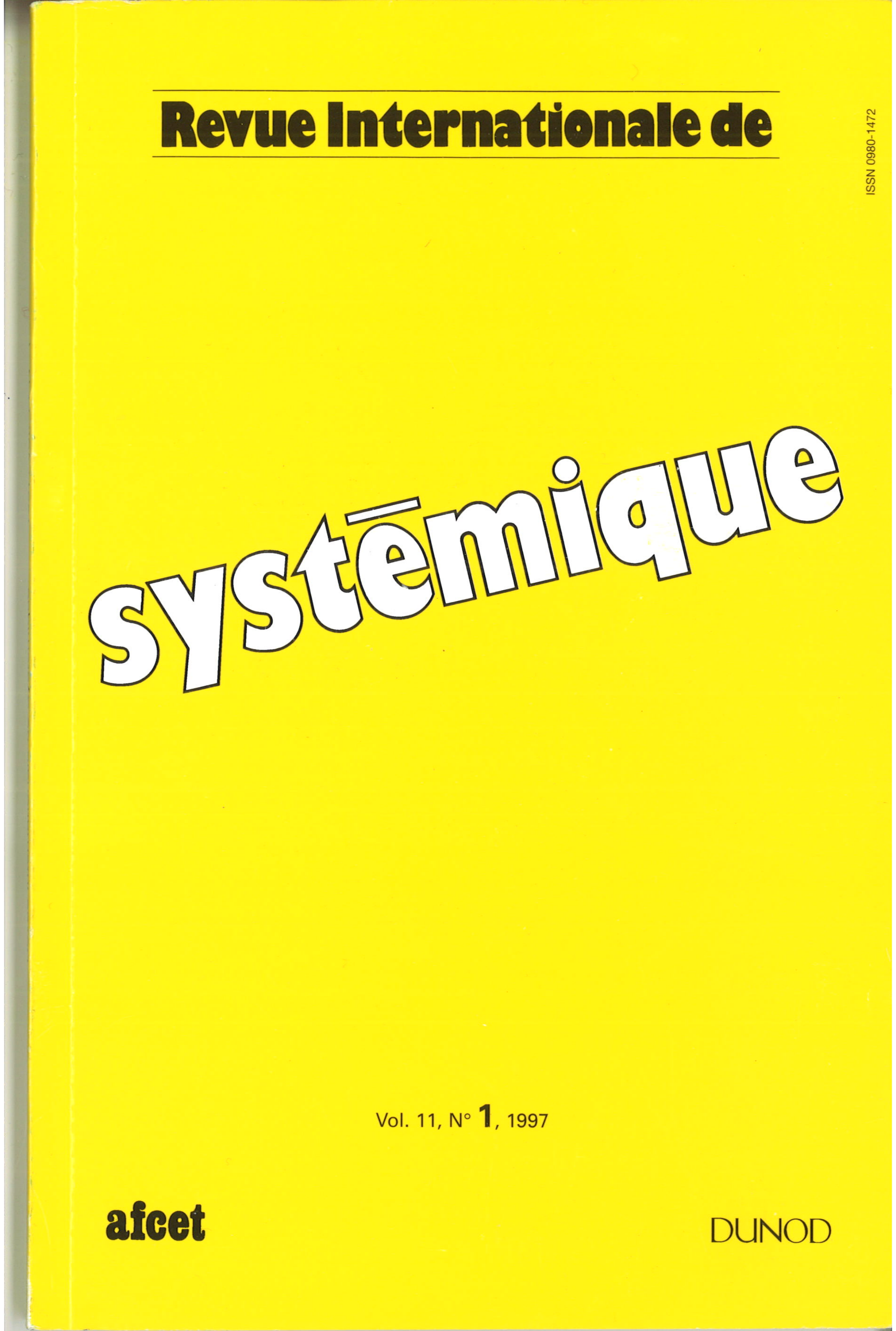

AFSCET

Revue Internationale de systémique

Revue Internationale de Sytémique

volume 11, numéro 1, pages 95 - 107, 1997

Semantic network of action for conception and evaluation of interfaces

Charles-Albert Tijus, Sébastien Poitrenaud, Javier Barcenilla and Jean-François Richard

REVUE INTERNATIONALE DE SYSTÉMIQUE
Vol. 11, N° 1, 1997, pp. 95 à 107

SEMANTIC NETWORKS OF ACTION FOR CONCEPTION AND EVALUATION OF INTERFACES

Charles-Albert TIJUS, Sébastien POITRENAUD,
Javier BARCENILLA and Jean-François RICHARD [1]

Résumé

Dans la lignée des modèles informatiques qui formalisent le savoir-faire de l'utilisateur sous forme de systèmes de production, sous forme de grammaire d'actions, ou encore de leurs extensions conceptuelles, Procope est un modèle de la représentation des connaissances sous la forme de Réseaux Sémantiques d'Actions (R.S.A.) dans lesquelles se réalisent l'équivalence et l'implication entre connaissances déclaratives ou procédurales représentées dans un même format. Ce modèle permet de décrire un dispositif du point de vue de son utilisation, d'élaborer un modèle d'utilisateur ou encore de formaliser l'apprentissage en montrant comment l'acquisition de procédures entraîne la reconceptualisation des objets et, inversement, comment la conceptualisation des objets peut faire inférer de nouvelles procédures valides ou non. Enfin, nous montrons comment Procope est utilisé pour amoindrir la complexité cognitive des interfaces.

Abstract

Using the Procope formalism of description of procedures (Poitrenaud, in press) and by taking as an example an R.N.I.S. telephone terminal (with digital transmission of telecommunication), we show (i) how to decrease interface cognitive complexity and (ii) how the Procope description of novice's know-how provides a model of the user. In both cases, effective realizations, involving this kind of research in the Cognitive Science of technical systems, require an interdisciplinary collaboration.

1. Laboratoire de Psychologie Cognitive du Taitement de l'Information Symbolique ERS, CNRS 139, Université de Paris-8, 93526 Saint-Denis Cedex 02.

Rev. intern. systémique. 0980-1472 Vol. 11/97/01/ $ 7.00/© Afcet Gauthier-Villars

I. SEMANTIC NETWORKS OF ACTION

In the lineage of computer models that formalize the user's know-how in the form of Production Systems (Card, Moran and Newell, 1983; Kieras and Polson, 1985; Bovair *et al.*, 1990), in the form of action grammar (Reisner, 1977, 1981; Payne and Green; 1983, 1986), and of their conceptual extensions (Tauber, 1988; Payne, 1987; Payne *et al.*, 1990), we have proposed (Poitrenaud *et al.*, 1990; Richard *et al.*, 1993; Tijus and Poitrenaud, 1993; Poitrenaud, in press) a model of the representation of knowledge under the form of Semantic Networks Of Actions (S.N.O.A.) that is based on the equivalence between declarative and procedural knowledge. This model allows representing the two kinds of knowledge that are usually distinguished (for instance in Production Systems) within a same formalism (Procope). Until now, the main applications of Procope have been (i) to describe a device from the point of view of its utilization, (ii) to elaborate a model of the user and (iii) to formalize apprenticeship by showing how the acquisition of procedures entails the reconceptualisation of objects and, conversely, how the conceptualisation of objects can make novices infer new procedures, valid or not.

Our description of knowledge comprises, besides the knowledge on objects, both the procedural aspects concerning the execution of procedures (decomposition into goals and subgoals) and the semantic aspects that are included in the know-how: equivalence of procedures that realize a same goal and that thus constitute a category; equivalence between objects to which the same procedures are applied; the knowledge of alternative procedures from the point of view of their generality/specificity; the knowledge of relationships between procedures and the properties of objects that justify them.

From the simple idea that known procedures are associated to objects, as are declarative properties in classic semantic networks, we describe and formalize know-how by what we have named Semantic Networks Of Actions that are hierarchies of abstract and concrete classes represented by oriented graphs $[G : \langle N, P, R \rangle]$, composed of nodes "$N$" (representing classes of objects), procedures "P", and relations "R" between nodes that allow factorizing procedures and properties that objects have in common in order to determine classes, according to simple and multiple inheritance principles.

This model not only allows simulating (i) the manner in which knowledge on objects and knowledge on procedures are structured for a domain (generalization and specification mechanisms are applied on procedures and properties of objects to produce cognitive categories) but also (ii) the manner in which a given task will be realized by using objects of this domain, according to psychological access to knowledge mechanisms, selection, arrangement and execution of procedures: all functions which are implemented in Procope software [1].

II. METRICS OF SEMANTIC NETWORKS OF ACTION

The metrics derived from the S.N.O.A. formalism provide a measure of the complexity, a measure of the power and a measure of the efficiency for any given device.

Complexity

The more the S.N.O.A. of a device are composed of classes of objects that behave differently and/or the more numerous are the procedures that one can apply to them, the more the device is complex. The complexity of a device is defined simply by the product of the number of classes and the number of procedures: $\mathrm{COMP}_{.\mathrm{dev}} = N_c \times N_p$.

Power

The power of a device corresponds to the sum of the power of each procedure: $\mathrm{POW}_{.\mathrm{dev}} = \Sigma\, \mathrm{POW}_{.\mathrm{proc}}$.

What is the power of a procedure? The greater the number of classes of objects to which a procedure applies, the more powerful it is given the total number of potential classes to which it could be applied.

The applicability of a procedure to classes of objects is defined by its scope or extension, which means by the number of classes to which it applies: $\mathrm{EXT}_{.\mathrm{proc}} = N_{c.\mathrm{proc}}$.

The power of a procedure reflects the relationship between its extension and the total number of classes of the device: $\mathrm{POW}_{.\mathrm{proc}} = \mathrm{EXT}_{.\mathrm{proc}}/N_c$.

Efficiency

A device is all the more "efficient" that its complexity is compensated by its power. We express this efficiency by the relationship between power and complexity. The efficiency of a device, $\mathrm{EFF}_{.\mathrm{dev}} = \mathrm{POW}_{.\mathrm{dev}}/\mathrm{COMP}_{.\mathrm{dev}}$

The value of efficiency of the Semantics Network of Actions of a device reflects its cognitive complexity: the higher is the efficiency value of a device, the lesser is the cognitive cost linked to its utilization...

We are going to show successively, using these metrics derived from the S.N.O.A.,

– how we conceive ergonomic interfaces,

– how we evaluate the cognitive complexity of existing interfaces, and

– how we elaborate a user's virtual model that is both predictive of errors and predictive of learning difficulties.

III. CONCEPTION AND EVALUATION OF INTERFACES

In short, the cognitive efficiency of an interface is reflected by the value of efficiency of the S.N.O.A. that represents it. The method of conception consists essentially in reducing the number of classes, to spread the scope of procedures, while having specific alternative procedures for specific classes.

Suppose the conception of the interface of a new telephone terminal that allows memorizing telephone numbers and calling a correspondant, either by typing the call number, or by using the content of the memory that contains this call number. One can conceive, (i) for the class of call numbers that are not memorized, the usual procedure that consists of composing a call number by typing the digits, and (ii) for the class of call numbers in memory, the procedure that consists of searching a call number in memory, displaying it, then validating it (the call of this number). This is described in the semantic network shown in Figure 1. Note that, designing the interface that way, one has no general procedures to call, but two distinct procedures:

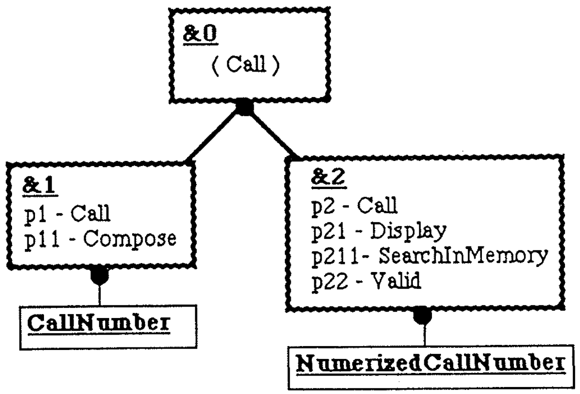

Figure 1. The Semantic Network corresponding to the user's interface of a new telephone terminal without generalization of procedures. The value of complexity is 18, the value of efficiency is .11.

p1-To call by composing[CallNumber]

p2-To call by displaying[numerizedCallNumber].

The application of our metrics provides a complexity value of 18 (3 classes × 6 procedures). The scope of each of the six procedures being 1 (1 class), their power is of .33 each, the total power of this version of the interface is 2 and its efficiency is of .11.

The description of the user's interface in the form of Semantic Networks of Actions allows to envisage procedures that we can generalize. For instance, instead of having two distinct procedures for calling, we can try instead to have only one: To call correspondants (with memorized call number or without): Display[Call Number] and Validate[Call Number]. This is only for the display of the number that one must take into account its type:

To display a non-memorized Call number: compose[CallNumber]

To display a memorized Call number: FindInMemory [MemorizedCall-Number].

The corresponding semantic Network is given in Figure 2. The complexity of this version of the interface is lesser (15); its power (3.66) and its efficiency (.25) have increased.

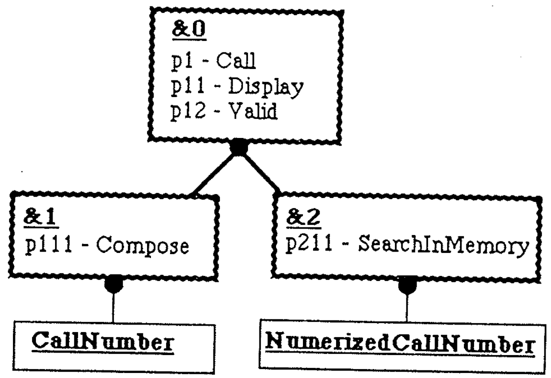

Figure 2. The semantic network corresponding to the user's interface of the telephone terminal of the Figure 1 with generalization of procedures. The value of complexity is 15, the value of efficiency of .25.

The designer of the telephonic device could have envisaged two types of memory for two types of memorized call numbers: call numbers that the user enters himself (typical personal repertory) and call numbers that are recorded automatically in a memory of the "appellants to which the user has not replied". The adding of new procedures complexifying the interface, the ideal would be not to increase the number of classes and, for that, the new memory of appellants call number must behave like the memory of the personal repertory.

To say that the two memories must similarly behave means that the user should find a single call number memory (in such a way, he will not have to enter in its personal repertory the call number of the "appellants to which he has not replied", if he wants to). If this ideal is not technically possible, it is necessary at least to provide a common procedure of items selection in the two memories:

To display[MemorizedCallNumber]: Select Memory of [MemorizedCall-Number],

then FindInMemory[MemorizedCallNumber].

Doing this here, we have added one procedure, but we have not increased the number of classes since the procedure of selection is common to the two memories.

Nevertheless, for our example, suppose that the system's designer has created two distinct procedures to FindInMemory[MemorizedCallNumber]. When it concerns the personal repertory, having seized the name of the correspondant and knowing that its call number is in the personal repertory memory, it suffices to type the first letters of its name to find the number. When it concerns the "appellants to which he has not replied", it is necessary to visualize the items in the list in order to choose the "appellant to call back".

There is here the adding of one procedure, but an increase of the number of classes since the search procedure is not common to the two memories. Figure 3 gives the corresponding semantic network.

One notices here that the addition of one procedure increases the complexity of the system not only because there is one supplementary procedure, but also because there is a supplementary class of memorized call number, and furthermore especially because there is a supplementary abstract (& 1). This class is the category of call number memories. There exists two memories, but they are similar from the point of view of their item procedure of selection. We postulate that shared procedures account for the fact that the two memories, although distinct, belong to a common category. Here, the complexity of the interface is raised (35), the power is 4.2 and its efficiency is decreased to .12.

One sees that the attribution of a new procedural property to objects can entail more or less important modifications of the structure of the device.

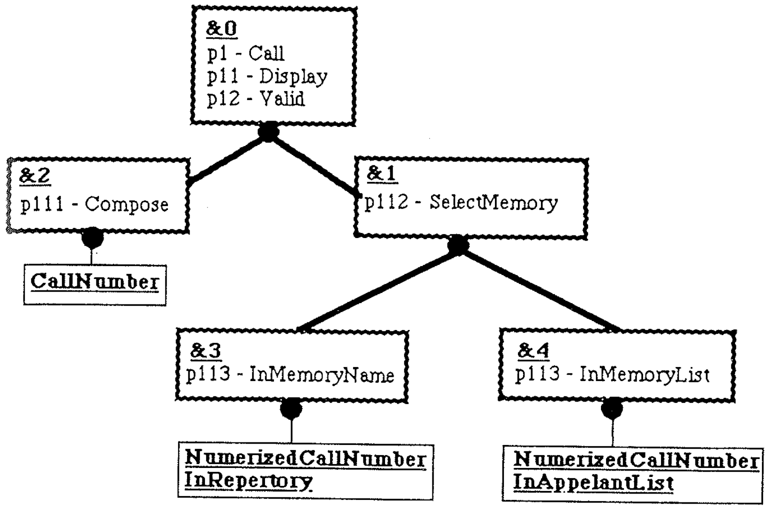

Figure 3. The semantic network of Figure 2 after adjunction of procedure p113-ListInMemory. This adding complexifies the device with two supplementary classes: & 1 that reflects what the memories have in common, & 4 that differentiates the call number of appellants in memory from call numbers in personal repertory. The value of complexity is 35, the value of efficiency is of .12.

These modifications are effects of semantic links that this new property maintains with pre-existing objects and properties. In fact, when one modifies the list of properties of a class of objects, by adding or withdrawing a property, one obtains one of the three following results which are not predicted by the fact that this property is added or suppressed: (i) the interface becomes more complex, (ii) the interface becomes less complex, (iii) the interface remains unchanged in its structure.

This new interface (Figure 3) is not very satisfactory. According to the principle of extension of the scope of procedures, we notice that if the search in memory by typing the name of the correspondant can not be done for the appellants who are unknown. However, we notice that search in a list could be possible for the correspondants in personal repertory. The extension to items in the personal memory of the procedure "ListInMemory[NumerizedCallNumber]" provides the following network (Figure 4).

This extension simplifies the system: the category of call number memories is the category for which there is a memory selection procedure and search for the call number in a list. These procedures apply on call numbers in Memory of appellants (NumerizedCallNumbers in AppelantList) as well as

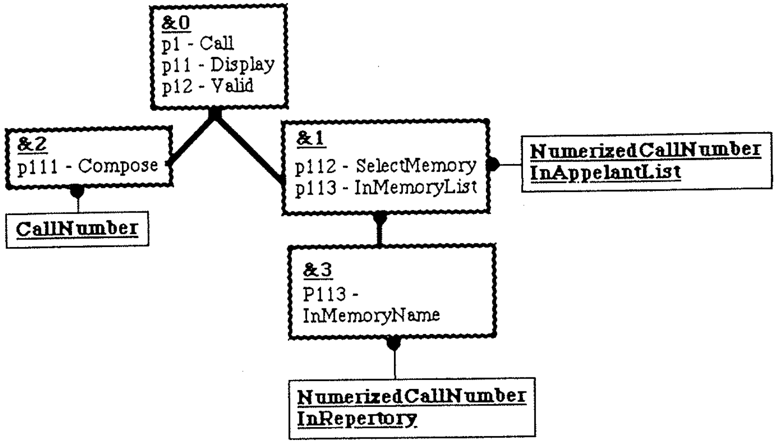

Figure 4. The semantic system of Figure 3 following the generalization of the procedure p113-MemList of the class of call numbers in personal memory (InPersMem). The value of complexity is decreased (28), the value of efficiency is increased (.16).

on call numbers in personal repertory (NumerizedCallNumber InRepertory). Nevertheless, the category of call numbers in memory of appellants is a kind of prototype of call numbers in memory (because directly fastened to this class); the category of call numbers in the personal repertory being categorized in a more specific class in such a way that we get, in addition, an alternative search procedure by typing the name of the correspondant which stands in the personal repertory.

At a first look, it could appear surprising that to have attributed a new procedure to objects brings about a simplification of the use of the device and an increase of the value of efficiency (and that to suppress a property drives to the reverse: a complexification). This point is essential: the categorical allocation rests on the equivalence or on the difference of procedures to apply, so that, fundamentally, the number of classes (and the complexity of a system) depends on the scope of procedures.

This real R.N.I.S. telephonic terminal device exists with two supplementary call number memories: the memory of the last called number (bis/repeat) and the memory of one special number stocked in order to be called without search (direct line).

For the number last called, the calling procedure consists of pressing "bis" (to display the number) then validating the call; for the direct line call number, it suffices to press "valid" (the number of the direct line is

called automatically). This interface, more complete, is described through the network of Figure 5. One notes that there is no common procedure for calling and that the interface is complex ($COMP._{dev} = 70$) and has a weak performance ($PERF._{dev} = .05$).

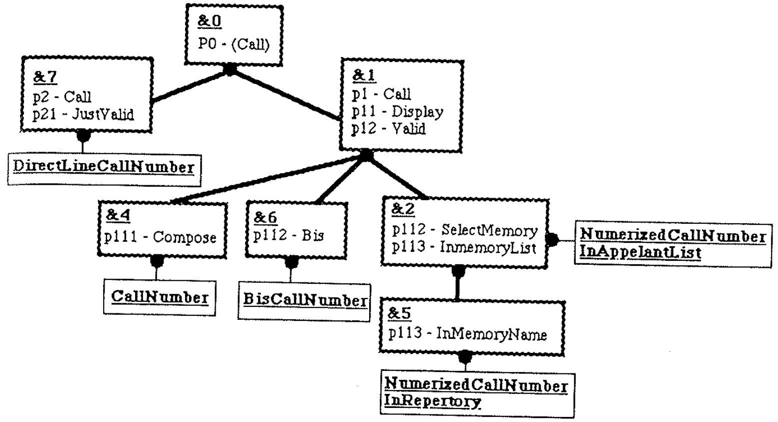

Figure 5. The semantic network of the telephone terminal with all its memories. The value of complexity is equal to 70, the value of efficiency is equal to .05.

The generalization of procedures so as to reduce the cognitive complexity of the interface consists here (i) of attributing to the call number that one calls by the "bis/repeat" function, the status of call number stocked in memory; the procedure of selecting this memory having the result of displaying this "bis/repeat" stocked number, and (ii) of attributing to the direct line call number the status of an instance of the personal repertory, the procedure of memorizing the direct line call number as an instance providing, in addition to procedures associated to the call numbers in the personal repertory, a specific and automatic calling procedure. Note that this option simplifies the task of memorizing call numbers. If the call number of the direct line (that could change frequently) is also a call number that figures in the repertory, it has to be typed twice in the old version. In this new option, the choice of what call number will be the direct line call number corresponds to a choice among correspondants in the personal repertory.

The semantic network corresponding to this new version of the interface is given in Figure 6. The value of the complexity decreases here from 70 to 54, the value of efficiency increases from .05 up to .09.

We have realized the computer simulation of the functioning of the two interfaces corresponding to Figures 5 and 6 and we have made an

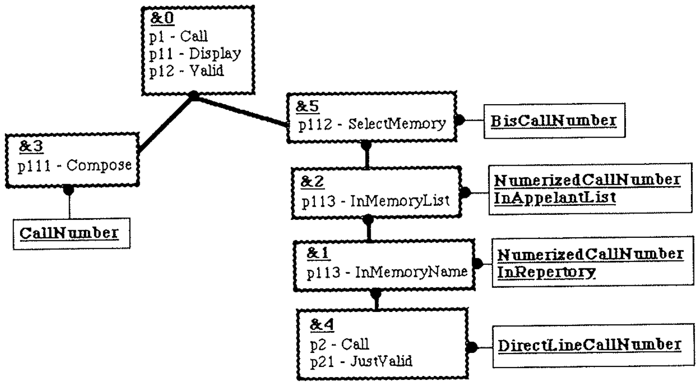

Figure 6. The simplification of the semantic network of the telephone terminal with all its memories. The value of complexity is equal to 54, the value of efficiency to .09.

experimental comparison of the utilization of these two versions, observing novices with instruction manuals. Results (errors and duration rate of task realization) are clearly in favor of the second version (Poitrenaud, Richard and Tijus, 1990; Tijus and Poitrenaud, 1993).

When we test the recommendations that derive from the application of this method for the conception and the evaluation of interfaces, or for the conception of instruction manuals (Barcenilla *et al.*, 1992; unpublished research of our laboratory), similar results are found with others devices such as

- VCR,
- French National Telephone Network System Supervisor,
- Air Conditioning Machines,
- Data-Processing Software,
- Drawing Software.

IV. THE MODELIZATION OF THE USER

If a user's knowledge (objects and procedures) is comprised the knowledge that is required to use the system, the Semantic Network Of Action that describes the interface of the system and the Semantic Network Of Action that describes knowledge of this user are identical. We call this network the "Ideal User Network". Procope Semantic Networks, that are used to describe the interface of systems from the point of view of their correct utilization, for purposes of conception and evaluation, are similarly used to describe the knowledge any novice has about the utilization of a device. We represent thus, not only the knowledge of experts, but also the knowledge of novices, that comprises both incomplete and erroneous procedures.

For any device interface, the construction of the S.N.O.A. of a user takes place in two steps. First, from a computer software that simulates the functioning of the device, we collect procedures applied by this user on objects of the interface. Second, with our software Procope, we build automatically the corresponding Network and we simulate the execution of tasks. Doing so,

1. we construct the Semantic Network Of Action that corresponds to the knowledge of common novice users. This provides a model of the user (User's Virtual Model),

2. we predict common errors as well as learning difficulties by matching the S.N.O.A. of the user (User's Virtual Model) with the S.N.O.A. of the device (or Ideal User Network) and by adding to the former, the analogical transfer of procedures known from other sources (other reference devices); learning difficulties being a function of the distance between the manner in which objects are structured in the two networks,

3. we predict and we describe the course of learning by showing the successive transformations of the network of the user; learning to use a new device consisting of elaborating successive semantic networks that are more and more compatible with the functioning (objects and procedures) of the device.

Finally, one of the next applications of our work, is the construction of an artificial and intelligent guide, in order to provide on-line instructions and learning in the course of the utilization. By automatically comparing the two networks (the User's Virtual Model and the Ideal User Network), the intelligent guide system would have to diagnose errors, to teach the novice and especially to be able to provide useful on-line assistance.

V. CONCLUSION

The Procope formalism allows describing knowledge about the functioning and knowledge about the utilization (Richard, 1983), since we describe both the device and the user from the point of view of

- procedure knowledge,
- knowledge about classes of objects of the devices, and

– knowledge about the relational properties that justify, from the point of view of the user, the functioning of the system during task execution (application of procedures).

The conception, the evaluation and the improvement of the performance of interfaces of technical systems, such that they derive from our method, do not consist simply in providing a new design for the appearance of the device. Suppression, adjunction and generalization of procedures are interventions that bear on the workings of the system. To realize such interventions, the co-operation between interface designers and technical functioning designers is crucial. This co-operation has to intervene very early in the conception of new technical systems and has to be effective when it concerns the ergonomic improvement of existing systems.

When one desires to integrate, as it is our case, a Dynamic User's Virtual Model directly into the interface, in order to favor efficient utilization and learning, one is confronted with the real functioning of the technical device. This aspect, such that it is accessible within the technical manual and the user's manual, is the way the device works for tasks that are correctly executed. A dynamic and interactive model has to take into account the effects that are produced, in the device, by users' unusual actions, and the manner to remedy these. We need here collaboration with automaticians that could conceive a model of the device that interacts with the User's Virtual Model; each providing inputs to the other and collecting outputs for the other (actions of the user, replies of the system; actions-presentations of the system, replies of the user).

Note and references

1. The language "Procope" is a language of specification of classe inclusion Networks supporting multiple inheritance. A class is defined by a set of superordonate classes, a set of attributes of objects of the class, a set of goals and procedures to reach these goals on objects of the class. Each attribute is provided a type that is itself a class of one of the networks. Several procedures can be defined to reach a same goal on a same class of objects. These alternative procedures are tidied and this order reflects the accessibility of the information: a specific procedure is always more accessible than a general procedure. Finally, the mechanism of execution made by an interpreter associates an approach "oriented object" and an non-deterministic approach. When a goal is released on a type of objects, the interpreter chooses the most accessible procedure to reach this goal without examining possible alternatives. In case of failure, it abandons its choice (by backtracking) and searches for another solution.

J. BARCENILLA, C. A. TIJUS, J. F. RICHARD and S. POITRENAUD, How to do it? Description and simulation of skill within semantic of action networks. XXV International Congress of Psychology, Brussels, July 1992, pp. 19-24.

S. BOVAIR, D. E. KIERAS and P. POLSON, The acquisition and performance of text-editing skill: a cognitive complexity analysis. Human Computer Interaction, 1990, *5*, pp. 1-48.

S. K. CARD, T. P. MORAN and A. NEWELL, The Psychology of Human-Computer Interaction. Hillsdale, N. J.: Erlbaum, 1983.

D. E. KIERAS and P. POLSON, An approach to the formal analysis of user complexity. Interaction Journal of Man-Machine Studies, 1985, *22*, (4), pp. 365-394.

S. J. PAYNE, Complex problem spaces: modelling the knowledge needed to use interactive devices. INTERACT'87, B. SHACKEL Ed., Amsterdam: Elsevier, 1987.

S. J. PAYNE, H. R. SQUIBB and A. HOWES, The nature of device models: the yoked state space hypothesis and some experiments with test editors. Human Computer Interaction, 1990, *5*, pp. 415-444.

S. J. PAYNE and T. R. G. GREEN, The user's perception of the interaction language: a two-level model. 'CHI 83' Proceedings conf. on Human Factors in Computing Systems. New York: ACM, 1983.

S. J. PAYNE, and T. R. G. GREEN, Task action grammars: a model of the mental representation of task languages, Human Computer Interaction, 1986, *2*, pp. 93-133.

S. POITRENAUD (in press). The Procope Semantic Network: An Alternative To Action Grammars. International Journal of Human-Computer Studies.

S. POITRENAUD, J. F. RICHARD and C. A. TIJUS, An object-oriented semantic description of procedures for evaluation of interfaces. Proceeding of the 5th European Conference On Cognitive Ergonomics EEÇE. Urbino, septembre 1990.

P. REISNER, Use of psychological experimentation as an aid to development of a query language. IEEE Trans. Software Engineering, 1977, SE-3, pp. 218-229.

P. REISNER, Formal grammar and human factors design of an interactive graphics system, IEEE Trans. Software Engineering, 1981, SE-7, (2), pp. 229-240.

J. F. RICHARD, Logique du fonctionnement et logique de l'utilisation. Rapport de recherche 202, INRIA, Rocquencourt, 1983.

J. F. RICHARD, S. POITRENAUD and C. TIJUS, Problem-solving restructuration: elimination of implicit constraints. Cognitive Science, 1993, Vol. 17, pp. 497-529.

M. TAUBER, On mental models and the user interface in human-computer interaction in Working with Computers: Theory versus Outcome; G. C. Van DER VEER, T. R. G. GREEN, J. M. HOC and J. M. MURRAY Eds., Londres Academic Press, 1988.

C. A. TIJUS and S. POITRENAUD, Semantic Networks of Action. In: Stella Vosniasdou Eds., Psychological and Educational Foundations of Technology-Based Learning Environments. Springer-Verlag, 1993.