first research direction confronted a general analysis schema for decision support systems with the problem of contract writing. This made us sensitive to the structure of the task involved in contract writing, to the inherent dynamics of a decision support system, to the concepts and properties of the cognitive process of the contract writer and to contextual issues. The second research direction aimed at integrating existing solution types in a more general solution type that would be able to overcome the major drawbacks of existing solution types. The requirements being established we introduce gradually the proposed solution and the basic reasonings behind the proposed solution. These reasonings concern the different subparts of the contract writer's knowledge : contractual modelling, data base of contract clauses and the link between facts and clauses . Contractual modelling offers an interesting technique to represent a first subpart of the contract writer's knowledge : what aspects of the relationship of contractors a contract can regulate and how each aspect can be organised. It also permits to represent a contract's semantics in a stable form of symbolisation. This representation of semantics permits to compare contracts and to propose reconciling clauses when contractors have divergent interests. The data base of contract clauses represents a second subpart of the contract writer's knowledge and is organised as to permit the system to memorise the semantics of each clause by the same stable form of symbolisation. The link between the facts caracterising the situations of candidate contractors and the clauses of an adapted contract for those situations represents a third subpart of the contract writer's knowledge. We incorporated this knowledge in the system through a questionnaire, testing for the facts caracterising the situation of a particular candidate contractor, and by providing a mecanism permitting to deduce from the answered questionnaire a contract that is adapted to the particular situation . After having exposed and justified the basic reasonings behind the solution, we refined the system by introducing a coherence subsystem, a authorisation subsystem and a security subsystem. The coherence subsystem verifies before each execution of a function if the system is in an acceptable state or not to execute the function. The authorization subsystem protects the system against unauthorized access and use. The security subsystem guards the system against incidents that may to varying degrees destroy the system contents . We end the first part with a discussion of the possible uses of the proposed system :information retrieval, contract writing and negociation, draft of a call for tender, educative support.

Part 2 comments the successive phases of realisation : functional analysis, design and prototyping. We first discuss the functional analysis method and its representation : conceptual schema, data dictionnary, function dictionnary. Next we expose design methodology and representation : high level module specification, software architecture map, data base manipulating routines specification,schema of possible accesses and schema of required accesses. The retained design principles are defined and justified. We end the second part by discussing why and how we realised the proposed system in a prototype form.

We conclude the paper by indicating possible areas for additional research and development.

ACKNOWLEDGEMENTS

THE AUTHOR GRATEFULLY ACKNOWLEDGES THE SUPPORT AND ASSISTANCE OF

- PROF. PH. VAN BASTELAER, WHO PROVIDED STRONG PERSONAL SUPPORT AND STIMULATING SUPERVISION,
- PROF. F. BODART, WHO INTRODUCED ME TO THE DISCIPLINE OF DECISION SUPPORT SYSTEMS,
- PROF. C. CHERTON, WHOSE COMMENTS ON PROTOTYPING WERE MOST USEFUL,
- PROF. P. MAON, WHO CREATED THE REQUIRED ENVIRONMENT TO INVESTIGATE THE LINKS BETWEEN COMPUTER SCIENCE AND LAW,
- J. ROULIN, FOR THE MANY FRUITFUL DISCUSSIONS,
- MY WIFE CECILE, WHO SACRIFIED HOLIDAYS AND WEEK ENDS IN COMPANY OF EMACS AND CONSTANTLY ENCOURAGED ME DURING THIS SIX MONTHS PROJECT.

PART ONE : TOWARDS A SOLUTION

CHAPTER 1

THE PROBLEM OF CONTRACT WRITING

In this chapter the scope and the nature of the problem of contract writing are described (1.1.). Contract writing is an old problem and law practitionners have discovered a variety of ways to deal with it. After an overview of the existing solutions (1.2.), a comparison of the described solutions establishes the main differences relative to a set of selected and relevant comparison criteria (1.3.) The comparison is useful as it highlights the major drawbacks of the existing solutions and is thought generating for a new solution, in the sense that it helps to state what desirable qualities a better solution should have and what undesirable characteristics it should try to avoid(1.4.)

1.1. DESCRIPTION OF THE PROBLEM OF CONTRACT WRITING

The problem of contract writing cannot be handled independently of the law environment in which it is posed. Therefore a short discussion of the main concepts of the surrounding law environment is useful. We will introduce the concepts of contract, the principle of contractual freedom, enforcement , default law rule, qualification rule, imperative law rule, regulated contract, jurisprudential doctrine, contract life cycle, the interpretation of contracts, the negociation of contracts, the economics of contract writing (1.1.1.). Once the law environmnent is pictured, its implications on the activity of contract writing will be analysed (1.1.2.). We will conclude with a formulation of the problem of contract writing (1.1.3.).

1.1.1. The surrounding law environment.

A contract is an agreement between two or more subjects of law stating the rights and duties of each. The word contract designates also the document in which the rights and duties constituting the agreement are written down. In most judicial systems the enforcability of a contract is closely linked to the possibility of proving the existence and the contents of the agreement and for proving the existence and the contents of the agreement a written form is preferred. This explains why the same word contract is used both to refer to the agreement and to its written proof.

Defining a contract as an agreement between two or more subjects of law introduces the principle of contractual freedom. This principle is based on the belief that all men are equal and free and constitutes the judicial counterpart of liberal economic doctrine. The principle of contractual freedom essentialy means that:

- subjects of law are free to make agreements
- subjects of law freely decide upon the type of agreement they make
- subjects of law discuss freely the rights and duties constituting the agreement

In judicial systems inspired by the principle of contractual freedom, the State plays a complementary role which is not negligible for the problem of contract writing. First, the State must be able to enforce the respect of agreement made between people. Where the conclusion and loyal execution of a contract is basically a private matter,the enforcement of the agreement is a public matter. This implies that courts will examine the contract, interpret according to some rules the intent of the contractors, appreciate the evidence invoked and make a decision. Secondly, the State provides the rights and duties which are applicable by default to each contract type. This means that when two people make a certain type of agreement and the contract is silent about some matter, then that matter is considered to be governed by the rights and duties applicable by default. These rights and duties are created by default law rules. Thirdly the set of applicable default rules or imperative law rules is function of the type of agreement concluded. Therefore Courts apply qualifications rules in order to determine of which type of agreement the particular contract is an occurrence.

Since the end of the nineteenth century, the principle of contractual freedom has become under constant attack. The basic criticism was directed to the underlying hypothesis of equality between men : all men are not equal, there exist men who have and men who don't have; there are the men who know and those who don't know. In the framework of contractual freedom the inequalities tend to become stronger and the weakest to be left without protection. As a result of the socialising movement, the legislator softened the principle of contractual freedom by enacting more and more exceptions to the principle. The role of the state evolved from a passive supervsion to active intervention wether directive wether protective. This evolution is not irrelevant for the problem of contract writing in at least 3 ways :

- the introduction of imperative law rules
- the recourse to regulated contracts
- the development of jurisprudential doctrine

The imperative law rule is a law rule that is applicable to a contract whatever contractors have agreed themselves. The imperative law rule qualifies as an exception to the principle of contractual freedom because it limits the freedom of contractors to decide what rights and duties are part of the contract.

The regulated contract is a contract in which the rights and duties are completely determined by the legislator. The regulated contract qualifies as an exception to the principle of contractual freedom in that contractors only have the freedom to contract or not to contract. If they contract, the terms of the contract will be those of the regulated contract.

The concept of jurisprudential doctrine refers to the duty of Courts to apply abstract law rules to particular cases. The whole body of court decisions - being controlled by a Super Court - tends to precise the abstract law rules, to systematise the interpretation. The set of prevailing interpretations constitutes the jurisprudential doctrine at a given moment of time.

To complement the rather static definition of a contract given above, let's have a view at the dynamics of a contract and examine the contract's life cycle.

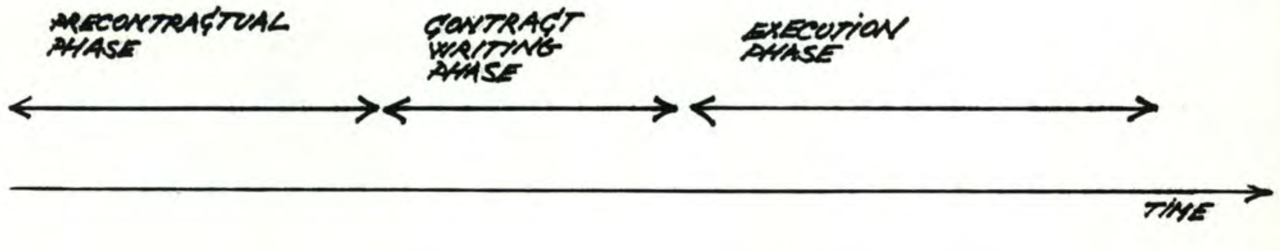


Figure 1-1: THE CONTRACT'S LIFE CYCLE

The precontractual phase is the phase in which candidate contractors enter into contact, discuss a possible deal and try to agree upon mutual rights and duties. The precontractual phase ends when an agreement is reached.

The contract writing phase is the phase during which a document containing the description of mutual rights and duties is written down. This phase ends with the signing of the contract.

The execution phase starts at the date of execution planned by the parties or decided by default law rules and ends when all rights and duties are legally extinguished.

The execution phase can take place according to different patterns. The first pattern is the normal execution pattern. The normal execution pattern is present when contractors execute faithfully the contract, behave according to the rights and duties governing the contract. The contract doesn't give rise to any dispute or difficulty of interpretation.

The problem execution pattern is present when contractors execution of the contract reveals a problem of interpretation of the rights and duties of each party. or when one or both of the contractors don't partially or totally behave according to the rights and duties the other party expected to be present under the contract. The problem is settled by mutual agreement and after settlement parties find an acceptable modus vivendi.

The court execution pattern is present when the same problem is settled by having recourse to a court or eventually to arbitration.

Several remarks should refine the raw schema as represented on FIGURE 1-1.

1. The duration and the contents of each phase may vary greatly upon the type of contract considered. A whole spectrum is possible going from contracting for a packet of cigarettes to the turnkey delivery of a car factory taking several man-years of precontractual phase, four months of contract writing and a three year execution period.

2. The succession of phases is not necessarily serial. Phases may overlap to some extent. In the precontractual phase a call for tender may communicate the requirement of the caller to insert into the eventual contract a set of specified clauses. As another example the

contract may contain or refer to documents of the precontractual phase. Also during the execution phase contractors may decide to explicit or modify some clauses by the use of addenda.

3. The duration and contents of the precontractual phase is influenced by the negociation power of candidate contractors and may vary from a 'take it or leave it' to an intensively discussed deal.

We have already mentionned the interpretation of contracts. Interpretation is the activity of determining what rights and duties parties have under a contract. Interpretation will be done by the parties themselves during the contract writing phase and during the execution phase when it takes the problem or court execution pattern. More important is that courts will generaly make the definitive interpretation and have the last word about parties' rights and duties. The Law furnishes some rules concerning the way courts may interprete contracts. As a corrolary of the principle of contractual freedom the basic rule of interpretation is that the judge should find out the common intention of the contractual parties and interpret the contract according to that common intention. A second rule is that interpretation only happens when there is doubt : "interpretatio cessat in claris". Two complementary rules state that in case of doubt, the interpretation should be in favour of the debtor and that interpretation should be done by preference in the sense in which the contract remains valid.

Contrat writing in a given law environment can be analysed from an economic point of view. Contract writing can be viewed as an activity which on the one hand consumes ressources and on the other hand creates utility. The ressources consumed are mainly intellectual ones : the writers experience of the problem field, his knowledge of default law rules, imperative law rules and his imagination for constructing adequate clauses, his time to study a particular case and to apply his knowledge to a particular case. The utility created resides in the economic value of the solution of the problem of contract writing

1.1.2. Implications of the law environment on the activity of contract writing.

Following statements summarize the major implications of the law environment on the activity of contract writing

1. Contract writing is a necessary activity for proving the existence

and the contents of an agreement.

2. Given the basic principle of contractual freedom, contract writing is a very creative activity leaving room for a spectrum of clauses regulating the relationship of contractors.

3. Given the existence of default law rules and the triggering qualification rules, the contract writer must know the qualification rules in order to know what default law rules may be applicable and must know the default law rules to decide if he must write or not an overruling conventional clause discarding the default law rule.

4. Given the existence of imperative law rules, regulated contracts and their triggering qualification rules, the contract writer must know the qualification rules in order to know what imperative law rules or regulated contracts may be applicable and must know the imperative rules or regulated contracts to decide to what degree overruling conventional clauses may be written effectively.

5. Given the role assumed by a State's courts in the enforcement of contracts and the force of jurisprudential doctrine, the contract writer should have knowledge of the prevailing jurisprudential doctrine in order to anticipate how a Court will enforce a contract clause.

6. Given the basic rule of interpretation the contract writing should be the activity of perfect specification of the common intent of contractors. This implies absence of specification sins such as silence (except when default law rules reflect the common intent), noise, contradiction, redundance.

7. Given the fact that the execution phase of a contract's life cycle may evolve from the normal execution pattern to the problem or court execution pattern, and provided two last execution patterns are undesirable for any reason, the contract writer should enhance the probability of the execution phase taking its normal execution pattern.

8. Given the principle of contractual freedom and the presence of contractors with initially opposed interests who converge or not during the precontractual phase to an agreement, the contract writer has to determine the initially opposed interests and to propose reconciling clauses helping the agreement to be reached.

1.1.3. Formulation of the problem of contract writing.

Given the law environment as pictured in 1.1.1., and its implications as summarised in 1.1.2., we try to capture the problem of contract writing by providing following definition.

Contrat writing is the economic activity based on the knowledge of a given law environment and of an economic operation to be carried out. The activity of contract writing comprises as subactivities :

1. identification of the desiderata of each party,
2. identification of the common intent of both parties by negociation of the initially opposed desiderata
3. translation of the common intent into a written document such that the legal effects or facts obtainable under the contract are as close as possible to the common intent of parties.

1.2. OVERVIEW OF EXISTING SOLUTIONS TO THE PROBLEM OF CONTRACT WRITING

Law practitioners and theoreticians have imagined and put into practice different solutions to the problem of contract writing.

Our investigation on this field, enabled us to discover five major existing solution types:

- the model contract solution type
- the standard contract solution type
- the doctrinal treaty solution type
- the compilation of clauses solution type
- the by the example solution type

In this section each major solution type is defined, occurrences of the solution type are indicated and a description of its associated work method is given. It should be mentionned that the major solution types are not necessarily disjoint and can be combined in the contract writing for a particular market.

1.2.1. The model contract solution type

Definition

The model contract solution type is present when a contract writer or group of contract writers designs a contract adapted to the particular interests of a large group of possible contractors. The model contract solution type is based on the hypothesis that the group of possible contractors is sufficiently homogeneous (one model contract for all possible contractors) and on the hypothesis that the model contract is sufficiently close to the common intent of contractors.

Occurrences of the model contract solution type

1. the CECUA contract [3] for the acquisition of EDP-systems, defending the interests of small and medium entreprises.
2. the TESTS-ACHATS contracts [7] for the conclusion of most common contracts, principally defending the interests of consumers.

Associated work method

Consumers constitute a defending organisation. The defending organisation charges a group of contract writers to design a model contract. The model contract is distributed by the defending organisation among its members. The members of the defending organisation propose the model contract to their cocontractors.

1.2.2. The standard contract solution type

Definition

The standard contract solution type is present when one subject of law charges a contract writer or group of contract writer to design a contract adapted to his particular interests. The standard contract solution type is based on the hypotheses that

- it's economically unjustifiable to discuss individually the applicable contract,
- the negociation strentgh of the subject of law proposing the standard contract is sufficiently strong in order not to discuss with each individual cocontractor the terms of the agreement.

Occurrences of the standard contract solution type

1. I.B.M.'s contracts for EDP-goods and services
2. General agreements concerning telephone services, railway transportation or database consultation.

Associated work method

A subject of law having a predominating position on the market, charges a group of contract writers to design a standard contract. The standard contract is principally proposed to every cocontractor. If eventually the hypothesis justifying the standard contract are not realised, an individualised discussion of contract terms might take place.

1.2.3. The doctrinal treaty solution type

Definition

The doctrinal treaty solution type is present when a group of contract writers describe the state of the law environment applicable to a set of contract types independently of the particular interests of a particular subject of law or of a group of possible contractors. By description of the state of the law environment applicable to a set of contracts is understood a precise description and evaluation of the main law concepts (as introduced in 1.1.) as applicable to the set of contracts the doctrinal treaty deals with.

Occurrences of the doctrinal treaty solution type

1. Précis n. 4 Faculté de Droit. Le droit des contrats informatiques. Principes - Applications [5].This treaty deals with EDP-contracts.
2. Le Traité des Bâtisseurs de Delvaux [6] concerning contracts for construction of real estates

Associated work method

An organisation (university or gouvernment agency), relatively independent from the interests of specific economic actors on a market, charges a group of contract writers to draft a doctrinal treaty. The doctrinal treaty serves to the economic actors as a framework for constructing or adapting the contracts they practise or will practise.

1.2.4. The compilation of clauses solution type

Definition

The compilation of clauses solution type is present when a group of contract writers compiles most or all possible clauses for a given contract type. The compiled clauses cover all or most interests possibly present on the market associated to the contract type. As in the doctrinal treaty solution type, the group of contract writers is principally independent of the particular interests of specific subjects of law. However as opposed to the doctrinal solution type, the compilation of clauses solution type insists more on the formulation of clauses than on a description of the state of the law environment.

Occurrences of the compilation of clauses solution type

1. BRANDON & SEGELSTEIN, DATA PROCESSING EDP CONTRACTS [2] exposing in part 2 a cookbook of contractual clauses for EDP-contracts
2. THIRAN Compilation des clauses uselles des contrats informatiques [17].

Associated work method

A contract writer or group of contract writers recognises the interest of the market for a less or more exhaustive knowledge of possible contract clauses and constitutes a compilation of clauses. The compilation of clauses serves to the economic actors as a framework for constructing and adapting the contracts they practise or will practise.

1.2.5. The by the example solution type.

Definition

The by the example solution type is present when a group of contract writers designs a contract adapted to the most probable reconciliation point of contractors. The by the example solution type is based on the hypothesis that there exists types of contracts where either the interests of cocontractors are not strongly opposed or the cocontractors don't want to enter into a discussion of the contract terms.

Occurrences of the by the example solution type

1. CORNELIS & STAESENS . Formulaire d'actes notariés [4].
2. Recueil de modèles d'actes et de contrats professionnels et privés [15].

Associated work method

A contract writer or group of contract writers recognises the interest of the market for contracts adapted to the most probable reconciliation point of contractors. A diffusion of such contracts is organized. Contractors take the contract and sign it.

1.3. COMPARISON OF EXISTING SOLUTIONS TO THE PROBLEM OF CONTRACT WRITING

Given the description of the major existing solution types we will try to compare these solution types. The objective of the comparison is to evaluate each solution type in regard of another solution type and to be able to formulate some conclusions about the state of art in the field of contract writing. The objective of the comparison being fixed, the criteria used for comparison will be formulated in 1.3.1., a comparison is represented in 1.3.2. and the conclusions are summarised in 1.3.3.

1.3.1. Definition of the comparison criteria.

Any comparison criterion of different solutions to a same problem is to some degree arbitrary : it reflects the importance the evaluator attributes to a particular aspect of a solution or of the problem. Nothwithstanding this nuancing remark, lets try to formulate some criteria which might by useful to the evaluation of existing solution types to the problem of contract writing. Each comparison criterion will be defined and justified.

Criterion 1:	QUICKNESS OF OBTAINMENT OF A CONTRACT
definition:	what is the time between reaching an operational status of applicability of a solution type and the proposal of a contract that can readily be signed by the contractors ?
justification:	time is money
Criterion 2:	COHERENCE OF THE TOTAL CONTRACT
definition:	what's the degree of coherence between the clauses constituting the contract?
justification:	a contract is not just a set of individual clauses put together; clauses may refer to one another, contradict or precise one another. The degree of coherence depends on attributes such as non-contradiction, absence of noise, correct internal references.
Criterion 3:	COMPLETENESS OF CONTRACT

definition:	to what degree the contract covers the different aspects of the relationship between contractors?
justification:	aspects of the relationship between contractors about which the contract is silent are governed by the default law rules.
Criterion 4:	POSSIBILITY TO DRAFT A PERSONALISED CONTRACT
definition:	to what degree the solution type permits to design a contract adapted to a particular case?
justification:	confer our formulation of the problem of contract writing under 1.1.3.
Criterion 5:	POSSIBILITY TO UPDATE THE SOLUTION TYPE
definition:	to what degree the solution type can incorporate modifications of the law environment?
justification:	the law environment evolves through new default rules, new imperative law rules and changes in jurisprudential doctrine
Criterion 6:	POSSIBILITY TO INCORPORATE IN THE SOLUTION TYPE THE GROWING EXPERIENCE
definition:	to what degree the solution type can incorporate the growing experience of the contract writer?
justification:	the contract writer's experience evolves through discovery of new subproblems and the invention of conventional clauses to solution those new subproblems
Criterion 7:	COST ON CREATION OF THE SOLUTION TYPE
definition:	what are the ressources consumed in order to get the solution type operational?
justification:	ressources are money
Criterion 8:	COST ON USE OF THE SOLUTION TYPE
definition:	what are the ressources consumed in order to apply the operational solution type to a particular case?
justification:	ressources are money

1.3.2. The comparison.

	MODEL CONTRACT SOLUTION TYPE	STANDARD CONTRACT SOLUTION TYPE	DOCTRINAL TREATY SOLUTION TYPE	COMPILATION OF CLAUSES SOLUTION TYPE	BY THE EXAMPLE SOLUTION TYPE
QUICKNESS OF OBTAINMENT OF A CONTRACT	HIGH	HIGH	VERY LOW	LOW	HIGH
COHERENCE OF THE TOTAL CONTRACT	HIGH	HIGH	VARIABLE	VARIABLE	HIGH
COMPLETENESS OF THE CONTRACT	HIGH	HIGH	VARIABLE	VARIABLE	HIGH
POSSIBILITY TO DRAFT A PERSONALISED CONTRACT	REDUCED	REDUCED	HIGH	HIGH	REDUCED
POSSIBILITY TO UPDATE THE SOLUTION TYPE	HIGH	HIGH	REDUCED	REDUCED	REDUCED
POSSIBILITY TO INCORPORATE IN THE SOLUTION TYPE THE GROWING EXPERIENCE	HIGH	HIGH	REDUCED	REDUCED	REDUCED
COST ON CREATION OF THE SOLUTION TYPE	MEDIUM	MEDIUM	HIGH	HIGH	HIGH
COST ON THE USE OF THE SOLUTION TYPE	VERY LOW	VERY LOW	HIGH	MEDIUM	VERY LOW

1.3.3. Conclusion.

Each solution type has its advantages and limitations. Model contracts and standard contracts offer the advantage of quick obtainment, high coherence, completeness, easy modification and reduced cost of creation and use. On the other hand, their major drawback is their reduced possiblity to draft personalised contracts. The model contract solution type will work only when the group of possible contractors is sufficiently homogeneous and its terms and conditions are sufficiently reconciling in order to obtain the adhesion of cocontractors. The standard contract solution type will work only when either there is no interest for an individualised contract or the proposing contractor is strong enough to impose its will.

The doctrinal treaty solution type is not a to the point solution to the problem of contract writing. It is limited to establishing and clarifying the framework in which the problem of contract writing is posed. As such it offers on impartial tool (as opposed to the model contract and standard contract solution types) but suffers from very slow obtainment, uncertain coherence and completeness, reduced updating possibility and high costs on creation and use.

The compilation of clauses solution type constitutes as the doctrinal treaty solution type an impartial tool. It permits to draft personalised contracts . However the solution type suffers from slow obtainment, uncertain coherence and completeness, reduced possibility to update , important costs of creation and non negligible costs of use.

The by the example solution type offer the advantage of quick obtainment, high coherence and completeness and low cost of use. Major drawbacks of the example solution type are its high creation costs, its reduced possibility to draft personalised contracts and its limited possibility to update.

1.4. OUTLINE OF A MORE DESIRABLE SOLUTION TYPE. POSSIBLE RESEARCH DIRECTIONS.

The preceding analysis suggests the properties to be present in a more desirable solution type :

- quick obtainment of a contract through the solution type
- high coherence between the clauses of the produced contract
- completeness of the contract produced
- possibility to draft a personalised contract
- possibility to update the solution type in regard of the evolution of the law environment
- possibility to adapt the solution type in regard of the evolving experience of the contract writer
- low cost of creation of the solution type
- low cost of use of the solution type

If these are the objectives to be reached by a more desirable solution type, we feel the best way to work them out would be by following two research directions. The first direction derives from the belief a more desirable solution could try to integrate the desirable properties of existing solution types without having its drawbacks. The first research direction aims at integrating properties of the existing solution types in a more general solution type, that at the same time would help to overcome the defaults of existing solution types. The second direction derives from our formulation of the problem of contract writing. We believe that contract writing constitutes a semi-structured task. As such it permits a synthesis of human judgement and computer capabilities. The second research direction aims at evaluating the possibilities of a decision support system in the field of contract writing.

CHAPTER 2

BASIC REQUIREMENTS FOR A DSS FACING THE PROBLEM OF CONTRACT WRITING

The conclusion we reached at the end of chapter 1, suggested the belief a superior approach to the problem of contract writing should examine the possibilities of a decision support system (DSS) for the contract writer. We will justify this belief in 2.1. After a short description of KEEN's analysis schema for DSS, we will apply this structuring schema to the problem of contract writing (2.2.) and retain the basic requirements we think appropriate to the problem of contract writing (2.3.).

2.1. WHY THE DSS APPROACH CONSTITUTES A SUPERIOR APPROACH TO THE PROBLEM OF CONTRACT WRITING

2.1.1. The term DSS qualifies to the situation.

According to KEEN [12], the label DSS is meaningful only in situations where the "final" system must emerge through an adaptive process of design and usage. The reasons he invokes for adopting such a definition seem to be quite transferable to the situation of contract writers.

1. The designer and the contract writer cannot easily provide functionnal specifications or may be unwilling to do so. As dealing with a semistructured task such as the problem of contract writing we either lack the knowledge necessary to lay out procedures and requirements (i.e. the degree of structure is perceptual) or feel that such a statement can never be made (the lack of structure is intrinsic to the task).

2. Users do not know what they want and designers do not understand what users need or can accept : an initial system must be build to give users something concrete to react with.

3. User's concepts of the task or decision situation will be shaped by the DSS. The system stimulates learning and new insights which in turn stimulate new uses and the need for new functions in the system.

4. Intended users of the system have sufficient autonomy to handle the task in a variety of ways or to differ in the way they think sufficiently, that standardisation is prevented. In this situation any computer support must allow personalised usage and must be flexible.

2.1.2. Why the DSS approach is a superior approach to the problem of contract writing

The DSS approach as analysed by KEEN, constitutes a superior approach to the problem of contract writing in that it offers a methodological framework permitting to attack the problem of contract writing systematically. KEEN proposes to analyse DSS from the following points of view:

1. the structure of the task the DSS adresses.
2. the internal dynamics of a DSS.
3. the support of cognitive processes of the invididual decision maker.
4. contextual issues in DSS development.

Each of these viewpoints will be introduced and their relevance for a DSS in the field of contract writing will be indicated.

2.2. KEEN'S ANALYSIS APPLIED TO THE PROBLEM OF CONTRACT WRITING

2.2.1. The structure of the task addressed by the DSS.

KEEN caracterises DSS as systems adressing semistructured tasks. Structured tasks can be automated or routinized thus replacing judgement, while unstructured ones involve judgement entirely and defy computerization. Semistructured tasks permit a synthesis of human judgment and the computer capabilities. This explains the concept of support.

The task of contract writing qualifies as a semistructured task. First, there is no generally approved definition of the problem to solution. The formulation of the problem of contract writing we gave, may be viewed, despite our best effort, as a partial or biassed statement of the problem. Secondly, the task of contract writing involves subtasks which can be automated and subtasks for which a complete routinization is impossible. Thirdly, the choice between known solution types is uncertain.

KEEN poses the question whether structure is perceptual or intrinsic to the task. STABELL [16] points out that organisations often decide to treat an instructured task as if it were structured. Saying that the structure of the problem of contract writing is perceptual means that whatever the intrinsic structure might be it is the perception of the observer who discovers the structure of the task. We don't belief in the existence of intrinsicly unstructured tasks and adopt the view that science has a role of discovering and proposing structures it perceives in a problem space. We think it's preferable to have some structure as work object rather than stating that the problem is intrinsicly unstructured. This implies that we should find out and propose the structure of the problem of contract writing.

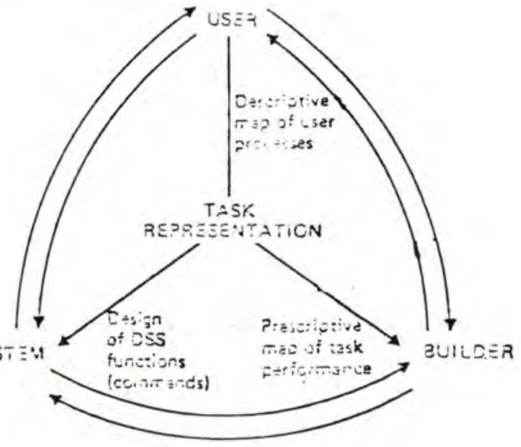

USER
Descriptive map of user processes
TASK REPRESENTATION
Design of DSS functions (commands)
Prescriptive map of task performance
SYSTEM
BUILDER

Figure 2-1: TASK ADDRESSED BY THE DSS

FIGURE 2-1 introduces components of the DSS. A DSS has a USER, the person which communicates with the builder in order to obtain a system. The BUILDER is the person designing and implementing the system. The SYSTEM is the set of data and functions that should assist the user to solve his decision problem more effectively or more efficiently. The task representation of the DSS involves a descriptive map of user processes, a prescriptive map of task performance, and the design of DSS functions. The descriptive map of user processes tries to link the task representation of the DSS to the decision processes of the user. The prescriptive map of task performance are the constraints under which or suggestions by which the user decision processes can be made more efficient or more effective. The design of DSS functions proceeds from the descriptive map and the prescriptive map and defines what functions the DSS should have to support the user decision process.

Applying KEEN's framework on the structure of task implies that we try to

1. describe the decision process of a contract writer
2. propose amendments in order to enhance efficiency and or effectiveness of the contract writer's decision process
3. discover DSS-functions adapted to 1) 2)

2.2.1.1. Description of the decision process of a contract writer

Given a demand for a contract expressed by a subject of law, the contract writer tries to find out what type of contract his client wants to conclude. Once he knows the type of contract he identifies the desiderata of his client for the contract. In function of these desiderata and the state of the law environment the contract writer has knowledge of, he must decide to select a set of clauses that are adapted for the case. The contract writer will also have to know what are the desiderata of the cocontractor. If the contract determined by the desiderata of the cocontractor is different from the contract for the client, the contract writer must identify the points on which the two contracts are different and propose clauses which may reconcile both parties. The result of his decision process is a contract reflecting the common intent of both partners.

Let's analyse the process of contract writing top down in order to discover subtasks

```
                WRITE A CONTRACT
               /
              /
             /
            /
           /
          /
         /___FIND OUT WHAT CONTRACT TYPE IS NEEDED
        *
        *____HAVE AN OPERATIONAL KNOWLEDGE OF THE STATE OF
        *     THE LAW-ENVIRONNEMENT
        *
        *----IDENTIFY THE DESIDERATA OF THE CLIENT
        *
        *----WRITE A CONTRACT ADAPTED TO IDENTIFIED
        *    DESIDERATA OF THE CLIENT
        *
        *----IDENTIFY THE DESIDERATA OF THE COCONTRACTOR
        *
        *----DETECT DIFFERENCES BETWEEN THE CONTRACT ADAPTED
        *    TO THE IDENTIFIED DESIDERATA OF THE CLIENT AND
        *    THE CONTRACT DETERMINED BY THE DESIDERATA OF THE
        *    COCONTRACTOR
        *
        *----PROPOSE CLAUSES WHICH MAY RECONCILE BOTH PARTNERS
        *    IF DIFFERENCES BETWEEN THE TWO CONTRACTS ARE
        *    DETECTED
        *
        *----PUT THE CLAUSES FOR WHICH THERE IS A COMMON INTENT
        *    TOGETHER IN A DOCUMENT CALLED CONTRACT
```

A further breakdown of subtasks delivers

```
FIND OUT WHAT CONTRACT TYPE IS NEEDED
        *
        *----have a knowledge of contract types
        *
        *----have selection criteria to choose a type of contract
        *
        *----have information to apply the selection criteria
        *
        *----apply the selection criteria
        *

HAVE AN OPERATIONAL KNOWLEDGE OF THE STATE OF THE LAW ENVIRONMENT
        *
        *----know the objects to be regulated in a contract type
        *
        *----know the different manners by which the objects in a
        *    contract type can be regulated
        *
        *----know about and have acces to documents describing the state
        *    of the law environment (court decisions, doctrinal treaties,
        *    laws)
        *
        *----know possible clauses for each manner by which an object
        *    can be regulated
        *
        *----modify, add, suppress objects to be regulated in a contract type
        *
        *----modify, add, suppress manners of regulating objects
        *
        *----modify, add, suppress clauses
        *
        *----modify, add, suppress references to documents describing the
                     state of law environment

IDENTIFY THE DESIDERATA OF THE CLIENT
        *
        *----know the set of possible desiderata for each contract type
        *
        *----determine the particular desiderata of the client
        *
        *----modify, suppress, add possible desiderata

WRITE A CONTRACT ADAPTED TO IDENTIFIED DESIDERATA OF THE CLIENT
        *
        *----know a mapping from the possible desiderata to the clauses
        *    of a contract type
        *
        *----apply the mapping on the particular desiderata of the client
        *
        *----modify the mapping
        *
        *----discuss the adapted contract with the client
        *
        *----adapt the contract in light of criticisms formulated by the client

IDENTIFY THE DESIDERATA OF THE COCONTRACTOR
        *
        *----know the set of possible desiderata for each contract type
        *
        *----determine the particular desiderata of the coconctractor
        *
        *----modify, suppress, add possible desiderata

DETECT DIFFERENCES BETWEEN THE CONTRACT ADAPTED TO THE IDENTIFIED
DESIDERATA OF THE CLIENT AND THE CONTRACT DETERMINED BY THE DESIDERATA
OF THE COCONTRACTOR
        *
        *----have a means to compare two contracts of a contract type
        *
        *----apply that means to the two contracts
        *
        *----determine the differences in terms of objects to be
        *    regulated, and manners of regulating an object

PROPOSE CLAUSES WHICH MAY RECONCILE BOTH PARTNERS IF DIFFERENCES
BETWEEN THE TWO CONTRACTS ARE DETECTED
        *
        *----know about clauses for each manner by which an object
        *    of a contract type can be regulated
        *
        *----detect manners which can reconcile both partners
        *
        *----propose clauses of the detected manners
        *
        *----modify, suppress, add manners which may reconcile
        *
        *----modify, suppresss, add clauses which may reconcile

PUT THE CLAUSES FOR WHICH THERE IS A COMMON INTENT TOGETHER IN A
DOCUMENT CALLED CONTRACT
        *
        *----determine the agreement of each party to the proposed
        *    clauses of reconciliation
        *
        *----take the clauses which didn't need reconciliation
        *
        *----format the agreed clauses of reconciliation and the clauses
        *    which didn't need reconciliation as to obtain a signable format
        *
        *
        *----value the parameters of the clauses
        *    (dates, identification of persons, money units)
```

2.2.1.2. Prescription of tools to support the decision process more effectively and more efficiently

1. As a computer presents memorising capabilites it can be used to memorize the contract types, the objects to be regulated, the different manners of regulating an object, the clauses and references. It can also be used to update the memorised representation of the contracts writers knowledge of the state of the law environment, by permitting to modify, suppress or add contract types, objects, manners, clauses or references.

2. The set of possible desiderata, as well as the particular desiderata of a contractor can be memorised by a computer. Updating facilities can be provided to modify, suppress or add possible desiderata.

3. A computer is capable to effectuate quickly a calculation on the representation of the desiderata of a contractor to determine a set of clauses adapted to those desiderata. You should give him the set of particular desiderata, the calculation rules and the set of clauses. The computer can quickly do such calculation and deliver a hard or light copy of the calculated adapted contract.

4. It's possible to develop a model of a given contract type in function of which a contract can be qualified precisely. As such the qualification of a contract proposed by a contractor can be viewed as a representation of the desiderata of the cocontractor.

5. The calculated adapted contract can also be qualified precisely in terms of a model of a given contract type. If both contracts adapted to the desiderata of each contractor can be qualified in terms of the same model it must be possible to compare these contracts. The computer can print out a listing helping to detect divergence between contracts or permit to consult the divergence at a video terminal.

6. The model could be used not only to qualify contracts, but also to

structure the representation of the state of the law environment.

7. The computer offers text editing, formatting and printing facilities to value the parameters of the clauses, to quickly print out a signable contract or other documents that may be useful.

These seven points - that will be detailed in later chapters - constitute the basic proposals of the builder. The proposals are believed to increase the efficiency and effectiveness of the process of contract writing. By effectiveness we understand the improvement of the quality of the contract produced through the DSS. By efficiency we understand the economic cost to produce a contract of a certain level of quality. Effectiveness is hoped to be increased by enabling the contract writer to consult and update a representation of his knowledge of the state of the law environment, to let him describe and update the set of possible desiderata for a given contract type, to let him describe and update the mapping function between desiderata and clauses, by exhaustive detection of points of divergence and a extensive proposal of clauses for reconciliation. Efficiency is hoped to be increased by provision of storing often used contracts, by the quick production of useful documents (adapted contract, detection of divergence and proposition of reconciling clauses, reconciled contracts)

2.2.1.3. Adapted DSS functions

The discovery of DSS functions proceeds from the confrontation between the description of the decision process of a contract writer and the prescribed proposals of the builder. The following list of DSS functions is derived.

1. create an object to be regulated
2. modify an object to be regulated
3. suppress an object to be regulated
4. consult an object to be regulated
5. consult a contract type
6. create a contract type
7. modify a contract type
8. suppress a contract type
9. suppress a manner of regulating an object

10. create a manner of regulating an object
11. modify a manner of regulating an object
12. consult a manner of regulating an object
13. suppress a contract clause
14. create a contract clause
15. modify a contract clause
16. consult a contract clause
17. suppress a reference to a document describing the state of the law environment
18. create a reference to a document describing the state of the law environment
19. modify a reference to a document describing the state of the law environment
20. consult a reference to a document describing the state of the law environment
21. suppress a possible desideratum
22. create a possible desideratum
23. modify a possible desideratum
24. consult a possible desideratum
25. declare a desideratum for a client
26. modify a desideratum for a client
27. declare the mapping function between desiderata and clauses
28. modify the mapping function between desiderata and clauses
29. suppress the mapping function between desiderata and clauses
30. consult the mapping function between desiderata and clauses
31. modify the adapted contract
32. suppress the adapted contract
33. create the adapted contract
34. represent a contract proposed by the cocontractor
35. modify a contract proposed by the cocontractor
36. consult a contract proposed by the cocontractor
37. delete a contract proposed by the cocontractor

38. consult divergence between two contracts on screen
39. print a document stating the divergence between two contracts
40. consult reconciling clauses for divergent contracts
41. print a document providing reconciling clauses for divergent contracts
42. enter agreed clauses for the reconciled contract
43. modify agreed clauses of the reconciled contract
44. delete agreed clauses of the reconciled contract
45. format the reconciled contract in a signable form
46. value parameters of the reconciled contract

These DSS functions will be detailed and refined in succeeding chapters.

2.2.2. The internal dynamics of a DSS

KEEN represents the adaptive links between the major actors involved in any DSS development and the technical system by FIGURE 2-2.The user, system and builder are already introduced. The arrows represent a direction of influence. For example SYSTEM (S)$\longrightarrow$ USER(U) indicates that learning is stimulated by the DSS, while USER$\longrightarrow$ SYSTEM refers to the personalized, differenciated mode of use that evolves. The two adaptive processes work together : an effective DSS encourages to explore new alternative and approaches to the task (S$\longrightarrow$ U). This in itself stimulates new uses of the system, often unanticipated and idiosyncratic (U$\longrightarrow$ S).

The definition of DSS as applicable in situations where the final system must evolve from adaptive development and use implies the fellowing :

- a system is a "DSS" only if each of the arrows is relevant to the situation,
- where the arrows are relevant, the design process must ensure that they are not blocked by inflexible design structures, failure to allocate ressources for implementing new functions, or lack of direct relationship between user and designer
- each arrow represents a distinctive aspect of research and practice

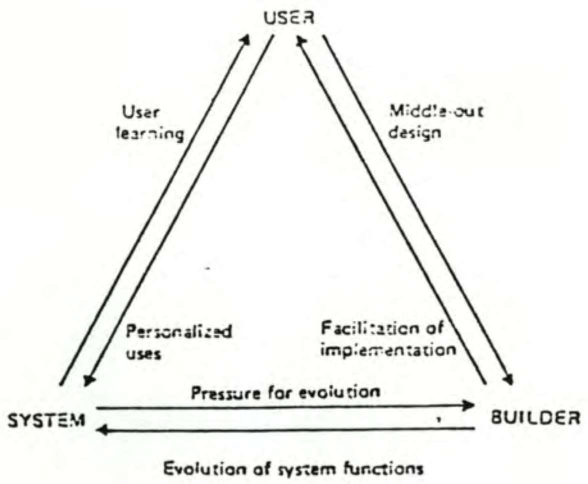


Figure 2-2: THE INTERNAL DYNAMICS OF A DSS

2.2.2.1. The system - user link

The arrows $S \longleftrightarrow U$ may be termed the cognitive loop. The link $S \longrightarrow U$ concerns the learning effect the system exercices on its users. The link $U \longrightarrow S$ concerns the individuals exploitation of the DSS capabilities, their own learning or both. The cognitive loop helps to explain the finding that individuals use a given DSS in different ways and that uses are thus often unintended and unpredicted. This seems a natural outcome of the sequence $S \longrightarrow U \longrightarrow S \longrightarrow U$...

REQUIREMENTS FOR THE SYSTEM BASED ON THE COGNITIVE LOOP

1. The system should enable personalised use. Personalised use of a DSS in support of contract writing could be obtained by letting each contract writer freely

- declare the contract types
- associate the objects to be regulated
- associate the manners of regulating each object
- provide clauses
- define the mapping function used

2. The system should enable a learning effect. The learning effect will result from the clarification of the decision process, the confrontation of desiderata and clauses on the mapping function level, the confrontation of the use of the system and the problems experienced by the contractors.

3. The system should not restrain the dynamics of the cognitive loop. As a contract writer's experience evolves in time, the state of the law environment evolves and the system exercises its learning effects, the user should be able to

   - introduce new contract types
   - update objects
   - update manners
   - update clauses
   - update the mapping function used

2.2.2.2. The user - builder link

The link $U \longleftrightarrow B$ constitutes the implementation loop. The arrow $U \longrightarrow B$ represents the need for middle-out design discussed by NESS and Courdon et al [14]. The middle out approach relies on the quick delivery of an initial system to which users can respond and thus clarifiy their desires. Middle-out-design is the means by which the designer learns from the user; it also ensures that the user drives the design process. The arrow $B \longrightarrow U$ has been explored in studies of DSS implementation that examine the role of "integrating agent" [1] intermediary [11], chauffeur [9] and change agent [8]. A DSS is a service rather than a product, and requires the designer to understand users' perspective and processes, to build credibility and to be responsive to users' evolving needs.

REQUIREMENTS FOR THE SYSTEM BASED ON THE IMPLEMENTATION LOOP.

1. KEEN's requirement for the arrow $B \longrightarrow U$ seems to be satisfied as the author qualifies both as a builder and a user

2. The arrow $U \longrightarrow B$ claims for quick delivery of a prototype.

3. Middle out design should be insured by incremental development of the system.

2.2.2.3. The system - builder link.

The arrows $B\longleftrightarrow U$ constitute the evolution loop. Learning ($B\longrightarrow U$) and personalised use ($U\longrightarrow S$) put strain on the existing system. This builds pressure for evolution ($S\longrightarrow B$). New functions are then provided ($B\longrightarrow S$). This obviously is feasible only if the design architecture is modular, flexible and easily modified, if the programmer can implement new functions inexpensively and quickly, and if the designer maintains ongoing contact with users.

REQUIREMENTS FOR THE SYSTEM BASED ON THE EVOLUTION LOOP.

1. The system to develop should not restrain the dynamics of the evolution loop. The software architecture should isolate in modules the aspects which will probably evolve.
2. As to reduce pressure for evolution the system should be able to insure a large part of the dynamics of the cognitive loop by action of the user on the system and not by action of the builder.

2.2.3. The support of the cognitive process of the individual decision maker

KEEN, in his overview of DSS case studies, mentions an often occurring characteristic that DSS support the cognitive process of individual decision makers. He further insists that if a function in a system does not directly relate to some concept in the user's mind it cannot be used.

This implies that we should find out the concepts which are in the mind of the contract writer, and discover the properties of the cognitive process. Another implication is that whatever the inside of the system to be developped will be, the interface system - user should speak in a terminology familiar to the user.

2.2.3.1. Concepts characterising the cognitive process of contract writers

Beside the concepts introduced in 1.1.1. (contract, principle of contractual freedom, default law rule, qualification rule, imperative law rule, regulated contract, jurisprudential doctrine, contract life cycle, interpretation of contract) we believe the following concepts are implicitely or explicitely present in the mind of most contract writers, be it under varying names.

NAMES TO DESIGNATE THE CONCEPT

contract type, species of contract, kind of contract,

DESCRIPTION OF THE CONCEPT

Common denominator for a selected group of contracts. The selection is made on basis of the economic operation to be carried out or on the juridical nature of the group of contracts.

EXAMPLES

maintenance contract for EDP-hardware, EDP-leasing contract, software licence, software development contract, employment contract for software engineers.

*
* *

NAMES TO DESIGNATE THE CONCEPT

object to regulate in a contract, aspect of relationship of contractors

DESCRIPTION OF THE CONCEPT

For a given contract type a certain number of aspects of the relationship may or can be regulated.Each aspect of the relationship constitutes an objet to regulate in a contract.

EXAMPLES

In a maintenance contract for EDP hardware objects that can be regulated are for instance

- intervention delay after call
- price to be paid for the service
- frequency of preventive maintenance

*
* *

NAMES TO DESIGNATE THE CONCEPT

manner of regulating, way of regulating, generic fashion to rule an aspect of the relationship of contractors

DESCRIPTION OF THE CONCEPT

For a given contract type and a given object to regulate in that contract type, the aspect of the relationship can be regulated in different fashions

EXAMPLES

In a maintenance contract for EDP-hardware the object to regulate 'intervention delay after call' could be regulated in following manners:

- intervention delay not mentioned in contract
- intervention delay mentioned
- intervention delay mentioned and enforced by penalties

*
* *

NAMES TO DESIGNATE THE CONCEPT

contract clause, contract provision, phrase in contract, contract text unit

DESCRIPTION OF THE CONCEPT

Contracts are texts. These texts are composed of text units. The criterion often used in splitting up the contract text in text units is that a text unit is by its semantics relative to an object to regulate. At the same time the unit is an occurrence of a manner of regulating associated to the object.

EXAMPLES

A clause found in existing contracts or imagined by a contract writer to regulate in the contract type 'maintenance contract for sold EDP-hardware ' the object ' intervention delay after call' might be 'The maintenance team will arrive at the site of the client maximum 9 hours after call '.

*
* *

NAMES TO DESIGNATE THE CONCEPT

useful document for writing a contract

DESCRIPTION OF THE CONCEPT

Texts representing knowledge about the state of the law environment: books, articles, Court decisions, personal notes, laws.

*
* *

NAMES TO DESIGNATE THE CONCEPT

mapping function between facts and clauses, correspondance between desiderata and clauses, triggering of clauses in a given context, very useful clause when

DESCRIPTION OF THE CONCEPT

When a contract writer drafts an adapted contract for a particular client, his activity implies the knowledge of when a contract clause is or might be adapted for his particular client. The mapping function represents such knowledge.

EXAMPLES

When 'an EDP configuration has great capacity and low workload ' it is or might be adapted to have a contract clause ' the maintenance team will arrive at he site of the client maximum x hours after call' in wich no penalty is insured. When 'the residual capacity of an EDP configuration is very reduced and unavailability of the configuration certainly has severe financial consequences' it is or might be adapted to have a contract clause ' the maintenance team will arrive at the site of the client maximum x hours after call. The maintenance firm shall incur a penalty of y moneyunits for each hour of delay depassing the guaranteed intervention delay '.

2.2.3.2. Properties of the cognitive process of a contract writer

The cognitive process of a contract writer seems to be caracterised by following properties:

1. SPECIALISATION

When observing the set of all contrat writers and the set of all contract types, and think about the relation between the first set and the second set representing "contract writer X is familiar with or is highly confronted with contract type Y",we discovered that a given contract writer is only interested in a limited group of contract types and that not all contract writers were interested in the same contract types. The property of specialisation requires that the system to be developped should be general to cover the set of possible contract types and should be tailorable to the specific interest sphere of a given contract writer or group of contract writers.

2. INDIVIDUALISATION

When observing two contract writers A and B interested in the same contract type and comparing their cognitive process in terms of objects to be regulated, manners of regulating and contract clauses it's highly probable to remark the following differences:

- the number of objects to be regulated
- the set of objects to be regulated
- the definition of an object to be regulated
- the number of manners of regulating associated to an object to be regulated
- the set of manners of regulating associated to an object to be regulated
- the number of known contract clauses

- the set of known contract clauses

The property of individualisation requires that the system to be developped should be flexible to the individiualised perception and knowledge a contract writer has of a given contract type. The flexibility required implies that the system should admit that a contract writer communicates:

- his set of objects to be regulated
- his definition of each object to be regulated
- his set of manners of regulating associated to an ob ject to be regulated
- his definition of each manner of regulating associated to an object to be regulated
- his set of known and preferred contract clauses

3. EVOLUTION

When observing a given contract writer A interested in a given contract type and comparing the cognitive process of A at two points of time T1 and T2 in terms of objects to be regulated, manners of regulating and contract clauses it's highly probable to remark following differences :

- new objects to be regulated are present at T2
- objects present at T1 have disappeared at T2
- definitions of an object to be regulated are changed
- new manners of regulating associated to an object to be regulated are present at T2
- manners associated to an object to be regulated have disappeared at T2
- new contract clauses are present at T2
- contract clauses present at T1 have disappeared or are modified at T2

This evolutionary caracter of the cognitive process is due on the one hand to the evolution of the state of the law environment, on the other hand to what can be called the "experience" of the contract writer. New laws modifying old or creating new default law rules or imperative law rules and changes in jurisprudential doctrine oblige the contract writer to adapt his knowledge. During the elapsed period T2-T1, the contract writer has practised his knowledge on different cases. The contract writer learns from his clients, becomes more familiar with the operation to be carried out, discovers new subproblems and new ways to deal with them.

The property of evolution requires that the system to be developed should be adaptable to the growing experience of the contract writer and capable of facing the evolving states of the law environment.

4. GENERALITY

When observing the cognitive process of a large set of contract writers and a large set of contract types, subjected to properties of specialisation, individualisation and evolution ,we think that what remains stable are the concepts contract type, object to be regulated, manner of regulating, clause, mapping function and the types of association between those concepts. What varies are the occurrences of the concepts and the occurrences of the association types.

2.2.4. Contextual issues in DSS development.

KEEN proposes figure 2-3 to represent contextual issues in DSS development.

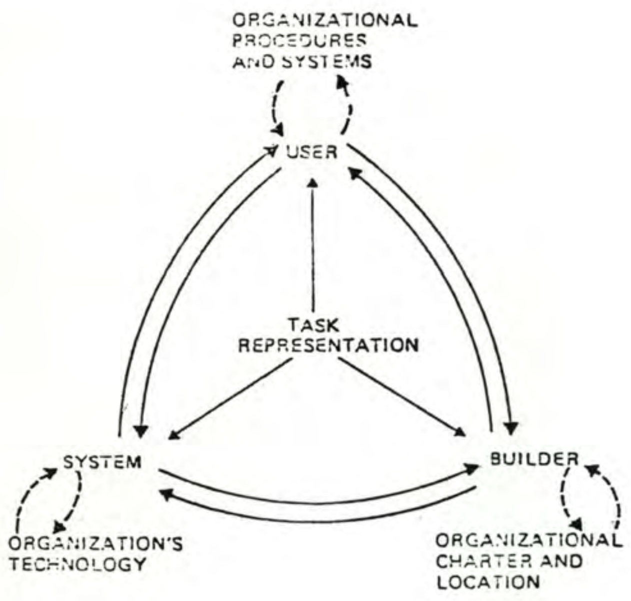

ORGANIZATIONAL PROCEDURES AND SYSTEMS
USER
TASK REPRESENTATION
SYSTEM
BUILDER
ORGANIZATION'S TECHNOLOGY
ORGANIZATIONAL CHARTER AND LOCATION

Figure 2-3: CONTEXTUAL ISSUES IN DSS DEVELOPMENT

The additional links are not so much adaptive as limiting influences. For example, organizational procedures may constrain user discretion and behaviour (O⟶ U). In turn the extent to which the user or users can influence procedures (U⟶ O) limits the organizational learning a DSS can stimulate.

The DSS itself is constrained by the organization's available technology (T⟶ S).A system can be realised using different technologies. The used technology influences performance attributes of the system. The link $S \longrightarrow T$ is a reminder that learning and evolution can be blocked by the unavailability to obtain additional technology.

The implementation loop relies on facilitation and middle out design. This requires a close cooperation between the user and the builder and in particular that :

- builder and user are not geographically or psycologically isolated
- the builder has a mandate for innovation
- the charging organization reasons not exclusively in terms of 'hard' benefits, but is sensitive to qualitative benefits. DSS "improve" a decision process and it is unlikely that one can point in advance to a "bottom line" payoff, especially if the value of the system is in the end determined by an adaptive evolutionary process.

2.3. BASIC REQUIREMENTS FOR A DECISION SUPPORT SYSTEM FACING THE PROBLEM OF CONTRACT WRITING

The confrontation of KEEN's general framework for analysis of DSS and the problem of contract writing enables us to produce a list of basic requirements for a decision support system facing the problem of contract writing.

REQUIREMENTS ASSOCIATED TO THE STRUCTURE OF THE TASK ADDRESSED BY THE DSS

R1:	functions of the system should correlate to the subtasks discovered in the decision process of the contract writer
R2:	functions of the system should contribute to improving the effectiveness and (or) the efficiency of the contract writing process

REQUIREMENTS ASSOCIATED TO THE INTERNAL DYNAMICS OF THE DSS

R3:	the system should enable personalised use
R4:	the system should enable a learning effect
R5:	the system should not restrain the dynamics of the cognitive loop
R6:	the final system should be preceded be the delivery of a prototype

R7: the final system should be incrementally developable

R8: the software architecture should isolate in specific modules those aspects which are probable to change

REQUIREMENTS ASSOCIATED TO THE SUPPORT OF THE COGNITIVE PROCESS

R9: the system should support the concepts in the mind of a contract writer

R10: the system should speak the language of the contract writer

R11: the system should be usable for specialised contract writers

R12: the system should permit individualised use

R13: the system should permit an evolutionary use

REQUIREMENTS ASSOCIATED TO THE CONTEXTUAL ISSUES

R14: absence or reduced presence of contextual contraints

CHAPTER 3

THE INTEGRATION OF EXISTING SOLUTION TYPES IN A MORE GENERAL SOLUTION TYPE

The existing solution types of the problem of contract writing offer advantages and drawbacks of which the major ones were mentionned in 1.3. of chapter 1. This chapter proposes ideas for a general solution type which would incorporate desirable properties of existing solution types and permit to overcome their drawbacks. This research direction is useful as the existence of a new solution type is only justified provided it is better than existing solution types. The existing solution types offer each specific advantages and are good or bad depending on the situation they address. Integrating existing solution types in a more general solution type permits to carry over the advantages of the existing solution type and to apply a solution type when it is appropriate. Another justification for this research direction is that the users of the system to develop will accept more easily the system when finding back procedures which are not completely sew to them.

In following sections we will reexamine each existing solution type trying to discover what aspects of the solution type could be taken over to the more general solution type. Second we will indicate what drawbacks of existing solution types should be overcome in the more general solution type. We conclude the chapter by drawing up an inventory of requirements for the more general solution type.

3.1. THE MODEL CONTRACT SOLUTION TYPE

The main benefit of the model contract solution type is its caracter ' take it and sign it '. This benefit can only be obtained provided two underlying hypotheses are present. First the group of contractors using the model contract must be homogeneous in the sense that all contractors of the group have more or less the same interests. Second, the terms of the model contract should be sufficiently reconciling in order to be accepted without discussion by the cocontractor. As soon as these hypotheses are not verified, the contractors are forced to modify the model contract on the points they cannot accept. It would be nice if the more general solution type could incorporate 'take it and sign it' contracts that are adapted to groups which are homogeneous and that are sufficiently reconciling. It would be nice if the more general solution type could assist when the underlying hypotheses don't hold by proposing clauses that may reconcile contracting parties.

3.2. THE STANDARD CONTRACT SOLUTION TYPE.

The main benefit of the standard contract is also its caracter ' take it and sign it'. This benefit can only be otained provided the contractor proposing the standard contract is strong enough and has interest to impose the same terms for all possible cocontractors. As soon as this hypothesis is not verified, the contractors will amend the standard contract proposed. It would be nice if the more general solution type could incorporate standard contracts and assist the contract writer when the underlying hypothesis doesn't hold by proposing clauses that may reconcile contracting parties.

3.3. DOCTRINAL TREATY SOLUTION TYPE

The main benefit of the doctrinal treaty solution type resides in its impartial representation of the state of the law environment. The doctrinal treaty constitutes a synthesis of different documents applicable to a certain number of contract types : laws, regulateations, Court decisions, articles, books. As the contract writer must have access to those documents, it would be nice if the more general solution could incorporate references to needed documents. The doctrinal treaty sometimes suggests in what contexts, what clauses may be adapted to the interest of a particular contractor. It would be nice if the more general solution could manage systematically the link between contexts and clauses. It would be nice if the more general solution could incorporate the impartial caracter of the doctrinal treaty solution type. As the doctrinal treaty solution type lacks the possibility of efficient obtainment of a contract, it would be nice if the more general solution type could increase the efficiency of the obtainment of a contract.

3.4. THE COMPILATION OF CLAUSES SOLUTION TYPE

The compilation of clauses solution type offers an impartial representation of possible clauses usable in a certain number of contract types. It would be nice if the more general solution type could incorporate such impartial representation of possible clauses. The author of a compilation of clauses often gives indications about what clauses are appropriate under what circumstances. It would be nice if the more general solution could incorporate such knowledge. As the author of a compilation of clauses often gives indications about what clauses may reconcile contractors when they desagree, it would be nice if the more general solution type could propose clauses for reconciliation. The compilation of clauses doesn't offer an easy mechanism to obtain a contract. In fact the contract writer adopting the compilation of clauses solution type is, once he has selected the clauses adapted to his case, obliged to copy the retained clauses and to format them. It would be nice if a more general solution type would provide copying and formating facilities

3.5. THE BY THE EXAMPLE SOLUTION TYPE

The by the example solution type constitutes another occurrence of ' take it and sign it '. The hypothesis for applicability of this solution type is the existence of contract types where parties have little or no interest to discuss the contract. Either the contract is a regulated contract or the parties have not strongly opposite interests. It would be nice if the more general solution type could incorporate such contracts. The by the example solution type is not useful in the field of contracts governed by the principle of contractual freedom where there is interest for reconciling opposed interests. It would be nice if the more general solution type could cover also those contracts.

3.6. REQUIREMENTS FOR A MORE GENERAL SOLUTION TYPE

In preceding sections we have indicated for each existing solution type the properties which might be interesting to integrate in the more general solution. We also indicated what features the more general solution should have to overcome defaults present in existing solution types. Here follows a list of requirements summarising our findings.

R15. The system to develop should be capable of representing model contracts

R16. The system to develop should be capable of proposing clauses that may reconcile parties that disagree on the terms of model contract

R17. The system to develop should be capable of representing standard contracts

R18. The system to develop should be capable of proposing clauses that may reconcile parties that disagree on the terms of a standard contract

R19. The system to develop should permit access to documents describing the state of the law environment

R20. The system to develop should be able to represent knowledge on when to use what clause

R21. The system do develop should be able to represent an impartial knowledge of clauses usable in a given contract type

R22. The system to develop should be capable of selecting clauses adapted to a given situation (supposing R20 and R21 already satisfied)

R23. The system to develop should be capable of copying selected clauses adapted to a given situation and of formatting those clauses

R24. The system to develop should be capable of representing contracts of the by the example solution type.

CHAPTER 4

CONTRACTUAL MODELLING

In this chapter we try to evaluate the possibilities of modelling to meet the requirements stated at the end of chapters 2 and 3. In 4.1. we will most generally define the term model, justify its use and propose some quality criteria. In 4.2. we introduce the proposed modelling. In 4.3. we will provide some methodological guidelines for creating the model we propose. 4.4. deals with the dynamics of the model and exposes factors influencing the evolution of the model. 4.5. states the quality criteria by which a model can be evaluated and in 4.6. the economic feasibility of the proposed modelling is examined.

4.1. THE MODEL, ITS JUSTIFICATION AND SOME QUALITY CRITERIA

4.1.1. Definition of the term model

A model can be defined as a set of concepts and association types between those concepts used for analysing a given reality. Saying the same in a more imaginary language one could state a model constitutes a framework providing pair of glasses obliging the observer to look at a given reality in a predetermined way. This can be expressed by figure 4-1.

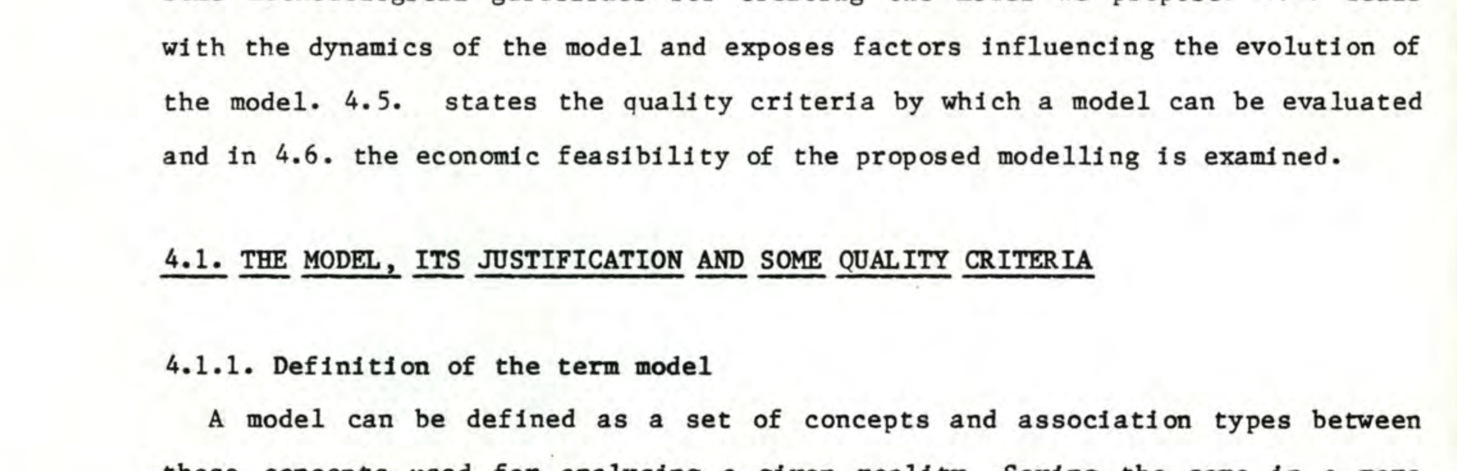

GIVEN REALITY
IMAGE OF GIVEN REALITY = CONTENTS OF STRUCTURE
FRAMEWORK = STRUCTURE OF CONTENTS

Figure 4-1: A WAY TO LOOK AT MODELS

Saying the same in a more precise language.

Given : - a reality R
- a model M composed of a set of concepts $C = \{C1, C2, C3\}$ and a set of association type between those concepts $A = \{A1, A2\}$

an observer O will examine the reality R and try to find out what aspects of the reality are occurrences of C1, C2, C3, A1, A2. The result of his observation will be an image of R, called I = { (C1, R1), (C2, R2), (C3, R3), (A1, R4), (A2, R5)} , where R1, R2, R3, R4, R5 are aspects of the reality R, where the relation I is defined as (X, Y) ∈ I when Y is an occurrence present in R of a concept or association type X of M.

4.1.2. Why using models to observe reality

1. The model helps to concentrate on particular aspects of the reality.

   Most observer are not interested equally in all aspects of a given reality. Often observers have a purpose which limits the aspects of the reality in which they are interested. When looking at the reality of people exchanging goods and services, an economist will be interested in economic aspects, a lawyer will be interested in juridical aspects. The model orients the view the observer has to those aspects the observer is interested in.

2. The model forces to have a systematic observation of particular aspects of the reality.

   When an observer must make many observation of a given reality, he will record different images I1, I2, I3, of a stable structure and varying contents. An image is complete when for each concept and association type of the model, the observer has noted the corresponding occurrences in reality.

3. The model facilitates communication between observers.

   Communication between observers is facilitated when they speak a common language. The model precises a language offering its concepts and association types as communication tool.

4.1.3. Quality criteria for a model

Some models are said to be better than others. This implies we have knowledge or at least intuition about criteria that permit to say when a model is better than another. We think following criteria may be useful. Each criterion is defined and justified

CRITERION 1: PURPOSEFUL CONCENTRATION

Comment: A group of observers of a given reality R is by its purpose only interested in some apects of the reality R. The model M determines the image I the observers have of the reality R. The model is bad when the image I is silent about some aspects of interest or when the image I provides information on aspects in which the observers are not interested

CRITERION 2: CLEARNESS OF DEFINITION OF THE MODEL

Comment: The model M is applied on a reality R to obtain an image I. The application of the model is facilitated when a clear, precise and understandable description is given of its concepts and association types.

CRITERION 3: QUALITY OF PERCEPTION OF THE INTEREST AREA

Comment: Suppose different models M1, M2, M3, all purposefully concentrating on the aspects of the reality R in which observer O is interested. We think there is some freedom for the builder of the model to use varying sets of concepts and association types which all will purposefully concentrate but will condition different qualities of perception.

CRITERION 4: EVOLVABILITY OF THE MODEL

Comment: Different aspects of the environment of the modeller may change in time : the reality R, the observer O, the purpose of the observer O, the perception of the observer O. To what degree a model M1 good at time T1 can de adapted to remain good at time T2 ? Also if the aspects of the environment of the modeller remain stable in time, a bad model at time T1 should be able to become a good one at time T2.

4.2. THE PROPOSED MODELLING

4.2.1. Meta-model and models

The analysis of the cognitive process of contract writers showed the presence of some universal concepts such as contract type, object to regulate and manner of regulating. The most universal association types between those concepts are:

AT1: a contract type 'can or may regulate' an object to regulate
AT2: an object to regulate 'can or may be regulated in' a manner of regulating.

If these concepts and association types seem to be fundamental in the knowledge of contracter writers, our analysis also discovered that their knowledge was specialised (limited to certain contract types), individualised (varying sets of objects to regulate and varying sets of manners of regulating among contract writers) and of an evolutionary nature (varying sets of objects to regulate and varying sets of manners of regulating at successive points in time for a same contract writer)

We believe the specialised, individualised and evolutionary knowledges of specific contract writers can be viewed as occurences of a general structure of their knowledge.

Therefore we propose the system should be based on a meta-model constituted by the concepts contract type, object to regulate and manner of regulating. The specific contract writer will be enabled to express his specialised and individualised knowledge as an occurrence of the meta-model. The evolutionary nature of his knowledge can be expressed by changing the occurrences.

The conceptual schema representing the meta-model is given in figure 4-2.

A model for a contract type is constituted by naming and defining the contract type, by naming and defining the objects to regulate in the contract type and by naming and defining the manners of regulating each object.

The specialised contract writer will constitute models only for those contract types he is confronted whith or interested in.

Each contract writer is free to name and define the objects he considers to be dealt with in a contract type. For each object he has knowledge of, the contract writer is free to associate different manners of regulating the object, which he names and defines as thought appropriate.

The evolutionary nature of his knowledge, due to changes in the state of the law environement and to the evolving experience of the contract writer is faced by enabling the contract writer to modify the model. Such modifications could be on following levels:

- the renaming of existing contract types, objects to regulate, manners of regulating
- the redefinition of existing contract types, objects to regulate, manners of regulating
- the introduction of new contract types, objects to regulate, manners of regulating
- the suppression of the existing contract types, objects to regulate, manners of regulating

A model for the contract type 'maintenance of sold EDP-hardwaré can be found in APPENDIX I.1.

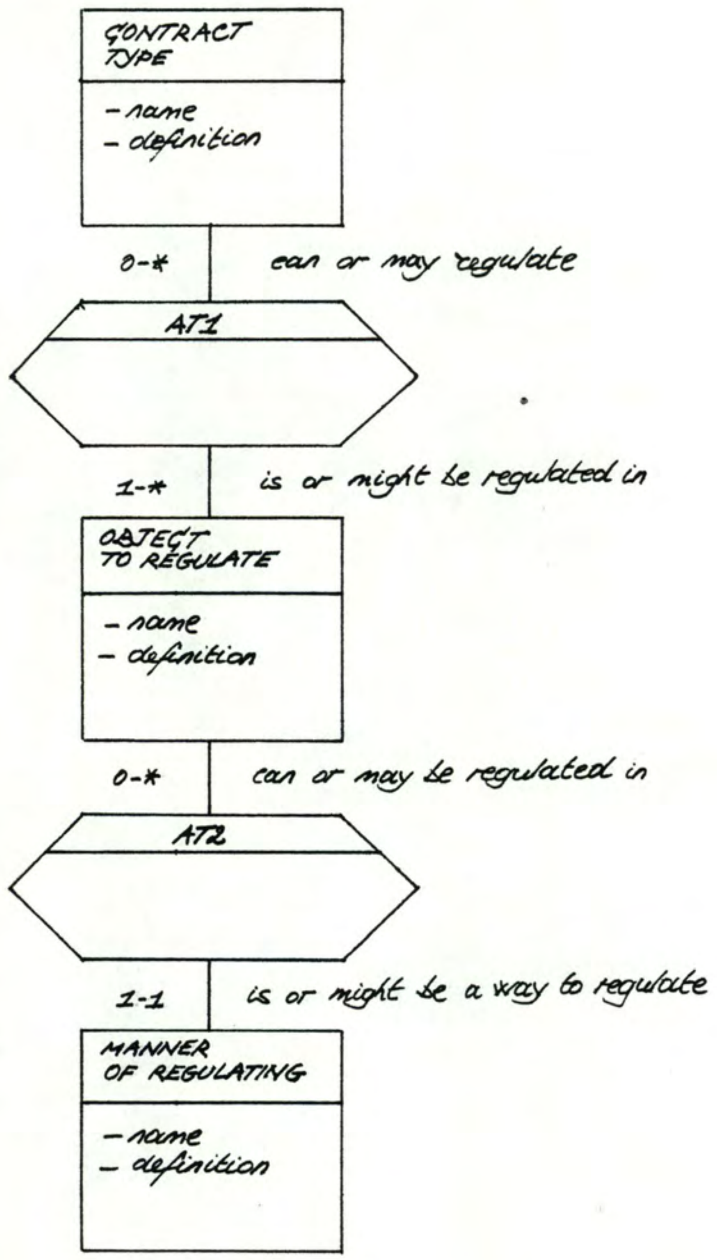


Figure 4-2: CONCEPTUAL SCHEMA NUMBER 1

4.2.2. The model as a tool for representation of a contract's semantics

A model is useful when observing the reality of contracts. The contract writer has a purpose - write contracts - which requires knowledge of the semantics of a contract. The model provides a structuring framework in terms of which the semantics of a contract can be analysed.

A contract can be viewed as a text composed of a certain number of text units which lawyers call clauses. A clause is by its semantics relative to an object to be regulated in a contract type. On the other hand a clause deals with that object to be regulated in a specific manner. Conclusion is to discover two forms of representation of a contract's semantics

1. as a text

2. as a set of triplets (object regulated, manner of regulating, clause)

The activity by which we derive from the first form of representation of a contract's semantics the second form is called analysis. The analysis supposes a contract's text to be analysed and a model of the contract type. The analysis may proceed in two passes. In pass 1, the analyst examines the contract text and discovers text units which deal with the objects of the model. In pass 2, the analyst decides for each text unit what's the manner of regulating by which the text unit deals with the object. The result of analysis is a qualified contract. This means the semantics of each of its clauses is expressed in terms of objects and manners.

An example of the analysis of DIGITAL's maintenance contract in regard of the model can be found in appendix I.2.

4.2.3. The model as a tool for comparison

We introduced the second form of representation in order to be able to satisfy the requirement of support of an important subtask of the decision process: the comparison of contracts.

To our knowledge computers can compare texts but not their semantics. The second way of representation of a contract's semantics being applied on two contracts it becomes feasible for a computer to detect divergence or non divergence between contracts. This is because the second form of representation has recourse to a stable symbolisation of semantics.

Given

- a contract type T
- its associated model M = {(O1, M1),(O1, M2), (O1, M3), (O2, M4), (O2, M5),(O3, M6)}
- a contract C1 of which the text is {C11, C12, C13}
- a contract C2 of which the text is {C11, C22, C23}
- the analysis of contract C1 delivered the set of triplets {(O1, M2, C11), (O2, M4, C12), (O3, M6, C13)}
- the analysis of contract C2 delivered the set of triplets {(O1, M2, C21), (O2, M5, C22), (O3, M7, C23)}

The computer could be used to detect that :

- O1 is regulated in the same manner M2 in contracts C1 and C2,
- O2 is regulated differently in contracts C1 (manner M4) and C2 (manner M5),
- O3 is regulated differently in contracts C1 (manner M6) and C2 (manner M7).

A listing providing a comparison between the maintenance contract proposed by DIGITAL and the contract wanted by his cocontractor is given in appendix I.5.

Preceding remarks introduce two new entities and four new association types to the conceptual schema of figure . The expanded conceptual schema is represented by figure 4-3.

4.2.4. The model as a tool for reconciliation

Our analysis of the subtasks of the contract writing process mentionned the reconciliation of divergent contracts. The model introduced so far could help to detect divergence but is in no way capable of proposing manners of regulating and clauses that reconcile divergent contracts.

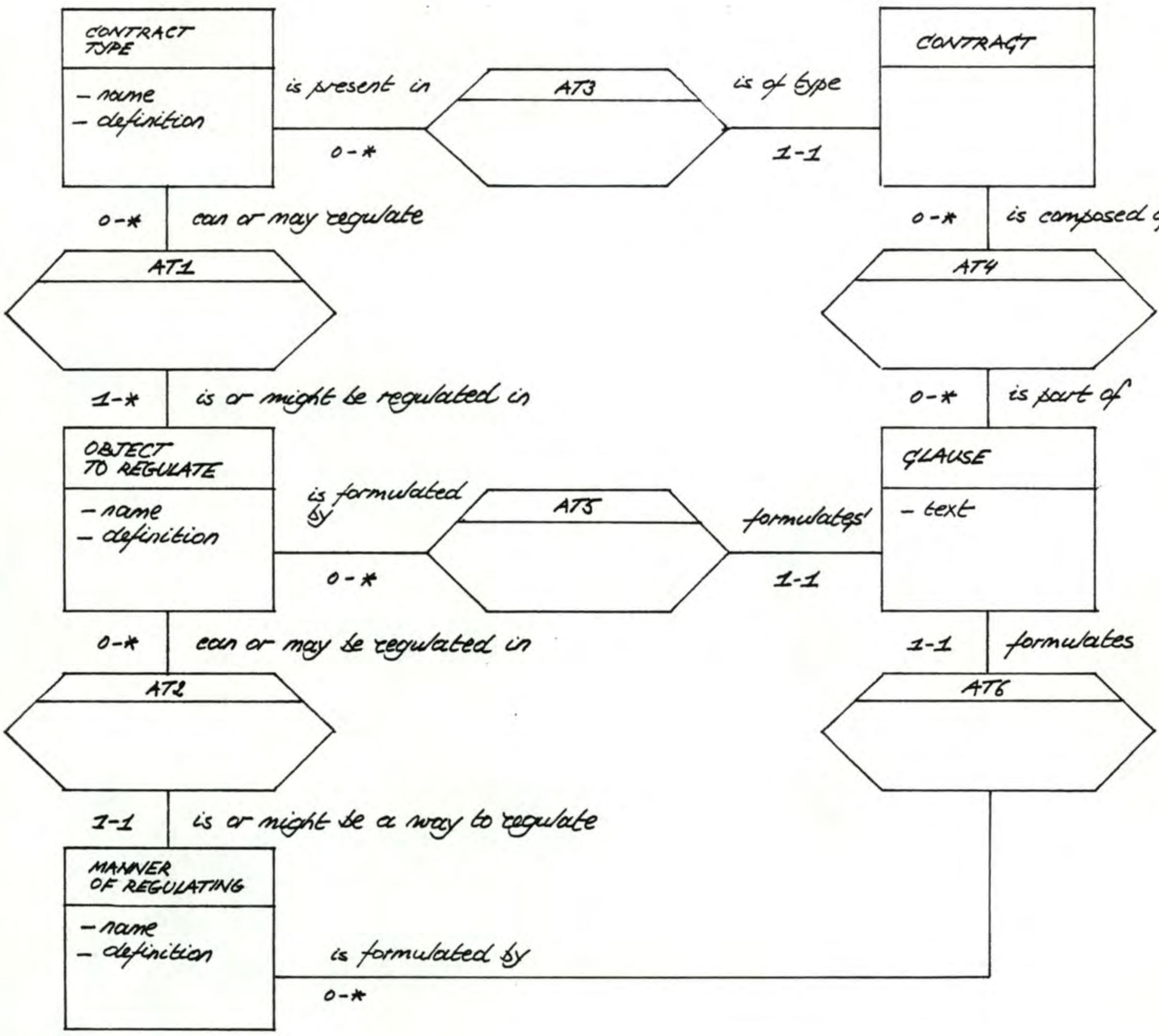


Figure 4-3: CONCEPTUAL SCHEMA NUMBER 2

Therefore we introduce the concept of scale of manners. Instead of viewing the set of manners of regulating associated to an object as an unordered set, we could define an order upon the manners. The order we need is in terms of degree of protection of the interests of each contractor.

If in a contract type T confronting contractor A and B, the object can be regulated by {M1, M2, M3} the scale of manners would be

contractor A M1 M2 M3 contractor B

if M1 is the manner which provides the strongest protection for contractor A and the weakest protection for B

if M3 is the manner which provides the weakest protection for contractor A and the strongest protection for B

if M2 is the manner providing to A a stronger protection than manner M3 and a weaker protection than M1

The introduction of the scale of manners permits to propose manners of regulating that may reconcile contractors when their contracts diverge for an object. If contractor A wants a clause that regulates the object O in manner M1, while contractor B prefers a clause that regulates the object O in manner M3, it's possible to think of clauses associated to M2 as candidates for reconciliation.

A listing providing reconciling clauses for the divergent contracts of DIGITAL and his cocontractor is given in appendix I.6.

4.2.5. The model referenced

The analysis of the process of contract writing pointed out that the state of the law environnement is described by different documents such as laws, Court decisions, articles and books. The contract writer should have access to such documents or at least be capable of managing references to such documents.

The system permits to manage pointers to such documents. The pointers may be associated to contract types, objects to regulate, manners of regulating and clauses.

4.2.6. The model facing default law rules

When a contract is silent about some aspect of the relationship of the conctractors, that aspect may be ruled by a default law rule. The default law rule constitutes a particular manner of regulating some object of the contract type. As the contract is silent, analysis will discover an empty clause and the empty clause will give rise as any other clause to a triplet (object regulated, manner of regulating, clause). The builder of the model should insert the manner of regulating associated to the default law rule at the correct position in the scale of manners.

4.2.7. The model facing imperative law rules

When an imperative law rule is applicable to a certain aspect of the relationship of contractors, the parties are forced to accept the manner of regulating prescribed by the imperative law rule. In terms of the proposed modelling this will result in the creation of an object with one manner of regulating. As there is no freedom to overrule the imperative law rule the associated scale of manners will have size one.

4.2.8. Commonality of models

Some objects to regulate may be aspects of relationship of several contract types. For example the object 'Applicable law' is an aspect of the relationship of nearly all contract types. As another example the object 'Delivery time' constitues an object to be regulated in the contract types hardware sale and hardware rental. Therefore the system provides facilities to take over to a new model selected objects of already existing models.

4.2.9. The model as a tool for retrieval of clauses

The contract writer may have to retrieve clauses. The model can be viewed as a part of the acces way to clauses, which permits the user to specify his search progressively form the general to the more detailed. This is illustrated by figure 4-4.

The system admits broad retrievals (all clauses of a contract type), narrower

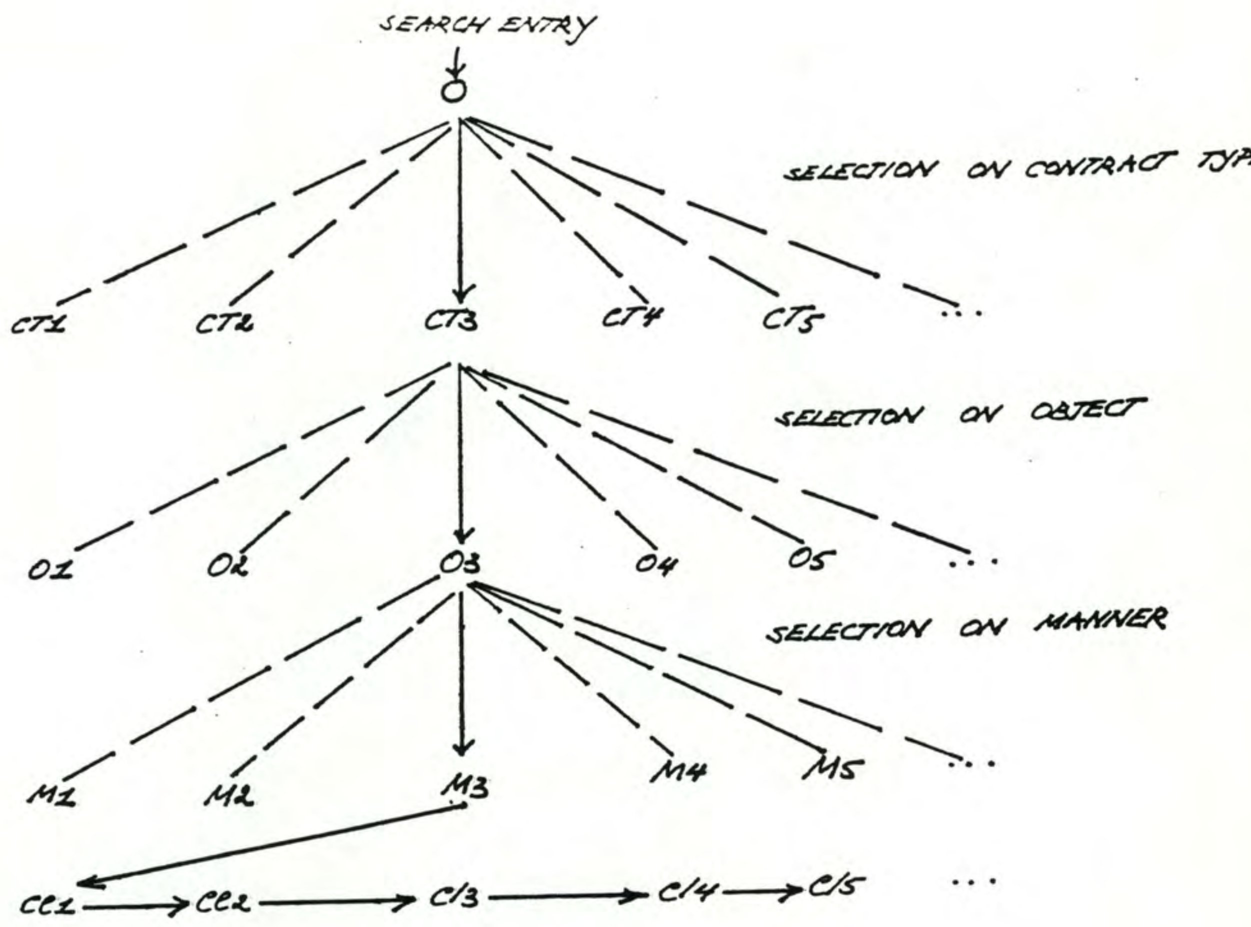


Figure 4-4: THE ACCESS WAY TO CONTRACT CLAUSES

retrievals (all clauses of an object of a contract type),and very narrow retrievals (all clauses that deal with an object of a contract type in a given manner). The retrievals can be made on screen and hard copies can be provided (appendix I.10.)

4.3. GUIDELINES FOR THE CREATION OF THE MODEL

1. compile documents describing the state of the law environment
2. compile model contracts, standard contracts, by the example contracts
3. compile known contract clauses
4. when reading 1 note possible objects to regulate and their manners of regulating

5. look if 2 - 3 provide additional objects to regulate or additionnal manners of regulating

6. determine the scale of manners for each object by comparing the discovered manners in terms of benefit for each contractor

7. discuss the model created with non-lawyers having specialised knowledge and experience in the field covered by the contract type

8. examine preexisting models to see if some objects of them can be transfered to the new model to create

9. determine for each object the applicable default rule. Formulate the manner of regulating carried out by the default law rule and insert that manner at the correct position on the scale of manners

10. if an imperative law rule applies create the object ruled imperatively and associate as manner of regulating the manner implied by the imperative law rule

11. Appreciate the created model in light of the quality criteria described in 4.5.

<u>4.4. THE DYNAMIC NATURE OF MODELLING</u>

The profond justification for distinguishing the meta-model from the models the user of the system can constitute, lies in the fact that this distinction softens the pressure the cognitive loop may exercise on the evolution loop. In essence the pressure of the cognitive loop is due to the fact the user of the system will acquire a progessively growing experience and knowledge, the state of the law environnement will evolve and the user may gain or loose interest in certain contract types. If the builder would have developed a system based on a fixed unchangeable model, the pressure of the cognitive loop would have required a builder which is constantly at the service of the user in order to adapt the fixed model or invent new fixed models. In this section we will precise the term dynamic nature of the model (4.4.1.), discuss the factors of evolution of the model which don't put pressure for evolution on the builder (4.4.2.),and give some indications about factors of evolution that might put pressure for evolution on the builder (4.4.3.)

4.4.1. The dynamics of a model defined

Speaking about the dynamics of a model implies that the model is observed at successive points in time. The model is at those successive observation still relative to the same contract type. As a model might be defined as relation M = (OBJECT, NAME-O, DEFINITION-O, MANNER , NAME-M, DEFINITION-M), we may compare models as relations. The model M1 existing at T1 may remain identical at T2 or may be different at T2. Let's denote by M2 the model M1 at time T2. If M1 = M2 the model hasn't changed. If M1 is different from M2 the difference may be due to one or more of following actions :

- adding a new object
- suppressing existing objects
- modifying the name of an object
- modifying the definition of an object
- adding a new manner
- suppressing an existing manner
- modifying the name of a manner
- modifying the definition of a manner

Notice that preceding precision of the term dynamics of a model, keeps invariant the underlying meta-model. Factors which claim for another meta-model are discussed in 4.4.3.

4.4.2. Factors of evolution of the model which don't put pressure on the builder

1. <u>The experience loop</u>

A contract writer gains experience through his daily practice, reading of articles, assistance to conferences. He may discover that an important aspect of the contractual relationship was not considered so far. So he will communicate that new object to the system. He may discover another useful manner of regulating which may reconcile contractual parties or protect more adequately their interests. So he will communicate that new manner to the system. Also he may recognise that his perception of some aspects of the relationship was confused or that some aspects of the relationship he considered should not be considered. So he will communicate the rearranged and suppressed objects to the system.

2. <u>The law change loop</u>

The state of the law environment applicable to a contract type may evolve. The legislator may enact new default rules, imperative law rules or change existing ones. Also jurisprundential doctrine may change the effects it traditionnally attributed to certain manners of regulating. If a new default rule is created or an old one changed, the user will communicate to the system the new manner and its place on the scale of manners. Changed jurisprudential doctrine may require a reorganisation of the manners of regulating associated to the object affected by the change.

3. <u>The analysis loop</u>

If the contract writer analyses existing contracts in terms of a given model M and he discovers in the analysed contract a clause which he cannot qualify in terms of objects and manners present in M, then it's probable the clause deals with an object not thought of before, or deals with an already perceived object but in a way not thought of before. Accordingly the contract writer will communicate to the system new objects and /or new manners.

4.4.3. Factors of evolution which put pressure on the builder

Factors of evolution which put pressure on the builder are those which make the user consider the actual system as imperfect and which require important intervention of the builder to make the system more perfect in the eyes of the user. These are the factors the system cannot handle by interaction between itself and the user. As the actual system is only realised in the form of a prototype and as such has not been criticised on a sufficient scale, we can hardly anticipate all evolution requirements precisely. However if the cognitive loop of the user would reach the point where ideas about a more powerful form of modelling come to raise, severe pressure will be exercised on the builder. The system is flexible in the sense that based on the meta-model, the models can evolve to face the experience loop, the law change loop and the analysis loop.

We recognize the system becomes unflexible if another meta-model should be integrated.

4.5. QUALITY CRITERIA FOR THE PROPOSED MODELLING

In this section we propose and justify some quality criteria for a model created under the system. The quality criteria provide at the same time some guidelines for the validation of a model, which complements 4.3.

CRITERION 1 : EXHAUSTIVENESS OF OBJECTS

The model created provides a set of objects O. Each object deals with a particular aspect of the contractual relationship. Let's name A the set of all aspects of the relationship that according to a "superior knowledge" should be dealt with in the contract type. Exhaustiveness of objects is present when $O = A$. On the other hand if $O \subset A$ some aspects of the relationship that - according to a "superior knowledge" - are important are not considered in the model.

CRITERION 2 : RELEVANCY OF OBJECTS

The model created provides a set of objects O. Each object deals with a particular aspect of the contractual relationship. Let's name A the set of all aspects of the relationship that according to a "superior knowledege" should be dealt with in the contract type. $O \cap A$ are relevant objects of the model. $O \setminus A$ are not relevant objects of the model. Objects in $O \setminus A$ deal with aspects which shouldn't enter in the contract's sphere.

CRITERION 3 : EXHAUSTIVENESS OF MANNERS

For each object the model may provide a set of manners M. Each manner rules the aspect in a particular way. Let's name W the set of all particular ways that according to a de "superior knowledge" may be useful to rule a given aspect of the relationship.

Exhaustivenees of manners is present when $M = W$. On the other hand if $M \subset W$, some particular ways that - according to a "superior knowledge" are useful are not considered in the model.

CRITERION 4 : RELEVANCY OF MANNERS

For each object the model may provide a set of manners M. Each manner rules the aspect in a particular way. Let's name W the set of all particular ways that according to a "superior knowledge" may be useful to rule a given aspect of the relationship. $M \cap W$ are relevant manners. $M \setminus W$ are manners which are not relevant. The manners in $M \setminus W$ are ways of regulating which can not be considered useful to rule the aspect of the relationship.

CRITERION 5 : DISJOINTNESS OF OBJECTS.

Two objects O1, O2, part of the same model, are said to be disjoint if they each concern a different aspect of the relationship. Objects of the model are not disjoint if they concern the same aspect. Disjointness of objects eliminates redundancy.

CRITERION 6 : DISJOINTNESS OF MANNERS.

Two manners M1, M2 asociated to an object are said to be disjoint if they each constitute a particular way of regulating the object. Manners of the model are not disjoint if they constitute the same way of regulating the object. Disjointness of manners eliminates redundancy.

CRITERION 7 : SUGGESTIVE NAMING

The model requires the objects and the manners to be named. Names are suggestive when the link to the corresponding concepts is easily made. Examples of suggestive naming of objects are 'delivery daté, 'system availability'. Non suggestive namings might be 'DD' 'SA' 'O1'...

CRITERION 8 : USEFUL DEFINITIONS

The model requires the objects and the manners to be defined. The definitions of the object are useful when they describe clearly, precisely, and unambiguously the aspect of the relationship the object deals with. The definition of the manners are useful when they describe clearly, precisely and unambigously the specificity of each way of regulating.

CRITERION 9 : CORRECT SCALING OF MANNERS

The manners of regulating an objects of the model should be ordered according to the degree of protection they offer to each cocontractor. Among all possible orders only one will provide correct scaling.

4.6. ECONOMIC CONSIDERATIONS ABOUT THE PROPOSED MODELLING

Stating a money amount for the cost of creation of a model is difficult. It depends on the contract type, the user's knowledge of the law environment and of the economic operation. Also consideration should be given to the quality of the created model. If the model would be nothing less nothing more than another form - an explicit and formalised one - of the existing knowledge of a contract writer, the cost of modelling would be essentialy the time during which the human capital of the contract writer is consumed.

Stating a money amount for the cost of analysing an existing contract in terms of a good quality model is easier. It essentially depends on the contract type and the familiarity of the analyst with the model. Contract types may have a varying number of objects to be regulated, the analyst may or may not have constituted the model himself; the analyst may or may not have made analysis for the same contract type before.

We made the analysis of the DIGITAL contract (appendix I.2.),in terms of the model (appendix I.1.),in about 3 hours. Entering the analysed contract into the system took about 1h30.

The cost of creation can be reduced by centralisation of the creation of models. An organisation could develop a high quality model and propose it to the community of contract writers. Each contract writer would be able to personalise the proposed model.

The cost of analysing contracts may be limited due to the fact that on certain markets a small number of actors represent the quasi-totality of the market. If actor 1 proposing contract C1 represents 50% of the market, actor 2 proposing contract C2 represents 15% of the market and actor 3 proposing contract C3 represents another 15%, then we can state that after having analysed C1, C2 and C3 the probability the contract writer has to analyse another contract is 0.20.

CHAPTER 5

A DATA BASE OF CONTRACT CLAUSES

Requirement R21 stated that the system to develop should be able to represent an impartial knowledge of clauses usable in a contract type. If we have a good model of a contract type (exhaustive objects, relevant objects, exhaustive manners, relevant manners, disjoint objects, disjoint manners, objects and manners all having received suggestive names and useful definitions) and obtain for each manner of an object at least one clause we think to have satisfied the requirement.

The clauses may be clauses of existing contracts (model contracts, standard contracts, by the example contracts), may be clauses from doctrinal treaties or compilations of clauses, or may be imagined by the contract writer himself.

As the clauses are qualified by the object to which they are relative and by the manner of regulating they carry out, the system has a grasp on the semantics of the clause.

As different formulations are possible to rule an object in a specific manner, we should enable the contract writer to associate several clauses to a manner of an object.

The functions provided by the system to manage the data base of clauses are :

- entry of a clause
- suppression of a clause
- modification of a clause
- list all clauses which regulate an object
- list all clauses which regulate an object in a specified manner
- print all clauses which regulate an object
- print all clauses which regulate an object in a specified manner

As some documents may describe the state of the law environment applicable on clauses, the system permits to associate to a clause one or more references pointing to these documents. Functions are provided to manage the references on clauses (entry, modification, suppression, consultation).

CHAPTER 6

THE LINK BETWEEN FACTS AND CLAUSES

6.1. THE MAPPING FUNCTION IN GENERAL

6.1.1. Statement of the problem

Our analysis of existing solution types pointed out that in some solution types (doctrinal treaty, compilation of clauses), indications were given about situations which claim for application of certain clauses. Also the analysis of the cognitive process of the contract writer showed the presence of a basic reasoning 'very useful clause when', be it under varying names and to various degrees of refinement.

As far we have precised our solution to the problem of contract writing in terms of a model and a data base. If the model respects the quality criteria enounced in 4.5 it is capable of representing some part of the knowledge about a contract type (objects, manners, references) It also permits to represent existing contracts (standard contracts, model contracts) in a form enabling the system to have a grasp on their semantics. Beside the qualified clauses of existing contracts, the solution proposed a database of clauses which may be imagined. This database represents another part of knowledge about the contract type.

The solution so far exposed provides assistance in comparison of contracts, provides clauses that reconcile divergent contracts, and information retrieval facilities. However it provides no assistance in the creation of a contract adapted to the particular situation of a contractor. The problem to solve is may be illustrated by figure 6-1.

Given

- a database covering all clauses usable in a contract type,
- the system has some grasp on the semantics of each clause,
- a candidate contractor in a particular situation,

how can we select from the database of clauses those clauses that will be

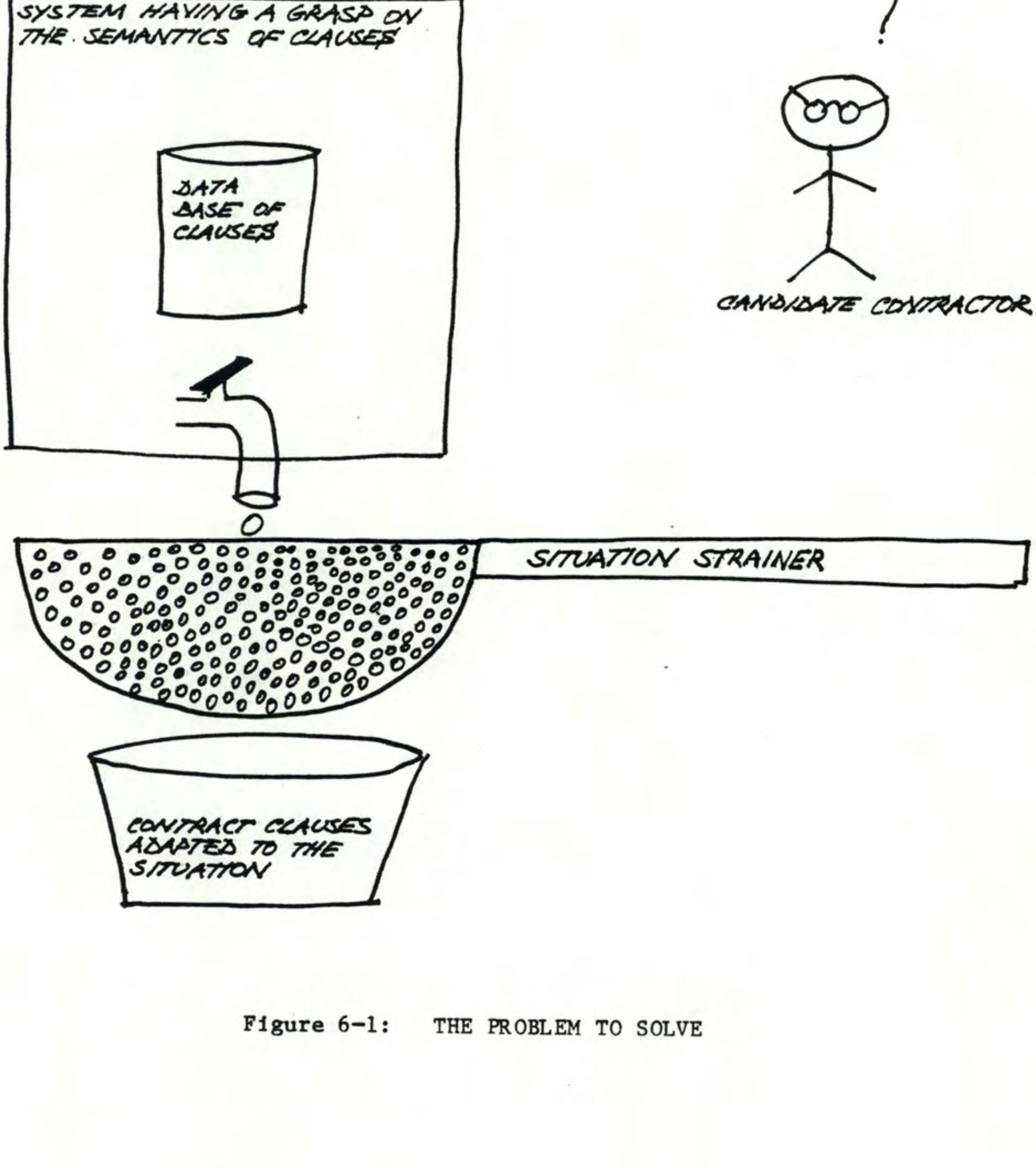

SYSTEM HAVING A GRASP ON THE SEMANTICS OF CLAUSES
DATA BASE OF CLAUSES
?
CANDIDATE CONTRACTOR
SITUATION STRAINER
CONTRACT CLAUSES ADAPTED TO THE SITUATION

Figure 6-1: THE PROBLEM TO SOLVE

adapted to the particular situation of the candidate contractor?

6.1.2. The relation 'requires a recourse to'

Consider the set, named M, of all manners of all objects provided in a good model relative to a contract type T.Consider also the set, named F, of all facts caracterising situations of possible contractors of contract type

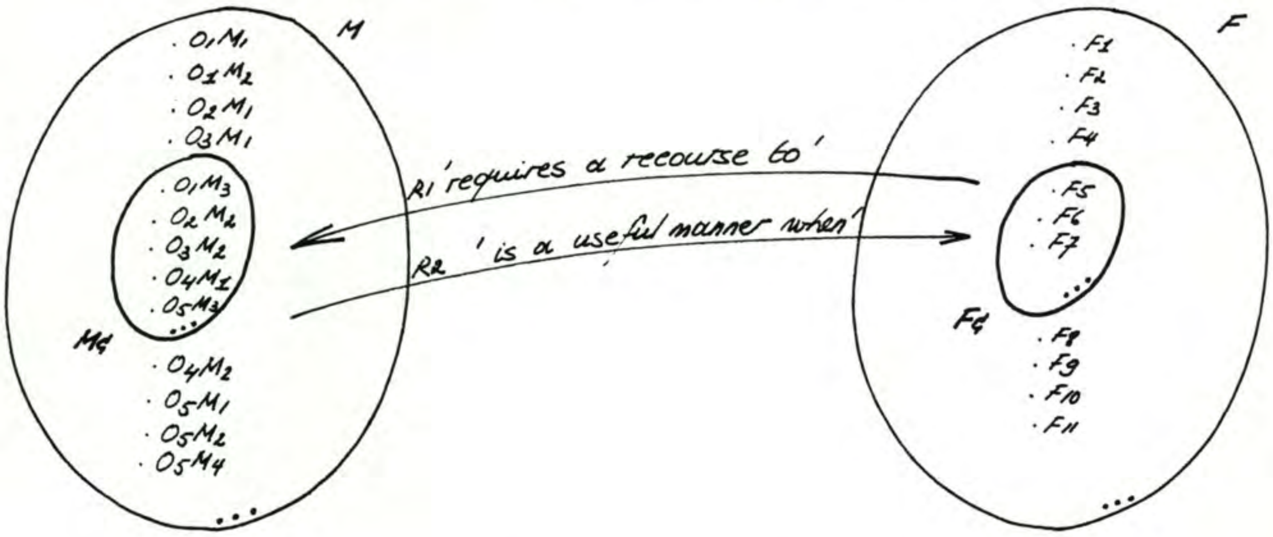


Figure 6-2: THE RELATION 'REQUIRES A RECOURSE TO'

Consider from F to M the relation R1 of which the definition is Fj R1 OkMl, when the presence of fact Fj requires a recourse to manner Ml of regulating object Ok. The corresponding inverse relation R2 can be interpreted as Ml is a useful manner of regulating object Ok when the fact Fj caracterises an aspect of the situation of the contractor. A contract writer having "supreme knowledge" of a contract type will be able to define M, F and R1. Applying his knowledge on a particular case he will view the situation of a contractor C as characterised by a subset Fc of F and the contract adapted to the situation of a client as one of which the semantics should be R1(Fc) = Mc.

Integrating this part of knowledge of the contract writer in to the system requires the system to be able to

- represent F
- represent M
- represent R1

- represent Fc
- represent Mc
- calculate Mc from Fc

6.2. THE PROPOSED MAPPING FUNCTION

6.2.1. General outline

As the candidate contractor is the only person who knows his particular situation, we got the idea that an efficient way for the contract writer to obtain knowledge of that situation would be through a questionnaire. For each contract type there would be one questionnaire. The questionnaire is composed of a series of questions. For each question a number of possible answers are given. The candidate contractor is asked to indicate the answer which seems most appropriate to his situation. By imposing a bijective relationship between the objects of the model and the questions of the questionnaire and a bijective relationship between the set of manners of each object and the set of answers of the corresponding question, it becomes possible to determine on base of the answers provided by the candidate contractor the needed semantics for an adapted contract. Knowing the needed semantics by the same qualification mechanism as the one by which the data base of contract clauses is structured, it is possible to retain the clauses which will provide the needed semantics. As the candidate contractor may have wrongly answered the questionnaire and the used mapping function may not be perfect, the system provides facilities to modify the automatically generated and adapted contract.

6.2.2. A questionnaire as a representation for F.

In 6.1.2. we defined F as the set of all facts characterising situations of possible contractors for a given contract type. It is possible to discover in F subsets SF1, SF2, SF3 regrouping facts which are to be considered when ruling aspects of the relationships. Let's name SF1 the set of the facts to be considered when ruling the aspect 'assistance of the maintenance firm in case of displacement of configuration'. That set SF1 may contain following facts characterising the situation of a possible contractor.

Fact 1 the contractor will certainly displace

Fact 2	the contractor will certainly not displace
Fact 3	the contractor may displace but is not sure
Fact 4	the contractor has employees that can displace the configuration provided the maintenance firm gives directives
Fact 5	the contractor has no employees that can displace the configuration

The following question and answers permit to identify the particular situation of a contractor.

Question "During the maintenance contrat, it is possible that the configuration may be displaced wholly or partially. On what support of the maintenance firm do you want to count ?"

Answer 1:	I need no support. I'm 100% sure there will be no displacement
Answer 2:	Displacement is sure or highly probable. As I dispose of competent personnel, supervision of the maintenance firm will be sufficient support.
Answer 3:	Displacement is sure or highly probable. As I don't dispose of competent personnel, I prefer a more extensive support.

The answers 'incorporaté the facts to be considered when ruling the aspect 'assistance of the maintenance firm in case of displacement of configuration'. The questionnaire is complete when each aspect of the relationship is questionned and for each question the answers incorporate the facts to be considered. The questionnaire associated to the maintenance contract for sold EDP-hardware can be found in appendix I.7.

6.2.3. A model as representation of M

In 6.1.2. we defined M as the set of all manners of all objects provided in a good model. We only remind that the semantics of an adapted contract can be expressed in terms of the model and refer back to our development on contractual modelling in chapter 4.

6.2.4. The link between answers and manners as representation of Rl

In 6.1.2. we introduced the relation Rl from F to M. Fj Rl OkMl meant that the presence of the fact Fj required the recourse to manner Ml of regulating object Ok. As the answers incorporate facts to be considered, we propose to represent Rl in our system as a relation T between the set A of answers and M. That relation T from A to M is defined as Al T OkMl when the answer Al given by a candidate contractor triggers usefully the manner Ml of regulating object Ok.

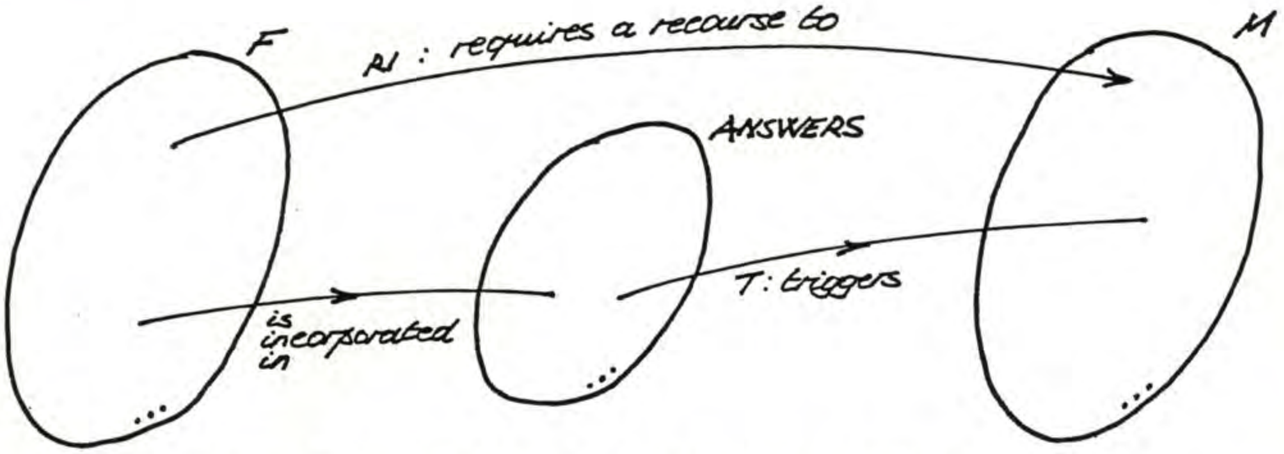


Figure 6-3: THE LINK BETWEEN ANSWERS AND MANNERS

6.2.5. The answered questionnaire as a representation of Fc

In 6.1.2. we defined Fc as a subset of F containing the facts characterising the situation of a given contractor C. The set of answers retained by the candidate contractor determines via the inverse of the relation 'is incorporated in' a subset of F.

6.2.6. The relation between the model and the questionnaire

We have discovered questions about aspects of relationship, manners that incorporate facts and a triggering relation between the answers and the manners. If these are the basic construction blocks of the proposed mapping function, nothing is said about the connectivity between questions and aspects, between answers and manners. The most general way would be by accepting many to many connectivities. However managing such connectivities would have increased the complexity of the system and may have confused the user of the system. Therefore we propose a bijective relationship between the set of the questions and the set of objects of the model. This forces the user to question each aspect of the contractual relationship. The proposed triggering relation is also a bijective relation between the set of answers of a question and the set of manners of the corresponding object. This forces the user to incorporate in each answer those facts of the situation of candidate contractors that trigger a manner of regulating. Appendix I.8. contains the mapping function for maintenance contracts of sold EDP hardware.

6.2.7. A subset of the model as representation of Mc

Applying the bijective relationship on the answers provided by a candidate contractor, we obtain a subset of the model. That subset has the property that each manner is only relative to one object and is a representation of the semantics to be contained in the adapted contract. Appendix I.9. contains the mapping function applied to a particular case.

6.2.8. The selection of clauses of the adapted contract

So far we have precised a mechanism to derive from the answered questionnaire the semantics required for an adapted contract. Also we dispose of a data base of clauses of which the semantics are known by the system through qualification. We insist that the semantics required for an adapted contact and the semantics of a clause are represented in the same form : in terms of objects regulated and manners of regulating. This common form of representation of semantics permits to filter out of the data base those clauses that will realise the semantics required in an adapted contract. The adapted contract will be made up of as many clauses as there are objects in the model. Each clause will formulate a manner of regulating, element of the subset described in 6.2.7. As the database accepts to memorise many clauses which formulate a specific manner of regulating an object, and by definition each of those clauses has the same semantics, we could choose any of them to be part of the adapted contract. However only one clause can be put in the adapted contract. Therefore we introduce the concept of 'privileged clausé. A privileged clause is the unique clause - among all clauses which formulate a specific manner of regulating an object - which will systematically be retained to be part of adapted contracts.

6.2.9. Human controlled automatic generation.

As the candidate contractor may provide wrong answers and the mapping function may be imperfect, the contract writer will have to discuss the automatically generated adapted contract with the candidate contractor. From the discussion may appear that for certain objects other manners of regulating would be more appropriate, or that other clauses than the privileged ones would provide a better formulation. The system provides functions to insure these modifications.

6.2.10. Basic data structure for the proposed mapping function and occurrences

The conceptual schema of figure 6-4 represents the basic data structure for the proposed mapping function.

We believe that this basic structure enables to represent the varying mapping functions in the mind of contract writers as occurences of that basic structure. The analysis of cognitive process of the contract writer showed properties as specialisation, individualisation and evolution. Because of the property of specialisation, the system admits the contrat writer to develop a questionnaire only for those contract types he is interested in. The property of individualisation is supported by the fact each contract writer has complete freedom to formulate questions and answers which incorporate those facts which the individual contract writer feels important. The property of evolution is supported by the fact a given questionnaire can be extended to fit a refined model, by the possibility to reformulate questions and answers as additional or other facts should be considered.

6.2.11. The system functions realising the mapping mechanism

Following list resumes the functions by which the system manages the mapping mechanism.

- declaration of an adapted contract : the user communicates the attributes of the adapted contract
- entry of questions
- entry of answers
- entry of links between answers and manners

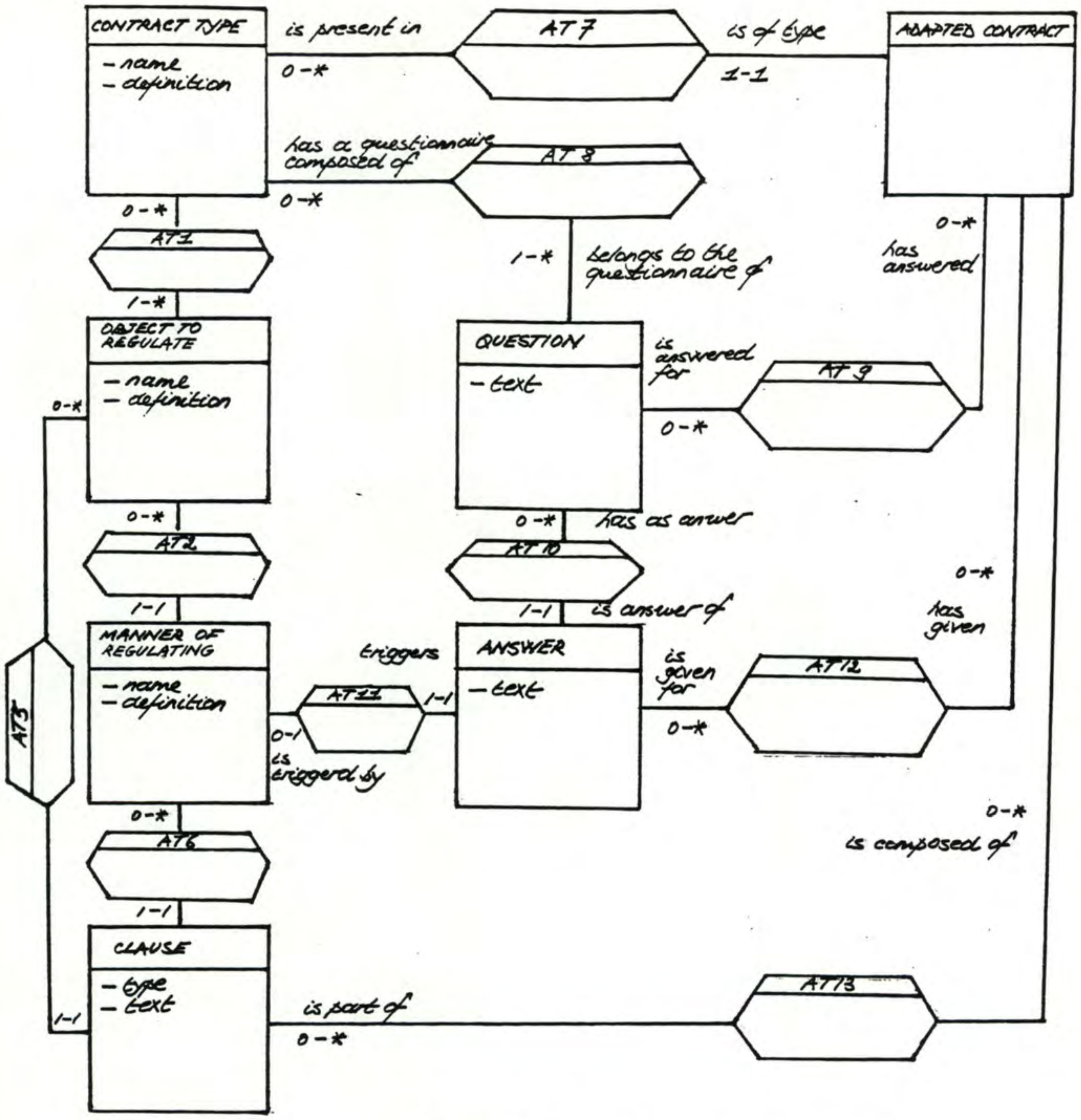


Figure 6-4: CONCEPTUAL SCHEMA NUMBER 3

- entry of privileged clauses
- modification of questions
- modification of answers
- modification of the links between answers and manners
- modification of privileged clause
- suppression of questions
- suppression of answers
- suppression of links between answers and manners
- suppression of privileged clauses
- consult the mapping function
- print the mapping function
- print the questionnaire
- entry of answers given by a candidate contractor
- modification of the entered anwers given by a candidate contractor
- generate the adapted contract (calculate from the answers given the clauses which make up the adapted contract)
- consult the adapted contract on screen
- print the adapted contract
- modify the adapted contract
- suppress the adapted contract

6.3. GUIDELINES FOR THE CREATION OF THE MAPPING FUNCTION

1. Develop a good model.
2. for each object of the model formulate a question
3. for each manner of the model think about facts that could require recourse to the manner. Incorporate those facts in an answer.
4. Read documents describing the state of the law environnement (doctrinal treaties, compilation of clauses) and filter their indications of when to use what clause
5. Discuss the mapping function created with non lawyers having specialised knowledge and experience in the field covered by the contract type.
6. Appreciate the created mapping function in light of the quality

criteria described in 6.5.

6.4. THE DYNAMIC NATURE OF THE MAPPING FUNCTION

In this section we precise the dynamic nature of the mapping function (6.4.1), describe the factors of change of the mapping function (6.4.2) and indicate the limits of the evolvability of the system (6.4.3).

6.4.1. The dynamics of a mapping function defined

The term dynamics can have two distinct meanings. The first meaning is relative to the creation of a complete mapping function : initially there exists only the model and a database of clauses, and after creation of the complete mapping function each object has its question and each manner has its answer. In a second meaning the dynamics refers to changes on a complete mapping function.

Speaking about the dynamics of a mapping function in its second meaning implies that the mapping function is observed at successive points in time. The mapping function is at those successive observation moments still relative to the same contract type. As the mapping function may be defined as a relation MF = (QUESTION, TEXT-QUESTION, ANSWER, TEXT-ANSWER, MANNER, CLAUSE, TEXT-CLAUSE),we may compare mapping functions as relations. The mapping function MF1 existing at T1 may remain identicial at T2 or may be different at T2. Let's denote by MF2 the mapping function MF1 at time T2 > T1. If MF1 = MF2 the mapping function has not changed. If MF1 is different from MF2, the difference may be due to one or more of following actions :

- modification of the text of a question,
- modification of the text of an answer,
- modification of the manner associated to the answer,
- modification of the privileged clause.

Notice that preceding precision of the terms dynamics of a mapping function, keeps invariant the basic structure of our mapping function. Factors which claim for other structures are discussed in 6.4.3

6.4.2. Factors of change for a mapping function

1. DEPENDENCY ON A EVOLUTIONARY MODEL

   In 4.4 we discussed the dynamic nature of the model and identified the experience loop, the law change loop and the analysis loop as factors of evolution of an existing model. Our mapping function - by imposing a bijective relationship between objects and questions and a same relationship between manners and answers - is dependent on the model. As such adding a new object in the model will imply the creation of a new question and adding a new manner will require the addition of a new answer. Modifications of the naming and or of definition of objects and manners may require redesign of corresponding question and answers.

2. THE INTERVIEW LOOP

   Candidate contractors may provide comments in the 'remark field' when they don't agree with the fixed set of possible answers proposed. This means that the questionnaire was not sensible to a fact which characterises the situation of possible contractors for the contract type. The contract writer may adapt the guilty question and answers. Eventually the model will have to be refined.

3. THE DISCUSSION LOOP

   The generated adapted contract will be discussed between the contract writer and the candidate contractor. This practice may help to discover anomalies of the mapping function especially on the link between the answers given and the clauses selected.

4. THE EXPERIENCE LOOP

   Beside the experience gained from the interview and discussion loops, the contract writer may through reading of books, articles, assistance to conferences, discussion with specialists in the technical domain, ameliorate his perception of the facts that should be considered when writing contracts of a given type. He will therefore incorporate newly discovered facts in the answers, change the text of the privileged clauses,reformulate the questions. Eventually also the model will have to be refined.

6.4.3. Limits on the evolvability of the mapping function

The factors of change mentionned in 6.4.2 are factors the can be managed by interaction between the system and the user. The basic structure of our mapping function permits the user to express and modify freely occurrences of that basic structure and therefore reduces pressure on the builder. However some factors could exercise severe pressure on the builder. It will be those that take under attack the basic structure of our mapping function. At this moment we can hardly anticipate all possible evolution requirements: the system is prototyped but not largely criticised. Conscient that the basic structure of the mapping function may change one day, we introduced two defensive mechanisms. First the software architecture isolates in specific modules the logic of the mapping function. Secondly other system functions - comparison of contracts, reconciliation of contracts - should make no hypothesis on how an adapted contract is obtained.

6.5. QUALITY CRITERIA FOR THE MAPPING FUNCTION

In this section we propose and justify some quality criteria for the mapping function created under the system. These quality criteria provide additional guidelines for the validation of a mapping function, complementing 6.3.

CRITERION 1 : QUALITY Of THE ASSOCIATED MODEL.

As there is a bijective relationship between objects and questions on the one hand and between manners and answers on the other hand, the quality criteria of the model could give rise to parallel quality criteria for the questionnaire. Therefore a bad model implies a bad questionnaire. A good model is a necessary condition for a good questionnaire but not a sufficient one.

CRITERION 2 : UNDERSTANDABLE QUESTION AND ANSWERS

As the questionnaire is to be answered by a candidate contractor, who is not necessarily a specialist in law nor in the economic operation to be carried out, the formulation of questions and answers should be in terms meaningful to the candidate contractor.

CRITERION 3 : DISTINCTIVENESS Of ANSWERS

The candidate contractor should find easily the answer adapted to his situation. Therefore it must be easy to distinguish among the possible answers.

CRITERION 4 : EXHAUSTIVENESS OF FACTS IN AN ANSWER

The answer associated to a manner must indicate all the facts that may trigger that manner of regulating. If some facts are not considered a candidate contractor may not find back his situation in the answers proposed and the system will not take care of those facts.

CRITERION 5 : CORRECT LINK BETWEEN ANSWERS AND MANNERS

Suppose we have a good model in which object O1 is regulated by the scale of manners {M1, M2, M3}. Suppose also we have a corresponding question Q and a set of answers A = {A1, A2, A3}. Between {M1, M2, M3} and {A1, A2, A3} 6 distinct bijections are possible. Only one bijection will be the correct one. The correct one is the one which respects the relation usefully triggers as defined at 6.2.4.

Two pratical tests may certify the quality of the mapping function.

The first test consists in observing the number of remark fields the candidate contractor fills in when answering the questionnaire. A filled remark field is a sign of possible violation of criteria 2, 3 and 4.

A second test can be made when discussing the generated contract with the candidate contractor. The discussion may lead to the conclusion either that the generated contract is good or needs to be modified. The mapping function is good when no or slight modifications are to be made. The intensity of modification can be measured in terms of number of objects concerned and distance on the scale of manners between the generated manners and the manners adopted after discussion.

6.6. ECONOMIC CONSIDERATIONS ABOUT THE PROPOSED MAPPING FUNCTION

Stating a money amount for the cost of creation of a mapping function is not easy. The cost depends on the contract type, on the user's knowledge of the operation to be carried out and also on the quality of the created mapping function.

More precise indications can be given for the use of the created mapping function. Printing out the questionnaire associated to the contract type 'maintenance contract for sold EDP - hardware'will take a few minutes. Answering the 44 question questionnaire may take between 1h30 and 2 hours. Communicating the answered questionnaire to the system will take no more than 15 minutes. The generation of the adapted contract is a matter of seconds. Printing out the adapted contract takes a few minutes.

CHAPTER 7

THE TOTAL SYSTEM AND ITS SUBSYSTEMS

So far we have discovered a basic system composed of data and functions to operate on those data. This only provides a crude analysis of the proposed solution. During functional analysis, several problems were discovered, by examining the basic system from specific viewpoints. The first problem discovered was that the description of the basic system did not specify in what context what functions could be meaningful, neither information was provided how the system should react when a function is tried in an illegal context. Therefore we should expand the basic system by data and functions insuring the coherence of the basic system. This viewpoint requires the specification of a coherence subsystem. A second problem discovered was that the use of the basic system enhanced by a coherence subsystem will lead after a time to an important investment in terms of created models and mapping functions, analysed contracts, and data base of contractual clauses. This investment should be protected against unauthorised accesses. Also we discovered that not all users of the system should be authorised to execute all the system functions. For instance a secretary entering the answered questionnaire should not be authorised to delete a high quality model which required several months of development. Therefore we should expand the basic system by data and functions insuring the access security. This viewpoint requires the specification of a authorisation subsystem. A third problem discovered was that the basic system enhanced by a coherence and authorisation subsystem should incorporate defensive mechanisms against all possible incidents that may destroy wholly or partly the investment made. This viewpoint requires the specification of a security subsystem.

In following sections we will examine each of the subsystems needed to make from the basic system a total system providing coherence, access control and incident resistance.

7.1. THE COHERENCE SUBSYSTEM

7.1.1. The coherence problem stated

When observing a system we can define it in terms of data and functions operating on those data. As such we introduced in preceding chapters the data and functions needed for the proposed solution. An important property of a function we neglected so far is the context a function requires to be executed meaningfully. Following examples will illustrate previous statement. The function 'entry of an analysed contract' by which the user communicates the set of triplets (object, manner, clause) can only be executed when the system has previously memorised the model associated to the contract type of the analysed contract. The function 'detection of divergence between a proposed contract and an adapted contract' can only be executed when the system has previously memorised the completely analysed proposed contract and when the adapted contract is generated. The function 'entry of the answered questionnaire'is only meaningful when the system has previously memorised the complete questionnaire. Solving this problem implies not only that we precise for each function the needed context but also that we have some way to represent the actual context of the system. Morever the successful execution of a function possibly changes the context. Therefore we should also precise for each function to what extent its execution changes the context. Another important property of a function we neglected so far is the reaction of the system when a function is tried to be executed in an illegal context. Userfriendliness requires the user to be informed by a significant message, why the functions cannot be executed. Solving this problem implies that we precise for each function in what kind of illegal contexts what messages should inform the user.

7.1.2. The concepts needed for a solution

Let's consider the system as a finite state machine M . The system M may be represented by figure 7-1 :

The system M has a certain number of possible states. Let's name Q the set of states : $Q = \{qo, q1, q2, ..., qp\}$. The system before any execution is in the initial state qo and then repeatedly performs an execution cycle. On each cycle M 'looks at' an input x, an identification of a function to be executed and its current state q ; on the basis of these, it outputs something, say y, - which is either the normal output when the current state represents an acceptable context

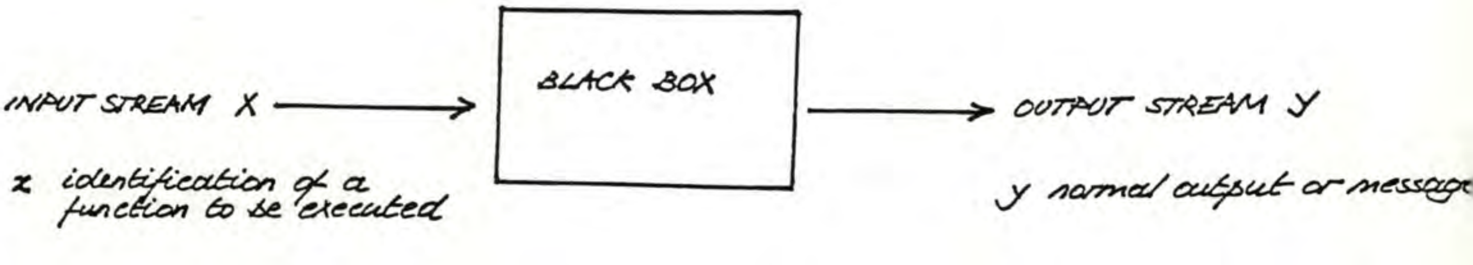

INPUT STREAM X
x identification of a function to be executed
BLACK BOX
OUTPUT STREAM Y
y normal output or messag

Figure 7-1: THE SYSTEM VIEWED AS A FINITE STATE MACHINE

for x or a message when the current state represents an illegal context for x - and switches to a new state, q' (it's acceptable to have q = q'). Both q' and y depend on x and q; that is we can characterise the output and the next state as functions of the current state and input:

```
y = OUTPUTFOR ( q, x)
q'= NEXTSTATEFOR (q, x)
```

We call NEXSTATEFOR the state transition function and OUTPUTFOR the output function.

The machine M is completely defined if we know Q, qo, X, Y, OUTPUTFOR, NEXTSTATEFOR. We know already X, the set of identifiers for all the functions of the basic system ,and a subset of Y, the output to be produced when the context of a function is acceptable (listings, information that should appear on the screen). But the complementary subset of Y composed of the messages to be produced when a function cannot be executed is still to be defined. Also we should discover O, qo and define OUTPUTFOR and NEXTSTATEFOR.

7.1.3. The discovery of system states.

We discovered that following aspects described the context required for functions to be executed meaningfully. Each aspect gives rise to a state indicator. Each state indicator will be named and its values defined.

1. the degree of development of the model for a contract type.

    state indicator: STATUTM

    values of the state indicator :

    0 = when the contract type is only named and defined and no object is associated to the contract type

    1 = when the contract type has only one associated object but the model of the contract type is not complete or is complete but has not been declared as complete.

    2 = when the contract type has only one associated object and has been declared complete with success.

    3 = when te contract type hase more than one associated object the model of the contract type is not complte or is complete but has not been declared as complete.

    4 = when te contract type has more than one associated object and has been declared complete with success.

   Note : a model is complete when at least one object is declared and each declared object has at least one manner.

2. the degree of development of the questionnaire for a contract type

    State indicator: STATUTQ

    values of the state indicator

    0 = when no question is associated to the contract type

    1 = when there exists at least one question and one answer of that question but not all questions and answers exists.

    2 = when for each object there exists a question and for each manner there exists an answer.

3. the degree of presence of privileged clauses of a contract type

    State indicator: STATUTC

    values of the state indicator

    0 = there exists no privileged clause

    1 = at least one manner of an object has received a privileged clause but there exist manners of an object that don't have a privileged clause

2 = each manner of each object has a privileged clause

4. the degree to which an object is questionned

State indicator: COMPLETELY-QUESTIONNED

values of the state indicator

0 = the object has neither a question neither answers

1 = the object has a question but no answers

2 = the object has a question and at least one of its manners has an answer but some manners have no answer

3 = the object has a question and each of its manners has an answer

5. the degre of presence of privileged clauses of an object of a contract type

State indicator: COMPLETELY-CLAUSED

values of the state indicator

0 = no privileged clause is associated to the object

1 = at least one manner of the object has a privileged clause but not all manners of the object have a privileged clause

2 = all manners of the object have a privileged clause

6. the number of manners associated to an object

State indicator: NUMBER-OF-MANNERS

Values of the state indicator: from 0 to 7

7. the fact wether a manner received an answer

State indicator: QUESTIONNED

Values of the state indicator

0 = when the manner has not received an answer

1 = when the manner has received an answer

8. the fact wether a manner has received a privileged clause

State indicator: CLAUSED

Values of the states indicator

0 = when the manner has not received a privileged clause

1 = when the manner has received a privileged clause

9. the degree to which an adapted contract is known by the system

State indicator: STATE

Values of the state indicator

0 = when the adapted contract is only identified

1 = when a least one answer is given but not all questions are answered

2 = when all questions are answered

3 = when the adapted contract is generated and needs no upgrading

4 = when the adapted contract has been modified after generation and needs no upgrading

5 = when the adapted contrat has been modified after generation or is generated and needs an upgrading

Note : a contract needs upgrading when one of its clauses has been deleted or has lost its qualification.

10. the degree to which a proposed contract is known by the system

State indicator: IND-ENREGISTERED-PC

Values of the state indicator

0 = when the proposed contract is only identified

1 = when a least one clause of the proposed contract is knows by the system but not all objects have received a clause of the proposed contract and the contract needs no upgrading

2 = when each object has received a clause of the proposed contract and the contract needs no upgrading

3 = when the proposed contract needs upgrading

11. the degree to which a reconciled contract is known by the system

State indicator: IND-ENREGISTERED-RC

Values of the state indicator

0 = when the reconciled contract is only identified

1 = when at least one clause of the reconciled contract is known but not all object have received a clause of the reconciled contract and the contract needs no upgrading

2 = when each object has received a clause of the reconciled contract and the contract needs no upgrading

3 = when each object has received a clause of the reconciled contract and the contract needs upgrading

7.1.4. The practical difficulties of defining the system as a FSM

The system state is caracterised by the values of each of the eleven state indicators defined in preceding paragraph. This would lead to a system in which $\#Q$ = 1.382.400 = 5 * 3 * 3 * 4 * 3 * 8 * 2 * 2 * 5 * 4 * 4 . The discovery of this number inspired fear and acted for a while as a nightmare on the author's sleep. Several findings softened our initial fear and desolation.

First defining precisely the system as a finite state machine in a limited time could only be done if $\#Q$ did represent a 'reasonable' number. Even if we could bring $\#Q$ down to let say 30, having 25 distinct basic fucntions would still have needed filling in 750 cases in the table defining the state transition function and another 750 cases in the table defining the output function.The eureka consisted in observing that often the decision to consider which states of the system constituted acceptable contexts and which states constituted illegal contexts for a function was based not on a complete observation of all state indicators but only on some of the state indicators. As such not all state indicators where relevant for each function.

Secondly the big number of 1.382.400 can be explained by the observation that the eleven state indicators were not completely independent. For instance the fact there exist an object that has COMPLETELY-QUESTIONNED = 3 implies that STATUTQ of the corresponding contract type = 1 or 2. Also the fact that STATUTQ = 0 for a contract type implies that each of its associated objects has COMPLETELY-QUESTIONNED = 0 . As there is redundancy we would have discovered impossible states and thereby reduced the number of possible states. The redundancy introduced however is not harmful because it permits a fast checking of the actual state. Suppose we eliminate the redundancy between STATUTQ and

COMPLETELY-QUESTIONNED. As the value of STATUTO could be derived from the value of COMPLETELY-QUESTIONNED of each of the associated objects of the contract type, we could drop STATUTQ as a state indicator and thereby reducing $\#Q$ to 1.382.400 : 3 = 486.800 enabling a "faster" definition of the system as a finite state machine. But on the other hand each time we did need to know the value of STATUTO we would have been forced to access the state indicators COMPLETELY-QUESTIONNED of each of the objects associated to the contract type.

Thirdly we found out that when a context was illegal, the next state was identical to the previous state. Therefore the table representing the state transition function and the table representing the output function can be combined in a unique table of which a case either contains the next state when the actual state is acceptable either an identification of the message to be displayed when the actual state is illegal.

7.1.5. A more practical way to describe the coherence subsystem

Reconforted by preceding observations we precised the state transition function and the output function for each state indicator individually and combined in one table both the state transition function and the output function. The result of this work was a table with two entries : the horizontal entry covers the functions, the vertical entry covers the different values of each state indicator. A case of the table either indicates the next state - if the actual state was thought acceptable- or a reference to a message to be send - if the actual state was thought illegal - . As during the filling we discovered that an acceptable state could give rise for a same function to different next states depending on certain conditions we might either have refined the set of defined states or maintain the existing set of defined states provided we give a clear description of such situation. For pragmatic reasons we prefered to sacrify the beaty of formalism and introduced a loose concept of 'conditional state change'. The table, the list of messages and the list of conditions can be found in Appendix II.

7.1.6. The concept of policy for the coherence subsystem

The policy can be defined as the major objectives to be obtained under the coherence subsystem. A policy determines what are acceptable states and what are illegal states for each function. Different policies could be considered for the coherence subsystem.

POLICY 1: ENABLING A MAXIMUM OF FUNCTIONS TO BE CARRIED OUT

This policy would imply that acceptable states correspond to the strictly required conditions for a function to be meaningfully executed. Illegal states are the states in which the stricly required conditions are not present.

POLICY 2: IMPOSING A GIVEN WORK METHOD ON THE USER OF THE SYSTEM

This policy would imply that acceptable states are those which respect the given work method. Illegal states are those which don't respect the given work method.

Following example may illustrate the concept of policy . The function "entry of a privileged clause" has as strictly required conditions the existence of an object and a manner. Policy 1 admits the user to enter a privileged clause as soon as the object and the manner are declared. Opposed to policy 1, policy 2 will impose a work method obliging the user to define completely the model before starting to provide privileged clauses. Therefore in policy 2 the acceptable states will be different from the acceptable states in policy 1.

The policy adopted in our coherence subsystem is a compromise between policy 1 and policy 2. The work method we try to impose on the user consists in an ordered succession of operations. First the complete model for a contract type should be communicated, then the privileged clauses may be entered and finally the mapping function should be communicated.

7.1.7. The basic structure of functions

In the light of the defined coherence subsystem it becomes natural to adopt for each function the basic structure, pictured in figure 7-2.

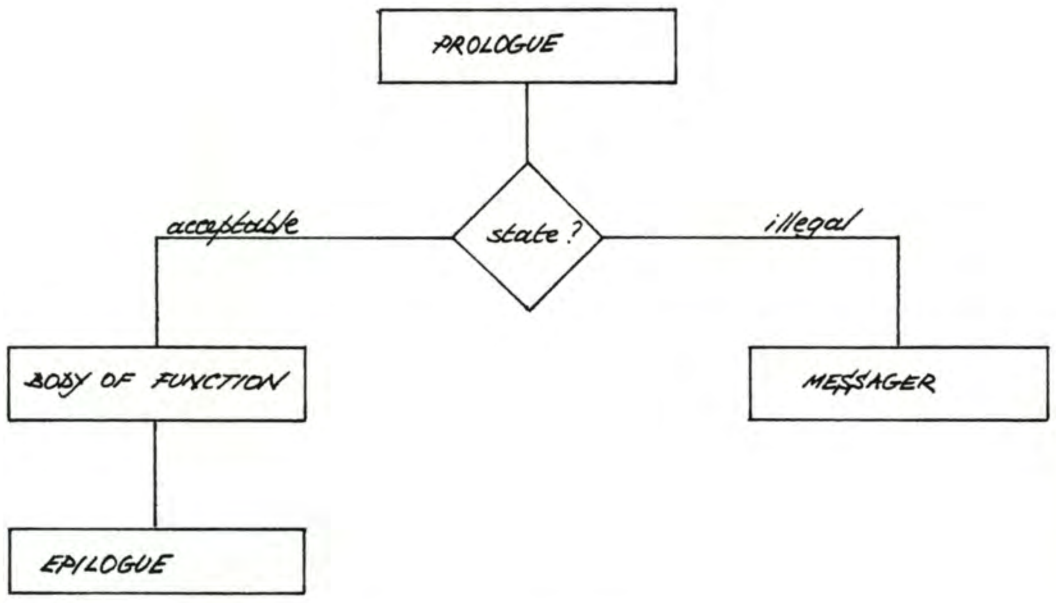


Figure 7-2: THE BASIC STRUCTURE OF A FUNCTION

The prologue consults the appropriate state indicators in order to determine if they constitute an acceptable or illegal state. The body of function executes the 'net' function. The epilogue modifies the appropriate state indicators to reflect the new system state. The messager determines and sends the appropriate message.

7.2. THE AUTHORISATION SUBSYSTEM

7.2.1. The authorisation problem stated

After a period of use the system will integrate an important part of the knowledge of a contract writer. The system will memorise models, mapping functions, analysed contracts, adapted contracts, reconciled contracts and clauses. All this may constitute a considerable investment that should be protected against undesired modification or destruction. Therefore two protections should be at least provided. First, people not belonging to the organisation of the contract writer should have no access to the system. Second, the people belonging to the organisation should only be enabled to use those functions they need.

7.2.2. Protection against unauthorised access.

The classical technique of password protection is both simple and well known. Each authorised user receives a password, memorised by the system. Upon each entry into the system the user is forced to communicate his password. The system checks the password entered and authorises or denies access. The system allows only a limited number of retrials when an entered password is wrong.

7.2.3. Protection against unauthorised use

Each user should not be enabled to use all system functions. Among the system functions there are functions which cannot destroy the investment made (consultation, print out) but also functions with can to varying degrees destroy the investment made (delete a contract type, delete a mapping function, delete an analysed contract, delete an object, delete a manner, delete a question, delete answers, delete clauses, delete references). Therefore we propose following outline for a solution. The use authorisation subsystem will contain a declaration mechanism, an identification mechanism and an enforcement mechanism. The declaration mechanism permits the system responsable to declare the users and their power in terms of the system functions they may execute. The declaration mechanism provides facilities for entry, modification, deletion and consultation. The identification mechanism forces each user to communicate his password (confer 7.2.2.). The enforcement mechanism is activated upon each tentative to execute a function.The enforcement mechanism checks if the identified user has the power to execute the function - by consulting the memory managed by the declaration mechanism - and either accepts the function to be executed or communicates its denial by displaying an appropriate message.

7.2.4. The basic structure of functions revisited

The basic structure of functions described under 7.1.7. is incorporated in FIGURE 7-3. This is because coherence checking needs only be done when a user is authorised to execute a function. The authorisation prologue checks if a user is intitled to execute a function. The refusal messager determines and sends the

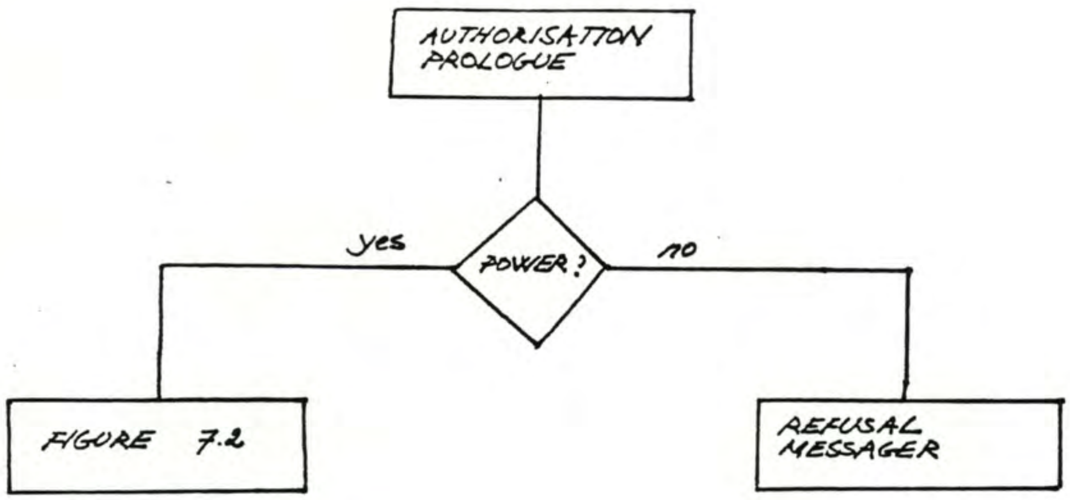


Figure 7-3: THE BASIC STRUCTURE OF A FUNCTION REVISITED

appropriate denial message

7.3. THE SECURITY SUBSYSTEM

Different incidents may destroy the reliable operation of the system. If no appropriate defensive mechanisms are incorporated, the investment made may be lost to various degrees and more importantly will ruin the confidence of the user. The incidents may go from disk head crashes to power failure. This aspect of the system is not studied by the author and constitutes an area for additional development of the present work.

CHAPTER 8

THE POSSIBLE USES OF THE SYSTEM

8.1. THE USE OF THE SYSTEM AS A DOCUMENTATION RETRIEVAL SYSTEM

The system offers facilities to consult on screen or print different kind of informations useful to the contract writer. Each search is guided by the model.

The system supports following searches :

- all objects to be regulated and their manners of a selected contract type
- all the manners that regulate a given object
- all clauses usable in a selected contract type
- all clauses that regulate a selected object
- all clauses that regulate a selected object in a selected manner
- all references about a contract type
- all references about a selected object
- all references about a selected manner
- all references about a selected clause
- the facts that should be considered in a contract type

8.2. THE USE OF THE SYSTEM IN SUPPORT OF CONTRACT WRITING AND NEGOCIATION

The following scenario resumes how the system will typically be used in the contract writing process. A candidate contractor contacts the contract writer and describes the operations he wants to contract for. The contract writer orders the system to print the questionnaire associated to the needed contract type. This questionnaire is send to the candidate contractor, who returns the answered questionnaire. The answered questionnaire is entered into the system and an adapted contract is generated automatically. The contract writer and the candidate contractor examine the adapted contract and eventually modify it. As soon as the candidate cocontractors are known, the contract writer verifies if the contracts proposed by the those cocontractors are already analysed and memorised by the system. If not he will analyse the absent proposed contract. The contract writer prepares the negociation by reading two types of documents produced through the system. The first document indicates on what objects and in what manners the proposed contract is different from the adapted contract . This enables the contract writer to concentrate on those objects that require negociation. The second type of documents proposes clauses that may reconcile the opposed interests of both contractors. This enables the contract writer to have a 'reponsive' and 'ad rem ' attitude during the negociation process. During negociations the contract writer notes systematically the clauses on which both parties agree and the agreed values for parameters of clauses. The contract writer will communicate those clauses to the system and thereby construct the reconciled contract. Finally the contract writer uses the system to format the reconciled contract in a signable form. As the system insures no coherence between the contract clauses, the contract writer will read the printed contract and eliminate possible incoherences. Such modifications can easily be introduced through the text-editor incorporated in the system. The same text-editor will be used to value the agreed parameters of the clauses. The resulting contract is communicated to the contractors and marks the end of a contract writing process.

8.3. THE USE OF THE SYSTEM IN SUPPORT OF THE DRAFT OF A CALL FOR TENDER

A call for tender is a document in which a candidate contractor communicates his needs to be satisfied to possible cocontractors. A call for tender may contain administrative provisions (object of the market, identification of the caller, the acquisition procedure, form and contents of the tender, time schedule), functional specifications (requirements to be respected by the goods and services in order to satisfy the needs), technical specifications (technical attributes of the goods and services that should be insured), support specifications (assistance to be provided by the cocontractor). The call for tender may also communicate the contractual specifications. These are the essential terms the caller proposes for the contract that will govern the market. The inclusion of contractual specifications is useful to test the willingness of a cocontractor to negociate the contract and to test his willingness to consider his proposals as legally enforceable. The system provides facilities to quickly determine the essential terms of a contract that is adapted to the particular situation of the caller. If a call for tender is made, the administrative provisions should require the adressee to communicate the contract he will propose for the market. Such provision facilitates the use described in previous section.

8.4. THE USE OF THE SYSTEM AS AN EDUCATIVE SYSTEM

Besides the study and discussion of models, mapping functions and clauses, which constitute basic material for a course on contracts, the system provides some facilities which can be exploited in an educative context. The teacher could provide a description of the situation of a candidate contractor for a given contract type, let's say the maintenance contract for sold EDP-hardware. The student is then asked to write a contract that is adapted for the candidate contractor, and to analyse his results in terms of the model. The teacher will have introduced the contract which according to his knowledge seems to be the preferable adapted contract. As the system provides a comparison facility the teacher can quickly detect the difference between the contract written by the student and the preferable adapted contract. This enables the teacher to guide individually each student or to review those parts of his course that have been badly understood by a majority of students. The system provides another facility that can usefully be exploited in an educative context. A student could be asked to discover the differences between the contract proposed by a hardware firm, let's say the IBM hardware maintenance contract and an adapted contract. If he sees differences, the student is asked to formulate reconciling clauses. The student's work will then be compared with the system delivered detection of divergence and the system proposed reconciling clauses.

PART TWO : TOWARDS A REALISATION

This part comments the successive realisation phases we travelled through : functional analysis, design and prototyping. This part has three objectives :

- discuss the methodology applied at each phase,
- justify the major options retained for each phase,
- introduce the reader to the available documentation.

As there is a considerable amount of documentation and the reader may prefer a quick overlook, we organised the documentation in two forms. The first contains the complete documentation and is provided in external appendixes. The second contains only a small subpart of the complete documentation and is provided in sections 9.3 and 10.3. The purpose of the second documentation form is to illustrate briefly style, structure and contents of the complete documentation.

CHAPTER 9

FUNCTIONAL ANALYSIS

9.1. THE LINK BETWEEN THE PRESENT PAPER AND FUNCTIONAL ANALYSIS DOCUMENTATION.

Preceding chapters related the problem to solve to its environnement,proposed two research directions to examine the problem and exposed the basic reasonings behind the solution adopted. The present paper, providing only a global outline of the essential aspects of our solution, needs to be completed by a precise and detailed description of all data and functions constituting the proposed system. The interested reader is therefore refered to the complete conceptual schema, the data dictionary and the function dictionary which were developed during functional analysis.

9.2. ON FUNCTIONAL ANALYSIS METHODOLOGY AND REPRESENTATION

9.2.1. The data dictionary

The complete conceptual schema is given in appendix III.1. It has been expressed in terms of the ENTITY-RELATION model. The accompanying data dictionary, given in appendix III.2 borrows from ISDOS vocabulary and names and defines the entities, attributes, association types and their properties.

9.2.2. The function dictionary

The function dictionary distinguishes between functions and subfunctions. A function is defined as composed of all necessary actions the system should execute to realise an operation which is meaningful to the user. A subfunction is defined as composed of actions that are usable by different functions.

Each function and subfunction was named and its effect specified.The specification of effect is expressed in natural language. One restriction we thought useful was that all data used by the function should be designated by the exact name they have received in the data dictionary. Another useful convention was to express a function when possible in terms of subfunctions . This eliminated redundancy without reducing readability.

The exact definition of screens was generally postponed to the design phase.

At functionnal analysis level, we stated the global requirements for screen interaction. These requirements contain the data to be displayed, the commands that should be available to the user, and how the system should validate and react upon each command. We did not feel at functional analysis time the need for a precise localisation of each information, neither for a precise specification of the dialogue.

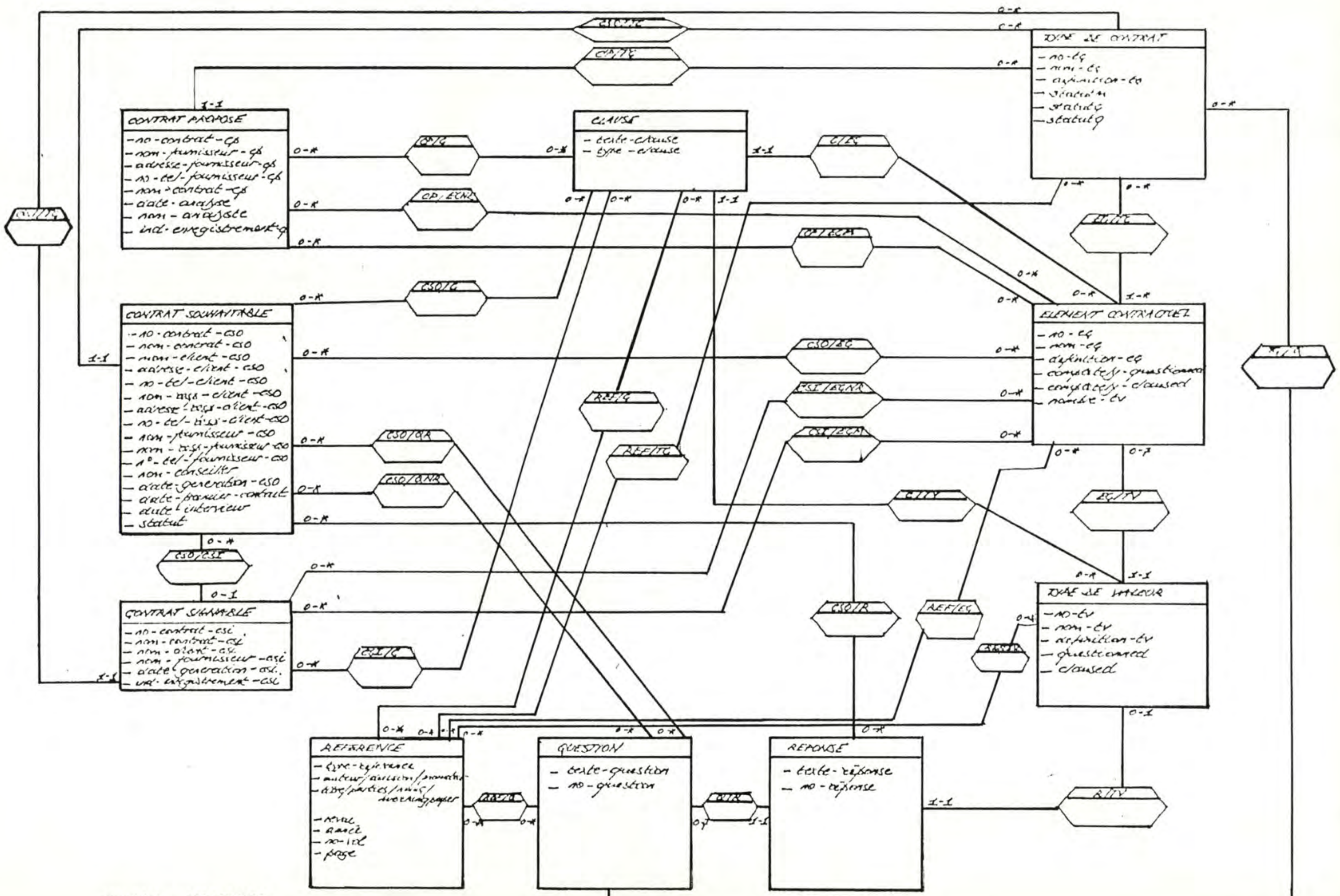
When a function outputted a listing gross layout was specified. The function dictionary is given in appendix III.3.

Many of the representation and methodology options we made, can be explained by the work context. The author behaved at the same time as futur user and builder of the system. As such we looked at the data and function dictionary as useful work documents that acted as a memory for the builder rather than as a precise and unmodifiable contract between user and builder. Also the author was the only builder. The absence of a community of builders reduced the communication difficulty and therefore our interest in a standardised language as provided by ISDOS-methodology. The counterpart of this option was the lack of cross references and the absence of coherence checking tools.

9.3. ILLUSTRATIONS OF FUNCTIONAL ANALYSIS DOCUMENTATION

The complete conceptual schema is given under 9.3.1. In 9.3.2. we illustrate the data dictionary providing the description of one entity, its attributes and the relations to which the entity participates. The function dictionary is illustrated in 9.3.3. by two examples. The first example concerns a typical general function and the second example concerns a typical printing function.

9.3.1. The conceptual schema



SCHEMA CONCEPTUEL

9.3.2. A typical extract of the data dictionary

```
DEFINE ENTITY :
                    CONTRAT -PROPOSE

DESCRIPTION :       contrat propose par une partie contractante

CONSISTS OF :       -no -contrat-CP

                    -nom-fournisseur-CP

                    -adresse-fournisseur-CP

                    -no-telephone-fournisseur-CP

                    -date-analyse

                    -nom-analyste

                    -nom-contrat-CP

                    -ind-enregistrement-CP.

IDENTIFIED BY :
                    no-contrat-CP

RELATED VIA :        CP/TC TO TYPE -DE-CONTRAT CP/C TO CLAUSE.  CP/ECNE TO ELEMENT
                    CONTRACTUEL CP/ECM TO ELEMENT CONTRACTUEL
```

DEFINE ATTRIBUTE :
NO-CONTRAT-CP

DESCRIPTION : no attribue par le systeme a un contrat propose

CONTAINED IN : CONTRAT-PROPOSE

FORMAT is : 9999

IDENTIFIES : CONTRAT PROPOSE

DEFINE ATTRIBUTE :
IND-ENREGISTREMENT-CP

DESCRIPTION : indicateur decrivant les operations deja effectuees en rapport avec le contrat propose.

CONTAINED IN : CONTRAT PROPOSE

FORMAT is : 9

VALUES : 0 = aucune clause n' a ete enregistree,le contrat propose est simplement identifie.

1 = il existe au moins un element contractuel pour lequel le contrat dispose d'une clause enregistree,mais pas tous les elements contractuels ont une clause enregistree, la liste de mise au point est vide

2 = tous les elements contractuels ont recu une clause enregistree du contrat propose, la liste de mise au point est vide

3 = la liste de mise au point n'est pas vide

DEFINE ATTRIBUTE :
NOM-FOURNISSEUR-CP

DESCRIPTION : nom de la personne physique ou morale qui propose un contrat a sa partie cocontractante

CONTAINED IN : CONTRAT-PROPOSE

FORMAT: X(70)

DEFINE ATTRIBUTE GROUP :
ADRESSE-FOURNISSEUR-CP

DESCRIPTION : adresse de la partie contractante telle qu'elle figure au contrat propose.

CONTAINED IN : CONTRAT PROPOSE

CONTAINS : RESIDENCE FORMAT X(50)

RUE FORMAT X(50)

NO-RUE FORMAT X(4)

LOCALITE FORMAT X(35)

PAYS FORMAT X(3)

CODE-POSTAL FORMAT X(6)

BOITE FORMAT X(3)

DEFINE ATTRIBUTE GROUP :
NO-TELEPHONE-FOURNISSEUR-CP

DESCRIPTION : no de telephone de la partie contractante qui propose le contrat a sa partie contractante

CONTAINED IN : CONTRAT-PROPOSE

CONTAINS : INDICATION-PAYS FORMAT X(4)

INDICATIF-ZONE FORMAT X(4)

NO ABONNE FORMAT X(10)

DEFINE ATTRIBUTE :
NOM-ANALYSTE

DESCRIPTION : nom de la personne qui a analyse le contrat propose . Par analyse on entend l'identification des clauses et leur qualification en termes d'elements contractuels et de types de valeur

CONTAINED IN : CONTRAT-PROPOSE.

FORMAT : X(70)

DEFINE ATTRIBUTE :
DATE-ANALYSE

DESCRIPTION : date a laquelle le contrat propose et analyse a ete enregistre dans le systeme

CONTAINED IN : CONTRAT-PROPOSE

FORMAT : JJMMAA

DEFINE ATTRIBUTE :
NOM-CONTRAT-CP

DESCRIPTION :	nom par lequel une partie contractante designe le contrat qu' elle propose
CONTAINED IN :	CONTRAT-PROPOSE
FORMAT :	X (70)

DEFINE RELATION :
CP/TC

RELATES : CONTRAT PROPOSE CONNECTIVITY 1-1

TYPE-DE-CONTRAT CONNECTIVITY 0-*

DESCRIPTION : si (cp,tc) appartient a la relation alors le contrat propose cp est du type de contrat tc.

DEFINE RELATION :
CP/C

RELATES : CONTRAT PROPOSE CONNECTIVITY 0_*

CLAUSE CONNECTIVITY 0_*

DESCRIPTION : si (cp,c) appartient a la relation alors le contrat propose contient la clause c .

DEFINE RELATION :
CP/ECNE

RELATES : ELEMENT CONTRACTUEL CONNECTIVITY 0-*

CONTRAT PROPOSE CONNECTIVITY 0-*

DESCRIPTION : si (cp,e) appartient a la relation alors le contrat propose cp n'a pas encore de clause enregistree pour l'element contractuel e

DEFINE RELATION :
CP/ECM

RELATES : ELEMENT CONTRACTUEL CONNECTIVITY 0-*

CONTRAT PROPOSE CONNECTIVITY 0-*

DESCRIPTION : si (cp,e) appartient à la relation alors le contrat proposé cp a besoin d'une mise au point pour l'élément contractuel e

9.3.3. Typical extracts of the function dictionary

Example 1

FONCTION : MODIFICATION TEXTE CLAUSE

DESCRIPTION :

1. - inviter l'utilisateur a choisir le type de contrat qui caracterise le contrat propose dont on veut modifier une clause en affichant succesive- ment tous les types de contrat deja connus par le systeme PAR SOUSFONCTION IDENTIFICATION TYPE DE CONTRAT

2. - verifier si le type de contrat retenu par l'utilisateur est en etat de recevoir l'operation. Cette verification conduit a rejeter la tentative d'effectuer la modification d'un contrat propose et a afficher un message adapte dans les cas suivants:

   a. le type de contrat n'a pas encore un modele associe (STATUTM = 0). Le message a afficher comprendra NO-TC,NOM-TC,DEFINITION-TC ainsi que le texte " le modele n'est pas encore defini,il n'existe aucun element contractuel"

   b. si le modele du type de contrat n'est pas complet (STATUTM = 1 ou 3) Le message a afficher comprendra NO-TC,NOM-TC,DEFINITION-TC ainsi que le texte " le modele est incomplet"

3. - acquerir l'identification du contrat propose du type de contrat retenu PAR SOUSFONCTION IDENTIFICATION CONTRAT PROPOSE

   s'il n'existe aucune clause enregistree pour le contrat propose retenu (IND-ENREGISTREMENT-CP = 0) alors

   - afficher NO-TC,NOM-TC,DEFINITION-TC, NO-CONTRAT-CP,NOM-CONTRAT-CP, NOM-FOURNISSEUR-CP ainsi que le texte " il n'y a aucune clause enregistree" terminer

4. - acquerir l'identification de l'element contractuel PAR SOUSFONCTION IDENTIFICATION ELEMENT CONTRACTUEL

5. - s'il n'existe pas de clause enregistree pour le contrat propose retenu et l'element contractuel retenu alors

   - afficher NO-TC,NOM-TC,DEFINITION-TC, NO-EC,NOM-EC,DEFINITION-EC ainsi que le texte " il n'y a pas de clause enregistree" terminer

6. - afficher la clause ancienne et obtenir la nouvelle clause modifiee

7. - enregistrer la clause modifiee dans la base de donnees, sauf abandon

8. - la clause ancienne et la clause modifiee doivent etre simultanement presentes a l'ecran

9. apres modification l'utilisateur doit avoir la possibilite de confirmer la modification, de deconfirmer ou d'abandonner l'operation en cas de deconfirmation, l'utilisateur doit pouvoir introduire une nouvelle modification

10. - toute intervention de l'utilisateur doit etre guidee et validee, en cas de faute un message d'erreur approprie doit etre affiche, tout en permettant la reprise de la situation avant l'erreur.

Example 2

FONCTION : IMPRESSION DU QUESTIONNAIRE A REPONDRE PAR UN CLIENT

DESCRIPTION :

```
1. acquerir l'identification du type de contrat dont on veut imprimer le
   questionnaire a repondre par un client PAR SOUSFONCTION
   IDENTIFICATION TYPE DE CONTRAT

2. si STATUTM = 0 alors afficher le texte "LE MODELE N'EST PAS ENCORE
   DEFINI"

        NO-TC, NOM-TC, DEFINITION-TC
        terminer

   si STATUTM = 1 ou 3 alors afficher le texte "LE MODELE EST INCOMPLET"

        NO-TC, NOM-TC, DEFINITION-TC
        terminer

   si STATUT = 0 alors afficher le texte "LE QUESTIONNAIRE N'EXISTE PAS
   ENCORE"

        NO-TC, NOM-TC, DEFINITION-TC
        terminer

   si STATUT = 1 alors afficher le texte "LE QUESTIONNAIRE EST
   INCOMPLET"

        NO-TC, NOM-TC, DEFINITION-TC
        terminer

3. imprimer un listing reprenant les informations suivantes
```

4. la page titre reprend les informations suivantes dans le format ci-apres

```
-----------------------------------------------------------------------------
*                                                                           *
*                          QUESTIONNAIRE                                    *
*                                                                           *
*   TYPE DE CONTRAT NO                                                      *
*   NOM                                                                     *
*   DEFINITION                                                              *
*                                                                           *
*   DATE D'ENVOI                                                            *
*                                                                           *
*   CLIENT                                                                  *
*   RESIDENCE                                           BOITE               *
*   RUE                                                 NO                  *
*   LOCALITE                   PAYS                     CODE POSTAL         *
*                                                                           *
*   TELEPHONE                                                               *
*      PAYS                    ZONE                     ABONNE              *
*                                                                           *
*   RESPONSABLE DU CLIENT                                                   *
*   RESIDENCE                                           BOITE               *
*   RUE                                                 NO                  *
*   LOCALITE                   PAYS                     CODE POSTAL         *
*                                                                           *
*   TELEPHONE                                                               *
*      PAYS                    ZONE                     ABONNE              *
*                                                                           *
*                                                                           *
*   RESPONSABLE DU FOURNISSEUR                                              *
*                                                                           *
*                                                                           *
*                                                                           *
*                                                                           *
*                                                                           *
*   DATE RECEPTION CLIENT                                                   *
*   DATE ENREGISTREMENT                                                     *
*                                                                           *
*                                                                           *
*                                                                           *
*                                                                           *
-----------------------------------------------------------------------------
```

5. on imprimera pour chaque question

```
----------------------------------------------------------------------
*                                                                    *
*                                                                    *
*    QUESTION NO                                                     *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                REPONSES POSSIBLES                                  *
*                                                                    *
*    -----                                                           *
*    !   ! 1                                                         *
*    -----                                                           *
*    -----                                                           *
*    !   ! 2                                                         *
*    -----                                                           *
*    -----                                                           *
*    !   ! 3                                                         *
*    -----                                                           *
*    -----                                                           *
*    !   ! 4                                                         *
*    -----                                                           *
*    -----                                                           *
*    !   ! 5                                                         *
*    -----                                                           *
*                                                                    *
*    REMARQUES                                                       *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
*                                                                    *
----------------------------------------------------------------------
```

6. on assurera un saut de page avant chaque question

CHAPTER 10

DESIGN.

10.1. ON DESIGN METHODOLOGY AND REPRESENTATION

10.1.1. High level module specification.

We viewed the design in terms of modules. For each model we specified its name, its interface and the effect of the model. The naming convention was to choose suggestive names that evoke as much as possible the significance of a module. The interface was specified by naming and defining the format of each input argument and each output argument. When thought useful some indications were given about the values of input and output arguments, which can be interpreted as loose preconditions and postconditions. The effect of a model is expressed in natural language and consists in a gross indication of the module's logic and functions. The description of the effect of the model indicates what the module should do and what other modules are useful to obtain the effect. The modules managing the screens were specified in the same way : name, input interface, output interface and effect. The screens were generally defined at the design level. The definition of a screen consisted in localising each information on the screen. The description of the effect of a screen managing module precises when what information should be displayed and precises the dialogue. Modules reponsable for printing were specified by name,input interface, output interface and effect.The description of the effect of a printing module consisted in a global structure of the output listing, expressed by the symbolism proposed by JACKSON [10]. For each component of that structure a gross layout was provided.

The high level module specifications can be found in appendix IV.1.

10.1.2. The software architecture map

Another document useful in the design activity was the map of the software architecture. The map of the software architecture borrowed from the symbolism proposed by MEYER [13]. The map represents in a condensed way named modules,their interfaces and the calling relationships between those modules. This representation is to a certain degree redundant with the high level module specifications. However it permits a quick global overview of the system structure, without extensive page swapping, as would be required to obtain equivalent information through the module specifications of appendix IV.1. The software architecture map which covers a total of 157 high level modules and 166 data base manipulating routines, can be found in appendix IV.2.

10.1.3. Data base specification

A third design document contains the specification of database manipulation routines. We developed this document by reading the high level module specifications and asking each time what database manipulation routines could assist the high level module. Each newly discovered data base manipulation routine was specified by name,interface and function. Once we had the complete list of data manipulation routines, we transformed the conceptual schema to a schema of possible accesses and indicated graphically the required accesses for each data base manipulating routine. Combining all these graphics resulted in the schema of required accesses.

The basic philosophy for the design of data base manipulation routines consisted to express the data base manipulation routines at the conceptual schema level. This option is justified in 10.2.2.

The specification of data base manipulating routines, the schema of possible accesses and the schema of the required accesses can be found in appendix IV.3, IV.4 and IV.5.

10.2. PRINCIPLES RETAINED FOR THE SOFTWARE ARCHITECTURE

Several principles are governing the software architecture. Each of these principles is discussed and justified.

10.2.1. Independence of screen managing utilities.

We isolated all logic and functions managing screens in specific modules. That means that we oblige each model that needs interaction with the user to obtain or provide information to pass through specialised screen managing modules. The reason for doing this is that we used a particular screen managing utility TRAFFIC 20 . If one day we want or are obliged to use another screen managing utility, the scope of modification is clearly delimited.

10.2.2. Independence of file or data-base software.

We isolated all logic and functions manipulating the data in specific modules. That means that we oblige each module that creates, modifies or deletes data to pass through specialised data base manipulating modules. The justification for doing this is that if one day we want or are obliged to change the file or data-base software, the scope of modification is clearly delimited. Another policy that contributes to independence is by specifying the data base manipulating modules on the conceptual schema level. As such the modules using the data base manipulating modules make no hypothesis on how data are stored, accessed or interrelated. The specification of a data base manipulating module only states what goes in and what comes out. This offers two benefits. First, the programmer gets a practical abstraction of the data base. Second, the system can be developed for different target file or databasesoftwares. The responsibility of the implementor of the data base manipulating modules is to link conceptual schema data structures and functions to the data structures and primitives offered by a particular file or data base software.

10.2.3. Reduction of the number of modules by discovery of common modules

We made an effort to discover a maximum of modules that could be used in different contexts. Each time some functionality might be used at different places we incorporated it in a module. The justification lies in reduced implementation effort and reduced program size. However we tried to avoid modules having low strengh and high coupling.

10.2.4. **Uniformity of man machine dialogue through modules**

When specifying the dialogue part of the effect of screen managing modules, we made an effort to have uniform dialogues.We believe this uniformity is a factor conditionning userfriendliness of the system.

Uniformisation of dialogues was obtained by having on all screens a net separation between the display field and the dialogue field. The separation line is always on line 20. The display field always occupies line 1 to 19 and the dialogue field always occupies line 21 to 24 . The separation line was always formatted in the same way and reminds the user what he is doing.

A second way by wich the dialogues were uniformised was by stable localisation of the dialogue structure. Line 21 generally displayed the commands available. Line 22 generally asked for the retained command and line 24 generally displayed an error message for wrong commands.

The third way to uniformise the dialogue was through a stable vocabulary. Line 22 displayed always "operation choosen ?", whenever the user had to choose a command . This as opposed to a non stable vocabulary in wich on one screen line 22 would display "operation chosen ?" on a second "your choice ?" or on a third "command retained ?".

A fourth way to uniformise the dialogues was through systematizing abbreviations for commands : N systematically stands for Next, P for Previous, A for Abandon, S for Stop, F for Found. As such N was used when the next object,the next clause, the next reference or the next questions was to be displayed.

10.2.5. Unspecified modules reserving seats in global design structure

A certain number of modules are left unspecified or have received only a gross description of function. These modules are situated in the global design structure, and can be developed when the required resources are available. This design principle results from the requirement of incremental development.

10.3. ILLUSTRATIONS OF DESIGN DOCUMENTATION.

In 10.3.1 we provide some typical examples of high level module specifications: we selected a general module, a screen managing module and a printing module. A vertical cut out of the software architecture map is provided in 10.3.2. Data base manipulating routine specifications are illustrated by two routines in 10.3.3. The schema of possible accesses and the schema of required accesses are provided in 10.3.4. and 10.3.5.

10.3.1. Typical examples of high level module specifications

Example 1 : a general module

```
NOM DE MODULE : MODIFICATION-TEXTE-CLAUSE

ARGUMENT D'ENTREE :

ARGUMENT DE SORTIE :

EFFET


1) acquérir l'identification du type de contrat

  via ID-TC  CALLER-ID
             ID-TC
             CODE-RETOUR


  si CODE-RETOUR = A alors terminer

  vérifier si pour le TC identifié, le modèle est défini ou complet
  en consultant STATUTM

  si STATUTM = 0

  alors afficher que le modèle n'est pas encore défini
        via AFF-MESS-TC    CALLER-ID
                           NO-TC
                           NOM-TC
                           DEF-TC
                           IND-MESS a 1
        terminer

 si STATUTM = 1 ou 3
 alors afficher que le modèle est incomplet
        via AFF-MESS-TC    CALLER-ID
                           NO-TC
                           NOM-TC
                           DEF-TC
                           IND-MESS a 3
       terminer.

2) acquérir l'identification du contrat proposé
   via ID-CP  CALLER-ID
              ID-TC
              ID-CP
              CODE-RETOUR

  si CODE-RETOUR = A  alors terminer

  vérifier si le contrat proposé a déjà des clauses enregistrées
  en consultant l'IND-ENREGISTREMENT-CP

  si IND-ENREGISTREMENT-CP = 0 alors afficher message qu'il n'existe aucune
  clause enregistrée pour le contrat proposé
  via AFF-MESS-CPCSOCSI    CALLER-ID
```

```
                                        IND-EC a 1
                                        NO-TC
                                        NOM-TC
                                        DEF-TC
                                        NO-CONTRAT
                                        NOM-CONTRAT
                                        NOM-CLIENT
                                        NOM-FOURNISSEUR
                                        IND-MESS a 15
          terminer.

3) acquérir l'identification de l'élément contractuel via
  via ID-EC   CALLER-ID
              ID-TC
              ID-EC
              CODE-RETOUR

 vérifier si pour l'élément contractuel identifie,il existe déjà une clause
 enregistrée pour ce contrat proposé
 sinon afficher message que pour l'élément contractuel, le contrat proposé
 n'a pas encore de clause enregistrée
 via AFF-MESS -EC   CALLER-ID
                    NO-TC
                    NOM-TC
                    DEF-TC
                    NO-EC
                    NOM-EC
                    DEF-EC
                    IND-MESS a 12
          terminer.

4) afficher le texte ancien et obtenir le texte modifié
  via  AFF-ACO-CLAUSE   CALLER-ID
                        NO-C
                        NOM-C
                        NO-EC
                        NOM-EC
                        NO-TV
                        NOM-TV
                        TEXTE-A
                        TEXTE-N
                        CODE-RETOUR

5) si CODE-RETOUR = V
      alors enregistrer le texte modifié de la clause dans la bd

6) si CODE-RETOUR = A
      alors terminer
```

Example 2 : a screen managing module

NOM DE MODULE : AFFACOCLAUSE

ARGUMENT D'ENTREE :	CALLER-ID	X(30)	
	NO-C	9999	
	NOM-C	X(70)	
	NO-EC	99999	
	NOM-EC	X(70)	
	NO-TV	9	
	NOM-TV	X(70)	
	TEXTE-A	X(480)	
ARGUMENT DE SORTIE :	CODE-RETOUR	X	VALEUR A,V
	TEXTE-N	X(480)	

EFFET :

1. afficher le CALLER-ID sur la memory-line

2. afficher les autres arguments d'entrée

3. inviter à introduire le texte modifié de la clause en affichant "introduisez le texte nouveau de la clause"

4. afficher la C D A ligne et inviter à retenir une opération en affichant "operation choisie?"

5. valider l'opération choisie si non valide afficher "entrée invalide, tapez C/D/A/ boucler en 4.

6. si valide

   si opération choisie = C alors renvoyer CODE-RETOUR a V
                                                       TEXTE-N
   si opération choisie = D alors effacer ce qui a été introduit
                                                    boucler en 3.

   si opération choisie = A alors renvoyer CODE-RETOUR à A

```
----------------------------------------------------------------------------
 1*CONTRAT PROPOSE  NO   no-c                                              *
 2*NOM              nom-c                                                  *
 3*ELEMENT CONTRACTUEL NO       no-ec                                      *
 4*NOM              nom-ec                                                 *
 5*TYPE DE VALEUR      NO            no-tv                                 *
 6*NOM              nom-tv                                                 *
 7*TEXTE ANCIEN DE LA CLAUSE                                               *
 8*texte-a                                                                 *
 9*texte-a                                                                 *
10*texte-a                                                                 *
11*texte-a                                                                 *
12*texte-a                                                                 *
13*texte-a                                                                 *
14*TEXTE NOUVEAU DE LA CLAUSE                                              *
15*texte-n                                                                 *
16*texte-n                                                                 *
17*texte-n                                                                 *
18*texte-n                                                                 *
19*texte-n                                                                 *
20*texte-n                                                                 *
21*------------------------------(caller-id)-------------------------------*
22*CONFIRMATION = C, DECONFIRMATION = D, ABANDON = A                       *
23*(INTRODUISEZ LE NOUVEAU TEXTE DE LA CLAUSE                              *
  *(OPERATION CHOISIE ?                                                    *
24*ENTREE INVALIDE, TAPEZ C/D/A                                            *
============================================================================
```

Example 3 : a printing module

```
NOM DE MODULE :        IMPRESSION-MOD

ARGUMENT D'ENTREE : ID-TC           999
                    ID-EC           9(5)
                    IND-EC-TC       X        VALEUR   E,T

ARGUMENT DE SORTIE :

EFFET :
```

1. si IND-EC-TC=T

alors imprimer un listing ayant la structure suivante

```
                  _________________
                  !   LISTING     !
                  _________________
                          X
                         X X
                        X   X
       ---------------     ----------------
       !    PAGE-TC  !     !      PAGE-EC *!
       _______________     ________________
```

a. * = nombre d'éléments contractuels associés au type de contrat de No ID-TC

b. PAGE-EC est relatif a un élément contractuel associé au type de contrat de No ID-TC

2. si IND-EC-TC = E

alors imprimer un listing avec la structure suivante

```
                  _________________
                  !   LISTING     !
                  _________________
                          X
                         X X
                        X   X
       ---------------     ----------------
       !    PAGE-TC  !     !      PAGE-EC !
       _______________     ________________
```

PAGE-EC est relatif a l'élément contractuel dont le NO-EC figure dans ID-EC

NOTE :

1) PAGE-TC a le format suivant

```
 ________________________________________________
!   IMPRESSION MODELE RELATIF AU                 !
!   TYPE DE CONTRAT                              !
!                                                !
!                                                !
!                                                !
!   NO                                           !
!   NOM                                          !
!   DEFINITION                                   !
!                                                !
!                                                !
!                                                !
!   DATE-IMPRESSION                              !
!                                                !
!   REFERENCES                                   !
!                                                !
!                                                !
!                                                !
!________________________________________________!
```

2) PAGE-EC a la structure suivante

```
                    _______________
                    !             !
                    !  PAGE-EC    !
                    !             !
                    !_____________!
                           *
                       *       *
                   *               *
               *                       *
         ____________            _____________
         !  EC      !            !   TV     * !
         !          !            !            !
         !__________!            !____________!
```

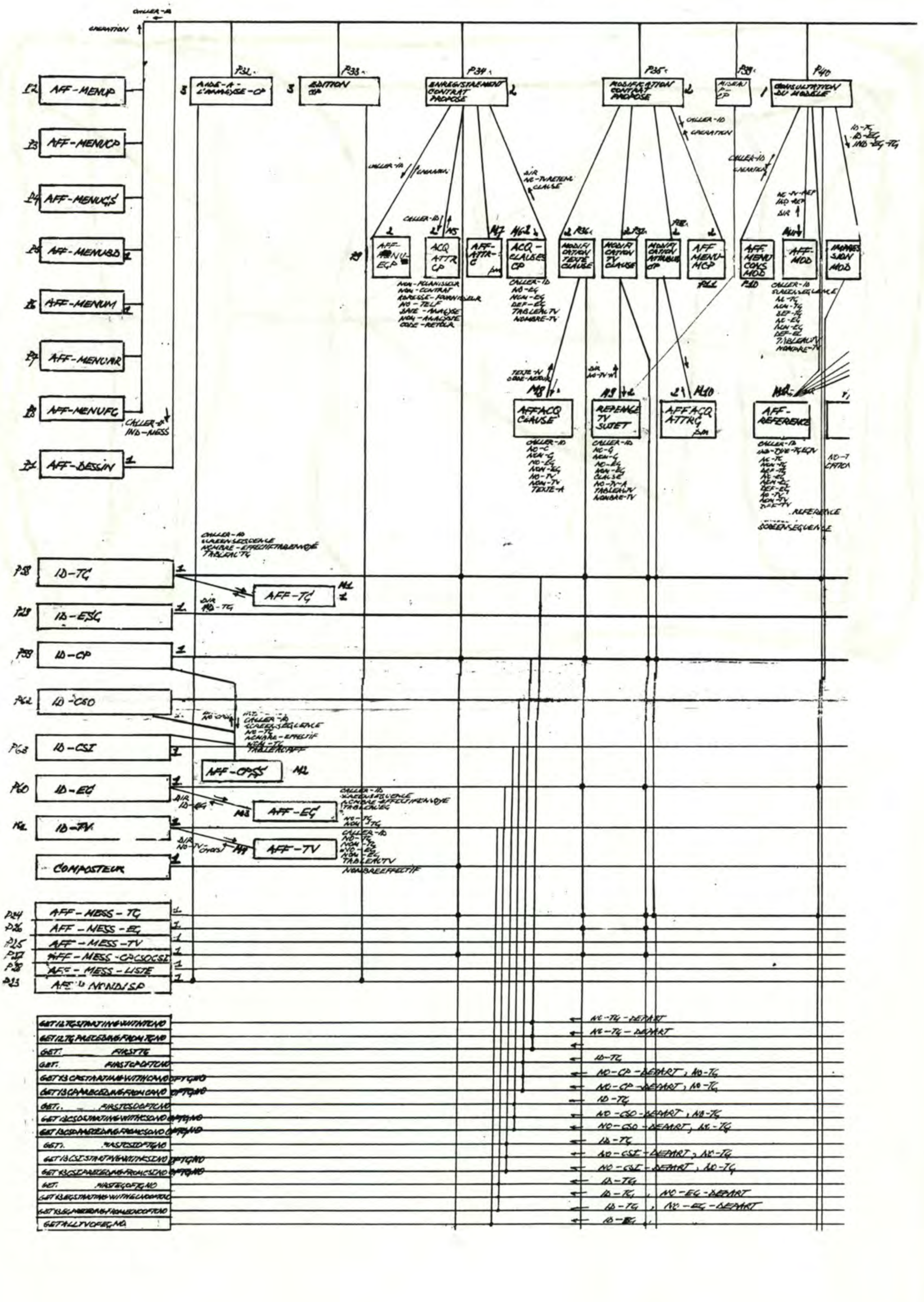
3) EC a le format suivant

```
 ________________________________________________
!                                                !
!   ELEMENT CONTRACTUEL NO                       !
!   NOM                                          !
!   DEFINITION                                   !
!                                                !
!   REFERENCES                                   !
!                                                !
!                                                !
!            TYPES DE VALEUR POSSIBLES           !
!            *************************           !
!                                                !
!                                                !
!________________________________________________!
```

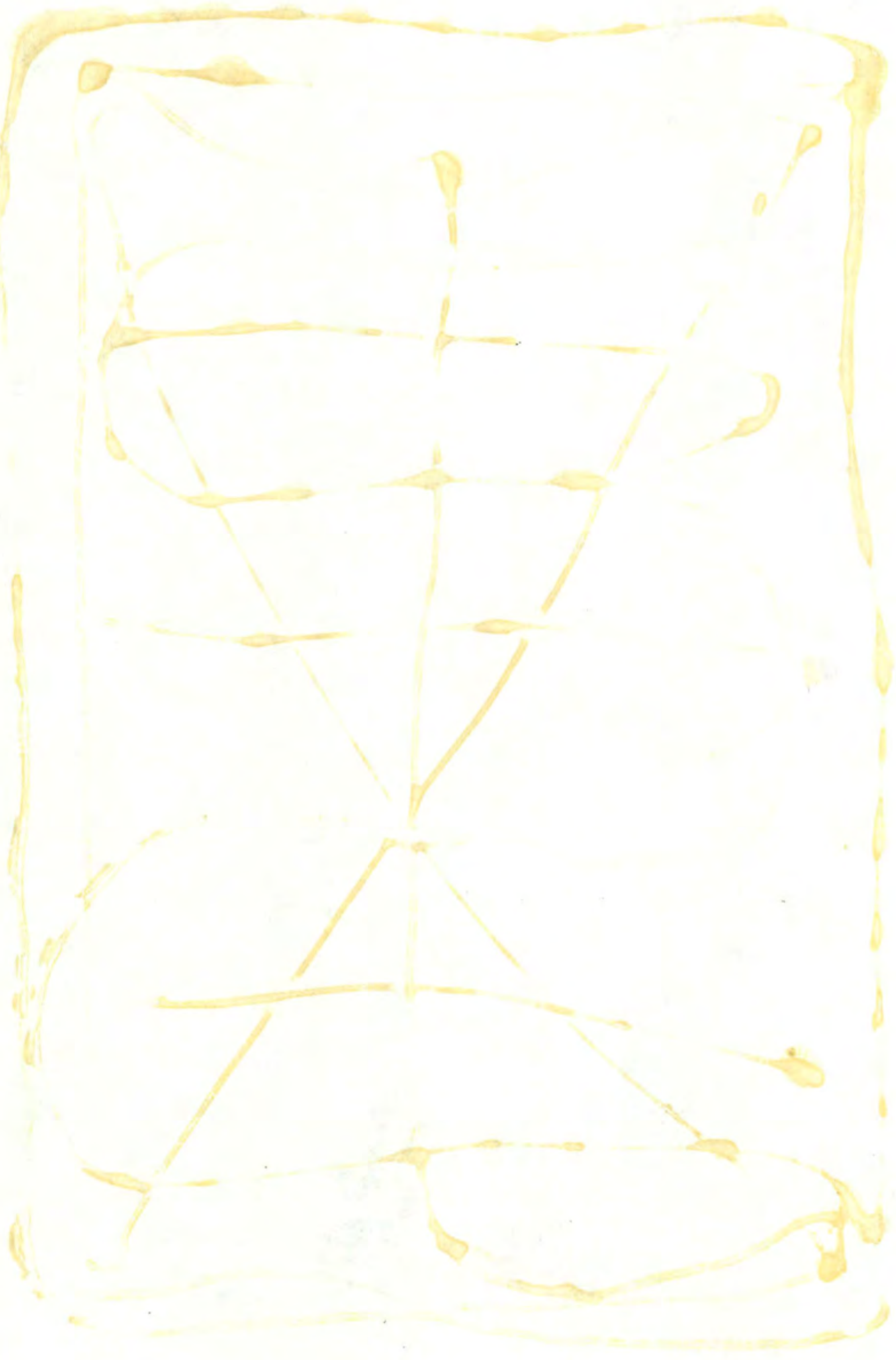
4)le programme doit assurer un saut de page avant chaque element contractuel

5) TV a le format suivant

```
 ______________________________________
!                                      !
!   TYPE DE VALEUR   NO                !
!   NOM                                !
!   DEFINITION                         !
!                                      !
!                                      !
!   REFERENCES                         !
!                                      !
!                                      !
!______________________________________!
```


AFF-MENUP
AFF-MENUCP
AFF-MENUCS
AFF-MENUSD
AFF-MENUM
AFF-MENUAR
AFF-MENUFC
AFF-DESSIN
AIDE-A-L'ANALYSE-CP
EDITION CP
ENREGISTREMENT CONTRAT PROPOSE
MODIFICATION CONTRAT PROPOSE
MISE EN P-CP
CONSULTATION DU MODELE
AFF-MENU-EGP
ACQ ATTR CP
AFF-ATTR-C
ACQ-CLAUSES CP
MODIFICATION TEXTE CLAUSE
MODIFICATION TV CLAUSE
MODIFICATION ATTRIBUT CP
AFF MENU MCP
AFF MENU CONS MOD
AFF-MOD
IMPRESSION MOD
AFFACQ CLAUSE
REFERENCE TV SUJET
AFFACQ ATTRG
AFF-REFERENCE
ID-TG
AFF-TG
ID-ESG
ID-CP
ID-CSO
ID-CSI
AFF-CPSS
ID-EG
AFF-EG
ID-TV
AFF-TV
COMPOSTEUR
AFF-MESS-TG
AFF-MESS-EG
AFF-MESS-TV
AFF-MESS-CPCSOCSI
AFF-MESS-LISTE
AFF-NONDISP
GET12TGSTARTINGWITHTGNO
GET12TGPRECEDINGFROMTGNO
GET FIRSTTG
GET FIRSTCPOFTGNO
GET13CPSTARTINGWITHCPNOOFTGNO
GET13CPPRECEDINGFROMCPNOOFTGNO
GET FIRSTCSOOFTGNO
GET13CSOSTARTINGWITHCSONOOFTGNO
GET13CSOPRECEDINGFROMCSONOOFTGNO
GET FIRSTCSIOFTGNO
GET13CSISTARTINGWITHCSINOOFTGNO
GET13CSIPRECEDINGFROMCSINOOFTGNO
GET FIRSTEGOFTGNO
GET13EGSTARTINGWITHEGNOOFTGNO
GET13EGPRECEDINGFROMEGNOOFTGNO
GETALLTVOFEGNO
NO-TG-DEPART
NO-TG-DEPART
ID-TG
NO-CP-DEPART, NO-TG
NO-CP-DEPART, NO-TG
ID-TG
NO-CSO-DEPART, NO-TG
NO-CSO-DEPART, NO-TG
ID-TG
NO-CSI-DEPART, NO-TG
NO-CSI-DEPART, NO-TG
ID-TG
ID-TG, NO-EG-DEPART
ID-TG, NO-EG-DEPART
ID-EG
CALLER-ID
CREATION
CALLER-ID
SCREENSEQUENCE
NOMBRE-EFFECTIFTRAENVOYE
TABLEAUTG
TABLEAUTV
NOMBREEFFECTIF
REFERENCE
SCREENSEQUENCE

10.3.2. A vertical cut out of the software architecture map



10.3.3. Typical extracts of data base manipulating routine specifications

Example 1 :

following routine delivers the clause of a specified proposed contract and of a specified object

```
NOM DE MODULE : GETCLAUSEOFCPNOANDECNO

ARGUMENT D'ENTREE : ID-EC    9(5)
                    ID-CP    9(4)

ARGUMENT DE SORTIE : ID-EC-EXISTS   BOOLEAN
                     ID-CP-EXISTS   BOOLEAN
                     CLAUSE-EXISTS  BOOLEAN
                     CLAUSE
                       TEXTE-CLAUSE   X(480)
                       TYPE-CLAUSE    9
                     OK             BOOLEAN.
```

EFFET :

1. vérifier s'il existe un élément contractuel de no ID-EC

   sinon mettre ID-EC-EXISTS a false, OK a false et terminer.

   si oui mettre ID-EC-EXISTS a true et passer en 2.

2. vérifier s'il existe un contrat proposé de no ID-CP

   si non mettre ID-CP-EXISTS a false, OK a false et terminer.

   si oui mettre ID-CP-EXISTS a true et passer en 3.

3. vérifier s'il existe une clause

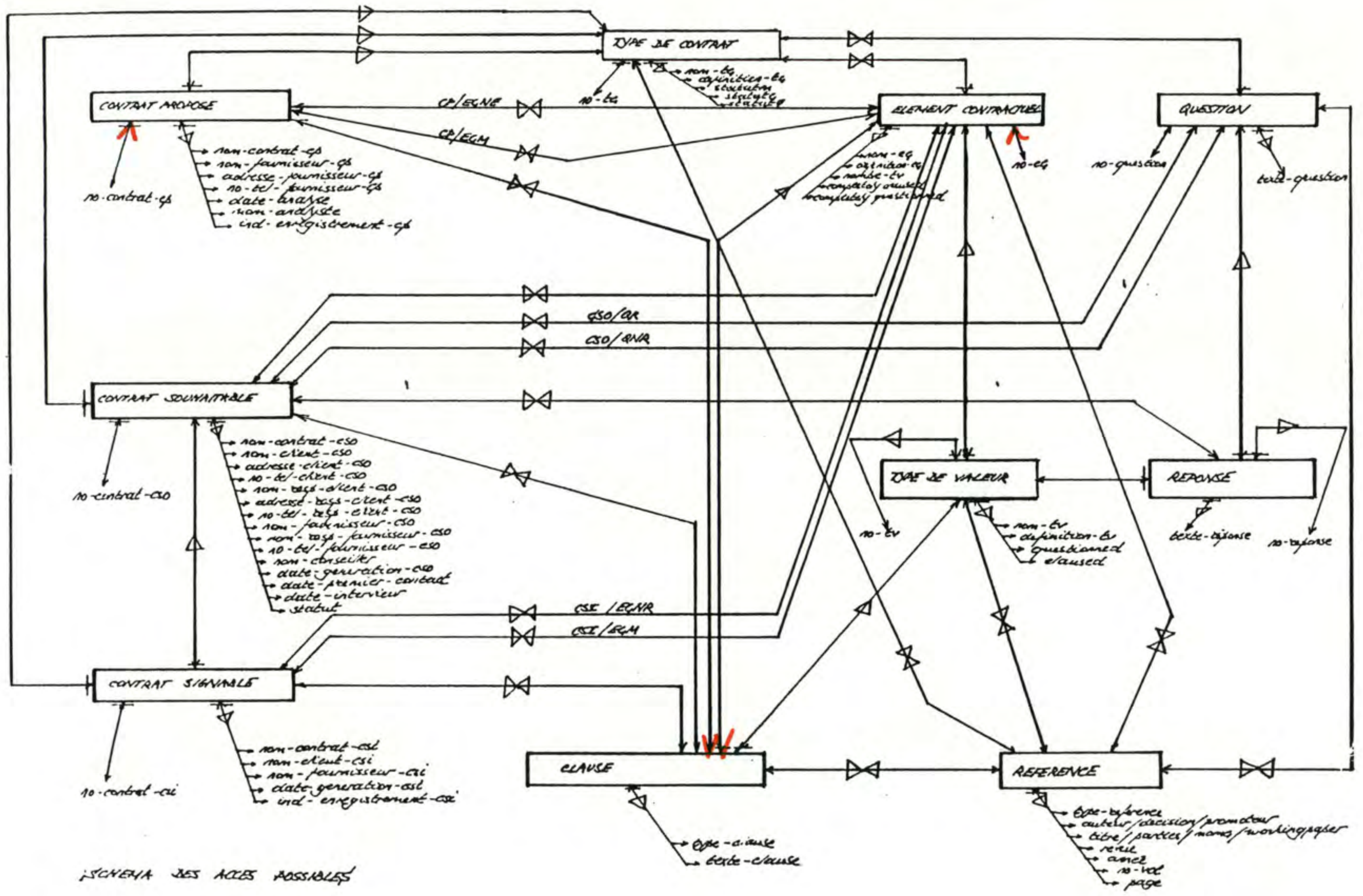
   si non mettre CLAUSE-EXISTS a false, OK a false, terminer.

   si oui mettre CLAUSE-EXISTS a true et passer en 4.

4. mettre texte-clause dans TEXTE-CLAUSE

   mettre type-clause dans TYPE-CLAUSE

5. si pas de problème mettre OK a true.



SCHEMA DES ACCES POSSIBLES

Example 2 :

following routine modifies the clause of a specifed proposed contract and of a specified object

NOM DE MODULE : MODIFYCLAUSEOFCPNOANDECNO

```
ARGUMENT D'ENTREE : ID-EC      9(5)
                    ID-CP      9(4)
                    NEW-CLAUSE X(480)

ARGUMENT DE SORTIE : ID-EC-EXISTS    BOOLEAN
                     ID-CP-EXISTS    BOOLEAN
                     CLAUSE-EXISTS   BOOLEAN
                     OK              BOOLEAN.
```

EFFET :

1. vérifier s'il existe un élément contractuel de no ID-EC

   sinon mettre ID-EC-EXISTS a false, OK a false et terminer.

   si oui mettre ID-EC-EXISTS a true et passer en 2.

2. vérifier s'il existe un contrat proposé de no ID-CP

   si non mettre ID-CP-EXISTS a false, OK a false et terminer.

   si oui mettre ID-CP-EXISTS a true et passer en 3.

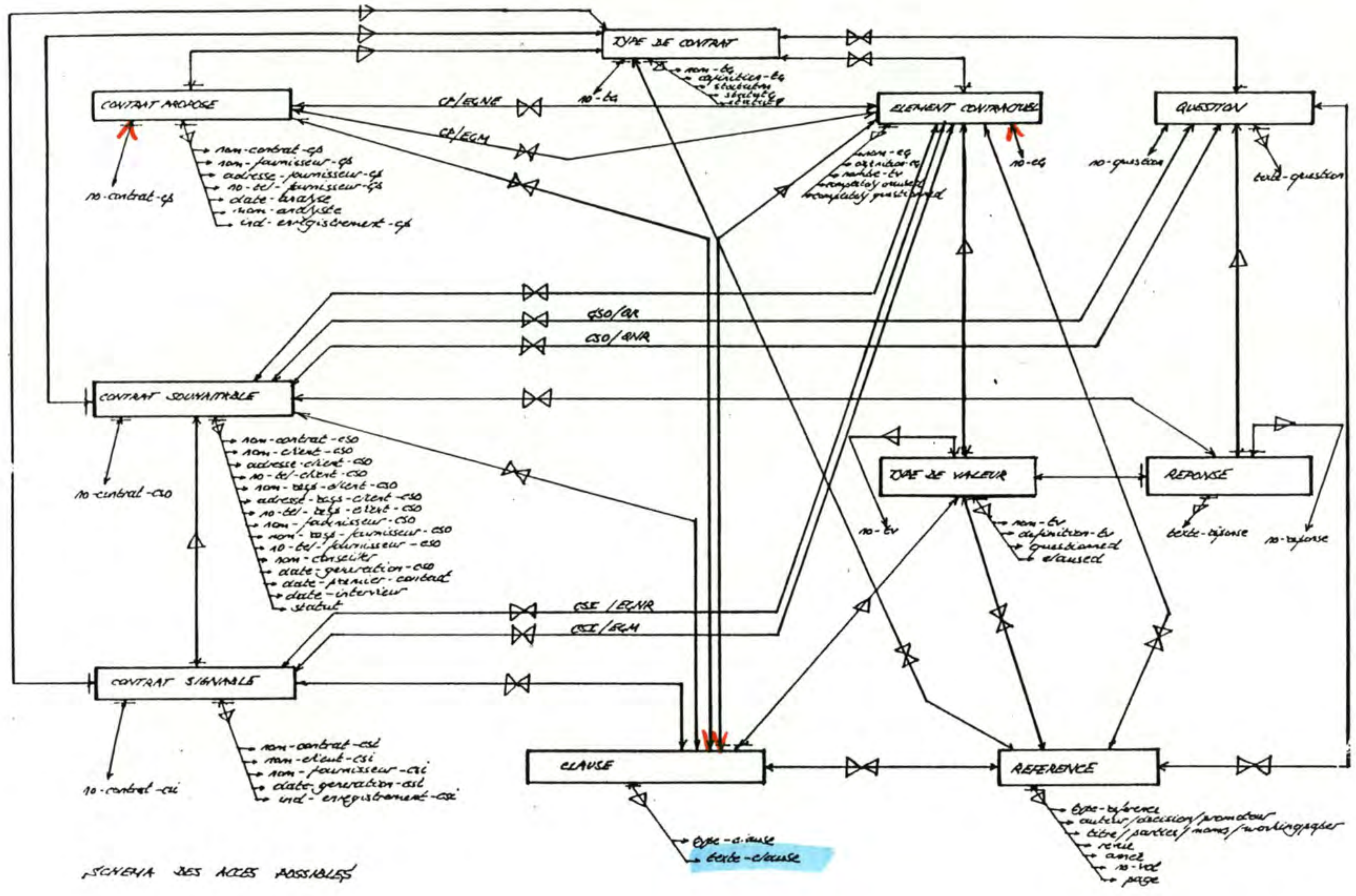
3. vérifier s'il existe une clause

   si non mettre CLAUSE-EXISTS a false, OK a false, terminer.

   si oui mettre CLAUSE-EXISTS a true et passer en 4.

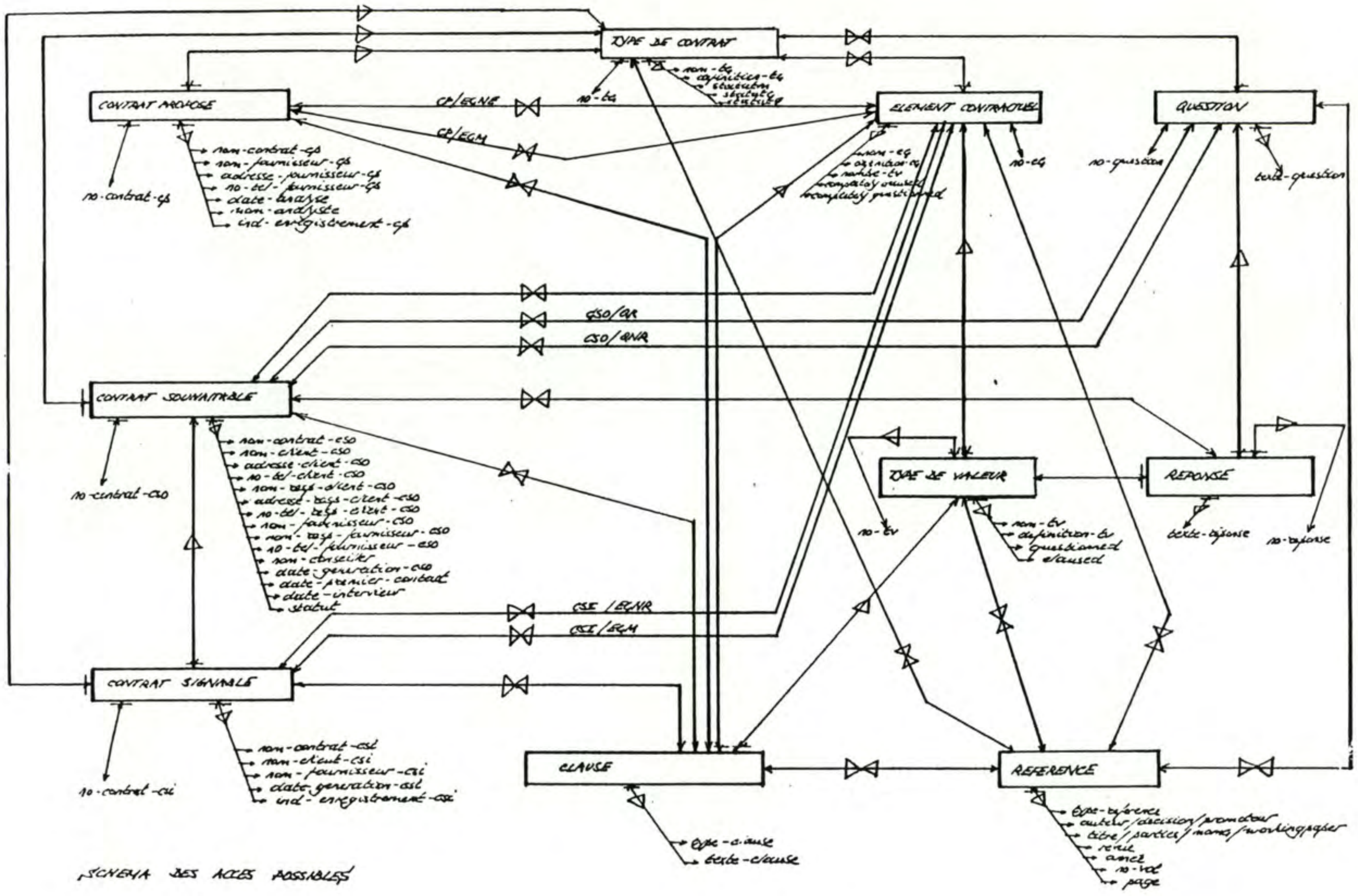
4. modifier la clause par NEW-CLAUSE

5. si pas de probleme mettre OK a true.



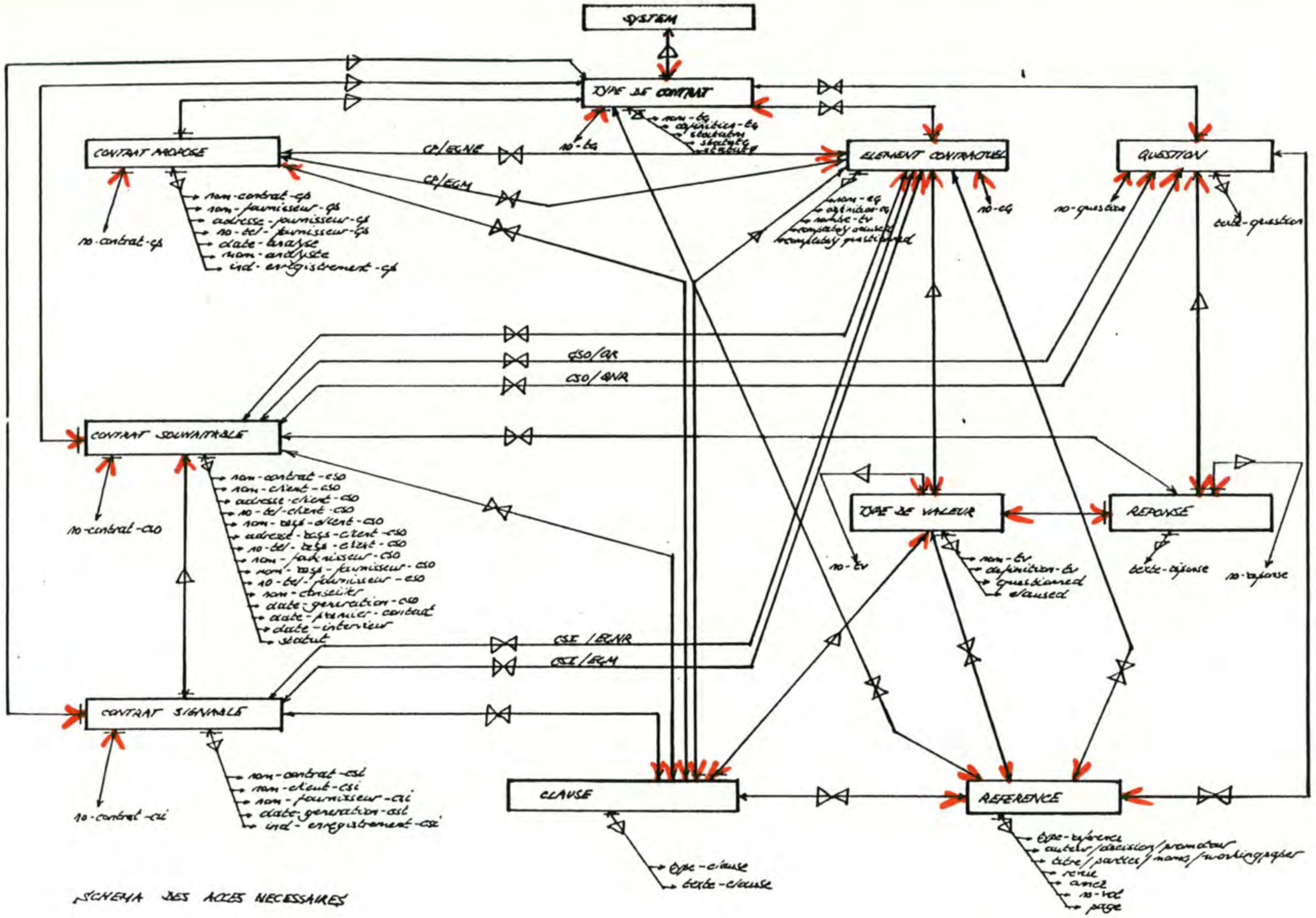
SCHEMA DES ACCES POSSIBLES

10.3.4. Schema of possible accesses



SCHEMA DES ACCES POSSIBLES

10.3.5 Schema of required accesses



SCHEMA DES ACCES NECESSAIRES

CHAPTER 11

PROTOTYPING

As conception and specification of the proposed solution took 5 months, and our time ressources were limited, we had to make a choice on the degree of implementation of the proposed solution. This choice is discussed in 11.1. In 11.2 we describe the applied prototyping methodology. 11.3 provides pointers to prototype documentation and 11.4 indicates the significance of the prototype for further development of the proposed solution.

11.1. THE CHOICE BETWEEN TOTAL IMPLEMENTATION, PARTIAL IMPLEMENTATION AND PROTOTYPING.

Making a choice about the degree of implementation of the proposed solution implies we know the possible implementation degrees and are able to compare them on basis of some comparison criteria. Therefore we precise the basic implementation alternatives (11.1.1.) and indicate the comparison criteria used (11.1.2.). In 11.1.3. we justify our choice for prototyping.

11.1.1. The implementation alternatives

We considered three implementation alternatives : total realisation, partial realisation and prototyping. By total realisation we understand complete data base development, coding of each module,complete validation and documentation of the system implementation. By partial realisation, we understand data-base development - comprising global schema implementation and development of required manipulating routines - and the coding,validation and documentation of a limited number of modules providing to the user some system functions. By prototyping we understand any technique that with a minimum of effort simulates as close as possible a specified system.

11.1.2. The comparison criteria

The different implementation alternatives may be appreciated in regard of the following criteria.

	TOTAL IMPLEMENTATION	PARTIAL IMPLEMENTATION	PROTOTYPE
COST	high	medium	low
TIME	7 months	4,5 months	1,5 month

REUTILISABILITY OF CODE	complete	complete	partial, absent
ADAPTABILITY TO CRITICISM	reduced	reduced	high
DATA CONTENTS DEVELOPMENT	absent	absent	possible
DEGREE OF REAL SYSTEM RESSEMBLANCE	high	low	high
LEARNING EFFECT	high	high	reduced

Cost refers to total resources consumed by the implementation alternative. Time refers to the estimated realisation time of each implementation alternative. Reutilisability of code is present when the code developed by the implementation alternative can become code of the system to develop. Adaptability to criticims is high when limited resources are wasted at the moment the need for change is recognised and changes can be made economically. The data content development refers to the degree the system will have been provided with data when the implementation alternative is finished at the estimated time. Degree of real system ressemblance refers to the likeness between the result of the second or third implementation alternative and the result of the total implementation alternative.The ressemblance is appreciated in terms of user's perception. The learning effect refers to the experience the implementor would gain in each implementation alternative.

11.1.3. Justification of the retained implementation alternative

We immediately rejected the total implementation alternative. The costs would be high and anyway we were short of time. Also the estimated 7 months would have delivered only an 'empty' system with no data (models,contracts, clauses, mapping functions). The remaining choice lead to the adoption of prototyping. The prototyping alternative was prefered versus the partial implementation alternative not only for cost and time reasons, but also because we discovered the following implementation law. This implementation law can be stated as "the less you implement user functions, the greater will be the per function implementation effort required". This law results from the fact that the design architecture was characterised by an important proportion of commonly usable modules that should be implemented whatever is the number of user functions realised. In contexts characterised by such implementation law, marginal time or cost will strongly decrease as the number of user functions realised increases : making the first user function available may represent 50 % of total cost. The last function added may represent less than 1% of total cost. Another reason why we eliminated the partial implementation alternative is that after 4,5 months the system delivered would also be an empty one. Furthermore a partial implementation doesn't permit the users to appreciate the non implemented functions. We believed prototyping was the only reasonable alternative: low costs, reduced realisation time, high degree of real system ressemblance, high adaptability to criticisms formulated by users and the possibility of execution on simulated data constitute qualities that are appropriate when the system to implement is based on non experienced new ideas. We accepted with some pain the drawbacks of the prototyping. First, the code of the prototype will be wholly or partially lost. Second, the challenge to realise completely a medium size project was left unanswered.

11.2. THE PROTOTYPING METHODOLOGY.

Each prototyping effort is characterised by a fundamental tension between two constraints that should be respected simultaneously. The first constraint is that the prototype should have a sufficient degree of ressemblance with the specified system. The second constraint is that the preceding objective should be realised as economically as possible in terms of costs and time. The two constraints combined interact diabolically : the more you simplify the prototype, the less the prototype will ressemble the specified system and by consequence the more you reduce the opportunity for valid reaction by the user. Inversely, the less you simplify the prototype, the greater will be the ressemblance with the specified system and by consequence the greater will be the opportunity for valid user reaction. These observations forced us to make a compromise of which the major aspects are discussed in following paragraphs.

11.2.1. Real user-system interface

As to increase the degree of ressemblance between the prototype and the specified system, we implemented completely the 60 screen managing modules. This seemed important as the user perception of the system is basically his perception of the screen interface and only secondary his perception of the internals of the system. This permits not only to have a prototype which has the same interface as the real system, but also makes the code of screen managing modules reusable when the real system will be totally implemented. The code of the screen managing modules is the only code of the prototype which is reusable.

11.2.2. Incorporation of experimental data

The specified system totally implemented would constitute an empty structure. As the user perception of the system is not only in terms of structure but also in terms of contents, we developped data for the prototype. This permits to experiment the system at a certain point of its history. We developped and integrated data for the contract type "maintenance of sold EDP hardware". The model, a proposed contract, an adopted contract, a reconciled contract, the mapping function, contract clauses and references were represented wether as tables in central memory, wether as the contents of relative COBOL files.

11.2.3. System dynamics not systematically simulated.

In the real system the succession of functions insures the dynamics of the system. For instance, when the function 'entry of an object' is executed and the user next consults the list of objects, he should discover the previously entered object. This perception of dynamics is not systematically insured in the prototype. The prototype permits the user to enter an object exactly as in the real system. However as soon as the dialogue organising the entry of an object is finished, the prototype discards the entered data. Therefore when the user of the prototype wants to consult the list of objects after having entered an object, he will not discover the previously entered object as part of the list. Such policy was adopted basically for two reasons. First, if we wanted to gain time, we had to simplify somewhere : the perception of real user-system interface with experimental data was thought more interesting than the perception of system dynamics. Second, enabling the user of the prototype to execute freely any function with the correct dynamics may have lead to the loss of the experimental data. However when thought appropriate, we simulated the dynamics of the real system for some system functions but limited to the duration of a work session. This means that as long as the user exercised some functions without leaving the system he will find back the old situation. As soon as he leaves the system, the prototype abandons the entered data and falls back to the experimental data.

11.2.4. Multiple system states

The real system is at given moment of observation only in one state. The perception of real system behaviour in all possible states requires to exercise successively functions modifying the system state. As we adopted the policy described in preceding paragraph, the prototype was incapable of realising such perception. Therefore we introduced the principle of multiple system states. This principle means that the prototype simulates the system as if it had many states at a same moment of observation. The following example may illustrate the objective behind the principle of multiple states. The function 'print out the model' is only possible when the model is complete. A model is complete when each object has at least one manner of regulating. In the real system it's impossible to have simultaneously the function 'print out the model' to be executed and another function 'display the manners of an object' to message that there exists no manner associated to the object. In the prototype such situations are tolerated and encouraged because they permit the user to perceive systematically the different possible reactions of the system.

11.3. THE PROTOTYPE DOCUMENTATION.

Appendix V. regroups all prototype documentation,comprising screen managing programs, data managing programs and 'functional' programs with their accompanying test programs. A cross reference table establishes the links between prototype programs and the modules of the software architecture. For each screen managing program the associated screen descriptions are provided.

11.4. THE SIGNIFICANCE OF THE PROTOTYPE

We believe the realisation of the prototype is justified as it enables an easy familiarisation with the proposed system and provides possibilities for thorough evaluation.

The prototype permits the user to examine the system without tedious reading of functional analysis documents. We estimate that reading and understanding the system specifications may take a complete work week. The prototype offers the advantage to familiarise the users with the system in less than a day. The only precondition for manipulating usefully the prototype is the reading of the present paper. As the prototype is highly interactive and by its nature defensive to any user action, we didn't find any necessity for drafting a prototype user manual.

We hope the prototype will be criticised extensively by the community of contract writers and computer scientists. An evaluation form is planned that will structure and standardise the remarks of the prototype users. On basis of the evaluation forms and informal discussions the decision will have to be made to abandon the system or to realise it totally taking into account the user observations. The second option starts another phase in the DSS development, but that's another story.

CHAPTER 12

CONCLUSION

I hope this paper has contributed to the research on the links between computers and law. This research field deals with computer law - the juridical aspects of EDP phenomena - and with legal informatics - EDP technology used in the law sphere -.

The main results of the present work are twofold : first, we traced the proposed solution from its requirements down to EDP-technology; secondly, we realised a prototype enabling to evaluate the specified system on simulated data.

Now the time has come for a serious evaluation. As the author is strongly convinced of the benefits of independent and "egoless" evaluation [18], he encourages the reader to confirm or to disprove St-Simon's phrase "... et tout ce que j'en appris ne fit que nourrir l'idée que je m'en étais formée." .

The present work is largely imperfect, suffering from incompleteness and a lack of exploration of alternative research directions. We didn't study at the same level of detail all system aspects. As such we gave only a major outline of the authorisation subsystem. Also important aspects of the system were not considered at all : security, data base quantification, hardware configuration, text-editors and formatters to be incorporated in the system, data base integrity in multiple user context and the technical and economic aspects of possible exploitation modes of the system were not examined. Alternative research directions were not explored because of lack of time, lack of imagination or both. However more powerful modelling techniques and other ways for generating adapted contracts could be imagined and developped. Each of these is worth a paper on its own.

BIBLIOGRAPHY

[1] BENNET.
Proceedings of the sixth and seventh Annual Conferences of the Society for management Information Systems: Integrating Users and Decision Support Systems.
J.D. White, Ann Arbor - University of Michigan, .
pages 77-86.

[2] BRANDON, D. H. and SEGELSTEIN, S.
Data processing Contracts - Structure, Contents and Negociation.
Van Nostrand Reinhold Cy - New-York, 1976.

[3] contrat CECUA.
Groupe de travail juridique de la Confédération Européenne des Associations d'Utilisateurs des Technologies du Traitement de l'information.
Computer Organ Inform, ISSN 07703848, 1983.

[4] CORNELIS & STAESENS.
Formulaire d'actes notariés.
SWINNEN - Bruxelles, 1980.

[5] Centre de Recherches Informatique et Droit des Facultés Universitaires de Namur.
Le droit des contrats informatiques . Principes - applications.
Précis de la Faculté de droit de Namur n.4 - Larcier, Bruxelles, 1983.

[6] DELVAUX, A.
Traité juridique des batisseurs.
Bruxelles, 1968.

[7] .
Editions de l'Association des Consommateurs.
Test-Achats, Bruxelles, .

[8] GINZBERG M.J.
Ph. D. dissertation: A Process Approach to Management Science Implementation.
Institute of Technology, Massachusetts, .

[9] GRACE, B.F.
Training Users of Decision Support Systems.
IBM Research Report RJ1790, San Jose - California, 1976.

[10] JACKSON, M.A.
Principles of Program Design.
Academic Press, 1978.

[11] KEEN Peter G.W.
Computer Systems for Top Managers : A Modest Proposal.
Sloan Management Review 18(1):1-17, .

[12] KEEN Peter G.W.
Decision support systems : a research perspective. Decision Support Systems. Issues and Challenges.
PERGAMON Press - OXFORD, 1981.

[13] MYERS, GJ.
Reliable Software through Composite Design.
Petrocelli/Charter, 1975.

[14] NESS D.N.
Decision support Systems : theories of design.
paper presented at the Office of Naval Research Conference on Decision Support Systems, Wharton School, University of Pennsylvania, Philadelphia, November 4-7, 1975.

[15] .
Recueil de modèles d'actes et de contrats professionnels et privés.
Ed. Service, Bruxelles, 1983.

[16] STABELL.
Decision Making Research: Decision research : description and diagnosis of decision making in organizations.
HERADSTREIT D. and NARVESEN O., Oslo, 1977.

[17] THIRAN.
Compilation des clauses usuelles des contrats informatiques.
not edited, available at CRID, Rempart de la Vierge 5, 5000 NAMUR, Belgium, .

[18] WEINBERG, G.M.
The psychology of Computer Programming.
Van Nostrand Reinhold Ltd, NEW YORK, 1971.

LIST OF APPENDIX

I. OUTPUT DOCUMENTS PRODUCED BY THE PROTOTYPE

II. THE COHERENCE SUBSYSTEM

II.1. State transition function, output function

II.2. List of Messages

II.3. List of Conditions

III. FUNCTIONAL ANALYSIS DOCUMENTATION

IV. DESIGN DOCUMENTATION

V. PROTOTYPE DOCUMENTATION

TABLE OF CONTENTS

LIST OF FIGURES