Int. J. General Systems
1976, Vol. 3, pp. 3–23

PROTECTION AS A GENERAL SYSTEMS PROBLEM

LADISLAV J. KOHOUT
University College Hospital Medical School, University of London, U.K.

BRIAN R. GAINES
Man-Machine Systems Laboratory, Department of Electrical Engineering Science, University of Essex, Colchester, U.K.

(*Received April 15, 1975; in final form October 29, 1975*)

It is argued that the problem of *protection*, of controlling mutual access rights to shared resources, is a topic appropriately treated as a major component of general systems theory. Although most widely studied and developed in the context of computer systems, protection models are equally applicable to biological systems, such as those involved in movement control. The paper first establishes the nature of the problem of protection in computer systems, noting that it only reaches its full potential complexity in large data-base systems with processes automatically invoked by "data interrupts". The Graham and Denning model of protection and the concept of a "capability" are then described and the appropriate mathematical tools for the analysis of such models discussed. A detailed model of protection is then developed with examples of the role of algebraic, automata-theoretic, topological and modal/multi-valued logical, techniques in its analysis. Finally, biological applications, general systems consequences and automatic design techniques, for protection structures are discussed.

INDEX TERMS Multi-user computer systems, hardware and software protection, biological systems, muscle and movement control, operating systems, data bases, capabilities, automata theory, modal logics, many-valued logics, topology.

1 INTRODUCTION

It is an essential objective of general systems theory to develop problem specifications and constructs that are common to many diverse areas of knowledge yet also capable of deep theoretical development. The power of such system-theoretic notions as identification, stability, and control lies in their being both sufficiently general to be widely applicable and also sufficiently well-defined for a substantial body of theory to be developed independent of the application. In this paper we propose that the concept of *protection*, made explicit in recent years by studies of *protection structures*[1] in computer systems, satisfies both these criteria for a powerful general systems construct.

The overall paper presents an exposition of the role of various mathematical techniques in the development and utilization of protection structures in general systems with particular emphasis on practical applications to two quite distinct system topics: firstly, that of computer systems where the concept first arose,[2] and secondly, that of human processes of adaption, particularly to neuromuscular disabilities, in which remarkably similar concepts of resource allocation, transmission and access also naturally arise.[3] In the broad context of general systems, protection structures represent necessary constraints on sub-systems in order to maintain the basic functions of the system as a whole and prevent them being impaired by the independent and uncoordinated activities of its individual parts. On one hand we are concerned to present the problems of developing and analysing such structures as a new systems area, similar in status to such areas as identification, stability and control, and worthy of the attention of theorists. On the other hand we are concerned to investigate the nature and magnitude of practical requirements for, and the current implementation of, protection structures to ensure that theoretical developments have a proper and useful semantics.

1.1 *The Nature of the Problem of Protection*

A systems theorist who attempts to apply his general principles to a specific applications area will almost invariably find that he is confronted with,

not a homogeneous system, but rather a "heap" of, at first appearance, chaotically connected subsystems interacting in an ill-defined manner. The relationship between subsystems, or between traits of their behaviour, is rarely simple or uniform. In general it is not the null relation, that no behavioural trait should interact with any other. Neither is it the simple order relation that each behavioural trait has greater precedence than some and less than the others. Such simple constraints are often overall objectives in the design and implementation of artificial systems but the general case is one of multiple, interrelated, mutually interacting behavioural traits and activities. In terms of resource allocation, subsystems need not be excluded from access to one another's resources, nor given a simple static priority of access, but may be allowed controlled access to mutual resources. In these circumstances the possible relationships between subsystem activities become indefinitely complex. Furthermore, the mutual relationship between subsystems and their activities need not be established once and for all, but can itself be a dynamic entity subjected to (constrained) modification from moment to moment. A static evaluation of the protection structure itself is then no longer possible and the management of resource allocation and its protection takes on a further dimension of complexity.

The problem is best illustrated by some practical examples:

a) The manager of a complex of geographically separate but interconnected multiple-processor configurations with international access through telephone and data networks in which users can form mutually interacting communities and establish their own sub-operating systems recursively mimicing that of the main system, has management information and resource control requirements which must eventually exceed those of other national and international utilities such as power generation and even government (which presumably will come to depend increasingly on such systems anyway[4]). The co-ordinating users of such systems also need to be able to finely tune their own systems of resource allocation and protection to ensure integrity of information and service at levels for which they are responsible. Even comparatively small dedicated systems for process or telecommunication control generate this problem since their development continues into their operational lifetime and it is necessary to be able to test new or modified facilities in a well-defined environment such that the possible consequences of errors in them are known and limited.

b) The brain co-ordinating the dynamic activity of various groups of muscle in a complex movement task must ensure proper resource allocation as well as eliminate improper simultaneous action of certain muscles in order to achieve harmonious co-ordination of movement. Any failure in this respect results in serious movement disorders.[5]

The studies reported arose from, and were motivated by, the coincidence of two apparently unrelated activities: on the one hand, by experience in the design of a descriptor-organized minicomputer[6,7] in which the full power of hardware-enforced protection ring crossing processes may be invoked by procedure calls in high-level languages;[8] and, on the other hand, by studies of muscle and movement control, of neuromuscular movement disabilities[9] and by attempts to model human adaption[10] and training[11] and the representations of movement in the brain.[3] Gradually, we have come to realise that both areas have many aspects in common, which could be eventually formulated as general systems concepts.

Whilst protection may be seen to have arisen as a distinct and significant problem largely through its practical applications, it also has a theoretical richness that makes even its abstract formulation of great interest. The natural logics of protection are not the Boolean algebras so basic to digital computers, but rather the modal logics[12–14] of possibility and necessity (alethic), permission and obligation (deontic), etc. For example, we typically wish to know whether it is *possible* for a subsystem (process), which is *permitted* to access another subsystem (resource, data structure) but *obliged* to obey certain synchronization disciplines in accessing it, to avoid these, or whether they are *necessarily* obeyed. Modal logics have been studied mainly in a philosophical context as models of linguistic inference,[15] or ethical structures,[16] and the problem of protection offers a new semantic domain for the logical theory. An important aspect of the theory of these logics is its close links with algebra[17,18], topology[19] and automata theory,[20] all of which are especially significant in the study-of the dynamics of protection structures.

1.2 *Organisation of Paper*

In presenting this material we face two problems; firstly, the idea of protection structures has not yet appeared in the abstract context of general systems,

so that proper motivation and applicability to particular specialised systems area should be demonstrated, in order to make sure that the problem actually exists and that we are not finally solely engaged in grappling with our own terminological obscurity. Secondly, topological, modal, and multi-valued logic techniques are not very well known outside their own specialised areas. For these reasons, we feel, it is appropriate to give an exposition of the problem together with the motivation in a particular applications area and only then to present a more abstract discussion of the problem in a general systems theoretical setting. Hence, we shall present the problem of protection first in the context of computing only, determining precisely how it arises and what aspects of it give rise to practical and theoretical complexities; then we shall briefly review the mathematical techniques available to deal with these problems leading to the basic Graham and Denning[21] model of protection structures; this formal model of protection will then be developed in its own right with the coherence of the various theoretical approaches demonstrated in relationship to a simple example used throughout these sections; then we will proceed with illustration of similar problems, this time from the biologically motivated area of control of muscles and movement with a stronger emphasis on the hierarchy of systems; finally we shall present the problem as a problem in general systems area with discussion of its relevance for analysis and synthesis of multi-level systems.

2 THE PROBLEM OF PROTECTION IN COMPUTER SYSTEMS

The protection of the security of potentially shared resources, both information and activities, has become a problem of major interest in computer science and engineering. Fundamentally, the problem is not different from those of personal, commercial and government security in the pre-computer era—the differences are quantitative ones of monitoring electronic activities whose speed, magnitude and inaccessibility far exceed the human transactions they mimic. Technically, aspects of security peculiar to computer-based systems may be seen to arise with the early time-sharing systems such as CTSS and MAC[22] which broke away from batch-processing of naturally isolated jobs and allowed users to share not only basic resources like storage and processing power but also to access joint data bases and processes for mutual interaction in real time. It was the announcement of the MULTICS[23] project in 1964, particularly the discussion of its aims and objectives in a group of 6 papers at the 1965 FJCC,[24] that awoke the computer community at large to the new technical problems, as well as the new potentialities, of systems accessed simultaneously by multiple, competing and collaborating, users. Even at this early stage the *social implications* of such systems were discussed[25] and these have become a matter of increasing public concern in recent years.[26,27]

Thus *protection* in computer-based systems has arisen as an important and distinct problem closely associated with many of the technical problems of operating systems, e.g., ensuring the correct functioning of co-operating sequential processes,[28] but having an identity of its own independent of the particular means by which it is enforced. Equally, the availability of adequate protection structures is itself a prerequisite to the full exploitation of modern techniques for modular,[29] or structured[30] programming.

The place of more complex analyses of protection in computing is discussed in the following section. It is inappropriate in this paper to attempt to survey all contributions to the protection literature, and we refer the reader to the comprehensive recent review by Popek[1] which lists some 84 references.

2.1 *Is there a Problem?*

Before any theorist moves in with an armoury of mathematical techniques it behoves him to ensure that the enemy actually exists and that he is not finally solely engaged in problems of his own making. Any computer manager will confirm that his installation has a security problem. However his anecdotal reports are more likely to demonstrate human errors, software bugs and design faults, rather than any deep and elaborate failures. His problem is still security in the negative sense of containment, and the hardware mechanisms of most commonly used macines are designed with this in mind.

Even MULTICS, with its objectives of supporting collaborative user communities, is based on a simple linear order of protection rings of monotonically decreasing capability which it is simple to express logically. It allows users to share, or not to share, major data objects but does not realistically support more subtle interactions between them. The wide use of computer systems with far less

complex protection facilities than MULTICS is evidence that a substantial part of the user community can get by without such subtlety for their current activities. This does not prevent them being adversely affected when manufacturers attempt to incorporate it, unsuccessfully, in their operating systems, but it indicates that we have to search with care for the positive requirements.

2.2 *Capabilities and the Graham and Denning Model*

A key paper in expressing these positive requirements and mechanisms for their satisfaction is that by Lampson[31] who introduces the term *capability* for the access right that a process may possess to an *object*, a generalized resource. Capabilities are themselves protected objects which may be created and passed between objects only according to prescribed rules. Graham and Denning[21] make explicit appropriate rules for the manipulation of capabilities in a second key paper. It is important to note that although these papers have abstracted the protection problem with a high degree of generality, the exemplary semantics given is still very simple (in terms of capabilities to read and write into files) and many basic problems are deliberately excluded (for example, access to data being dependent on its value). The concluding paragraph[21] (p. 428) is particularly important in summarizing the state of the art.

The implementation and use of capabilities has now been investigated in depth,[32] hardware realizations of capability-based protection structures are being developed,[33] and at least one commercial machine is now in production,[34] although Lampson has emphasized the extreme practical difficulties of attaining total security.[35] However, whilst the technical problems of implementing such systems are being overcome, from past experience the tools for managing them will lag way behind their implementation and use. In discussing the rules for dynamic protection control developed by Lampson and Graham and Denning, Popek[1] remarks, "Because of the complexity of those rules, it is not clear what changes in the data are possible by interactive application of the rules". Yet this is clearly an essential requirement for system management, to know *all* the implications of any particular act of protected resource allocation.

Thus, whilst current emphasis is on the *proving* of operating systems from the designers point of view,[36] future emphasis will tend to be on proving properties for the suspicious user within the protection sub-environment he creates, and hence investigation of the Graham and Denning model is important in its own right.[37] There is firstly the forward problem of determining the total implication of a particular protection structure, and, secondly, the reverse problem of determining what structure will lead to a desired set of final relationships. Here we have a problem-solving requirement: given a set of relationships that must hold (the required relation); a set that must not (the violation relation); and with all other relationships being permissible but not obligatory (the don't care relation); what possible protection structures satisfy the total implied relation? In general there will be many answers that can be winnowed down by further constraints, such as maximize the don't care relation (if it doesn't matter let it happen—the permissive computer society!). Ultimately there will be a number of alternatives, of which zero and one have obvious implications, and greater numbers require arbitrary choice or selection based on other criteria.

Whilst such problems clearly require theoretical study and computational aids, however, it is doubtful that the requirements of current protection hardware to support even advanced operating systems provides adequate motivation for their study. These systems are still designed with the coarse protection of large-scale resources in mind, files rather than individual data items, jobs rather than individual user activities, and are more concerned with prevention of conflict than with support of cooperation. They do not yet provide a rich enough semantics to test any theoretical approach to the problem of protection. However, such a semantics is beginning to develop in the development of large *data-base*[38,39] information systems.

2.3 *Data-Bases and Data-Interrupts*

A key paper on data-base protection is that by Conway, Maxwell and Morgan[40] who consider security requirements in practical information systems such as personnel records. Here the units which must be protected are far smaller than those previously considered, being individual fields in a single record rather than complete files of information. Equally importantly the rights to access certain fields may be dependent on the data stored in these or other fields of the record. Thus a typical protection predicate might be: "an assistant manager may read the personnel records, except

medical history, of employees in his division with salaries of less than $30,000". Languages have been developed which support protection of data structures and items at this level of detail.[41,42] They provide far richer and more complex examples of protection predicates and possibilities than does the literature on operating systems.

What these examples lack, however, is the dynamic complexity of operating systems in which the protected objects are not only passive data items but also active processes which themselves initiate further activities and accesses to protected items. This may be introduced into the data-base problem by considering a suggestion of Morgan[43] of "an interrupt based organization for management information systems" in which a predicate on the values of data items may be used to invoke a process. For example, an inventory control system might have processes attached to variables indicating stock levels that automatically re-order items if the stock falls below a prescribed level. Zelkowitz[44] has suggested a hardware implementation of this mechanism on the IBM 360 and it is feasible with any tagged[45] or descriptor-based[6] machine in which the tags are retained in file structures.

Examples of data-interrupts in use are currently probably found only in such "artificial intelligence" languages as PLANNER[46,47] and CONNIVER,[48] and are central to the "actor" based semantics of recent developments of these languages in PLASMA.[49,50] However, the use of "data-base-driven" processes is very much in line with concepts of modular programming[29] since they allow an activity dependent upon the value of a variable to be implemented as a single independent module rather than incorporated as conditional calls in every routine that may update that variable. They have a natural place in languages such as POP2[51] and EL1[42] which allow separate "selecter" and "updater" routines to be associated with an individual variable. Their availability is particularly attractive in quite simple transaction-processing systems where on-line users access the same data-base, e.g., dealing systems,[52] since all activities naturally centre around, and are driven by, the state of the data-base. Whilst the hardware necessary to implement the data-interrupt is comparatively new, we have reported elsewhere[53] the practical success of commercial and medical transaction-processing systems based on the interpretation of a high-level language on a minicomputer, and are currently extending the facilities to include data-interrupts, a simple extension to an interpretive language.

2.4 *Summary*

Thus a combination of the finely detailed, data-dependent protection requirements of data-base systems together with the dynamic protection requirements of data-interrupt driven systems provides a far richer semantics for models of protection than does the conventional "operating-system" requirements, and one that is both generated by current needs and is feasible in many applications with current hardware/software technology. The potential of such systems is well beyond our current intuitive conceptions of what computer systems can do. The possibility of adding arbitrary distinct processes, "unknown" to one another but mutually interacting through changes in state of a common data-base, allows a far more natural development of a system, based on mimicing the activities of individuals in an organization. Equally such a system may grow rapidly beyond the comprehension of its designers since the addition of a new activity may invoke a host of natural side-effects which have no referents whatsoever in the new activity itself. The problem of ensuring adequate security whilst at the same time taking full advantage of the mutual collaboration possible will become acute.

3 THE MATHEMATICS OF PROTECTION

3.1 *The Roles of Different Formal Models*

The natural representation of a protection structure relating processes to capabilities, adopted for example in both our key references,[21,40] is that of a *matrix* expressing the (algebraic) relation between them. Such relations, expressed as matrices, can also model the dynamics of protection, the permission to pass a capability to another process, etc. The overall model obtained is naturally *automata-theoretic* with its analytic basis being *algebraic*. The algebraic model itself has a direct application to questions about procedures to follow in attaining certain *aims*. "How do I write into file A", is answered by enumerating trajectories of communication through processes which do not violate the protection. There may be none (not allowed), a unique solution, or many possibilities with different properties. This corresponds to a *control* problem in the state space of the protection automaton.

However, many of the major questions of security are not of this nature but relate more to global

properties of reachability, "can any of these processes *access* this information", "is this process *contained* in this domain". Such questions are naturally ones of *closure*[19] and best treated within a topological framework. They may be seen as *stability* problems in the state space of the protection *automaton*. The actual closure spaces generated by any particular protection structure should reflect the intentions of users in setting it up. There are direct formal relations between such spaces and modal logics[54,55] so that the semantics of the model may be expressed in a communicable form. It is easier to understand, "it is desirable to do X and it is permissible to do Y but the system will not allow you to do Z", or, more globally, "the protection system of the HCN471 will not allow the user to follow this desirable practice and is dependent upon him obeying these rules", rather than "$X \in S_a(U)$, $Z \in S - S_c(U)$", or, "the HCN471 has no compatible closure relation".

In pratice although both topological and modal logic techniques and vocabularies are useful, any real protection structure will be finite and users will tend to superimpose on it a readily understood structure of nested protection domains. The many-valued logics thus generated may be formally regarded as finite approximations to modal logics,[54] and are an alternative natural expression of hierarchical, ordered structures (e.g., protection rings).

From a category-theoretic point of view[56–58] these distinctions are purely ones of terminology and perhaps the ultimate abstraction of protection structures should be expressed categorically. However, although the old lines of demarcation no longer exist, the old terminologies are still evocative and what is clumsily expressed in one may become quite elegant and transparent in another. Thus, in summary, we see the appropriate use of mathematical tools in the study of protection to be:

Algebraic formulation of protection axioms → topological formulation of closure properties → modal logics of resultant spaces → multi-valued logic representation in finite matrices.

3.2 *The Graham and Denning Model*

As noted in section 2.2 the best developed formal model of protection is that presented in[21], and we have based our analysis in the following section upon this. Briefly, Graham and Denning distinguished "subjects" which are active entities (a process and domain of access to resources) and "objects" which are essentially resources to which access must be controlled—a "subject" is also an "object". They represent a protection structure as a matrix of subjects against objects giving the access rights of each subject to the objects (including other subjects), together with a set of rules for changing the matrix (e.g. by adding or deleting subjects and objects).

The elements in the matrix form "capabilities" (an access right by a subject to an object) and the dynamics of the model arise to a large extent because capabilities can be *passed* from subject to subject. It is possible to treat the right to pass a capability (the "copy flag"[21]) itself as a capability and such generality is desirable for theoretical compactness. However, in explaining the model it is useful to separate out the protection matrix from its dynamics and we introduce a *pass* as the right to pass a capability, and a *permit* as the right to give this right—further recursive extension is unnecessary to the example.

One extension we have *not* made in our analysis is to consider relationships and interactions *between* capabilities. In management information systems it is unlikely that the capabilities would be themselves simple, unitary actions. Rather they would reflect the fine structure of possible actions so that a major action, such as writing into a record, would be possible only to the possessor of multiple capabilities. Equally the act of so doing is likely to be necessarily accompanied by other acts, e.g., associated with transaction monitoring. This implies that there will be rather more complex relationship between capabilities and actions than is assumed in any current model, but the extension to allow for this is straightforward. The only remark we make for the moment is that the algebraic structure of interaction between capabilities must be positive[59] (p. 125), i.e., one capability cannot cancel another out. This is implicit in the literature, but it is tempting in extending the models to add "anti-capabilities" (for example to allow a user of a subsystem to ensure that it is "memoryless" by removing its access to certain channels of communication). Non-positive capabilities make nonsense of the use of closures, and do not seem to have a proper place in the semantics of protection.

Two further concepts are necessary which are relevant to the *use* of Graham and Denning's model rather than its structure. Some ("privileged") subjects will have capabilities that would show up as dangerous in any analysis but which they will not use. We introduce an *intention* matrix that specifies what ones will be used. This enables the closures

computed to reflect relationships of *trust* between subjects. In analysing his protection a user would adjust an intention matrix to specify his own use of capabilities (assuming other users have malicious intentions) and a trust matrix to prevent non-significant paths for protection failure being continually drawn to his attention, but both may be represented in the model as a single matrix.

4 ONE FORMAL MODEL OF PROTECTION

4.1 *A Concrete Example*

The terminology of the following sections would be opaque without some concrete examples. Unfortunately examples tend to be either trivial or too lengthy in description. The following artificial situation has been generated to serve as a basis for illustrating each technique discussed.

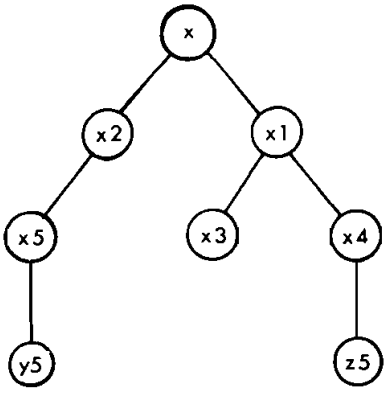

FIGURE 1 Data processing network example.

Start of example. The company X runs a network of data processing systems. The basic flow of information is shown in Figure 1: the system x can directly inspect x1 and x2, and indirectly inspect x3 and x4 or x5 via x1 and x2, respectively. In addition to this fixed hierarchical flow, the systems can exchange information within the network according to certain dynamic relations.

The type of problem we shall study is that there is exchange of information with similar systems operated by competitors: x5 with y5 of company Y and x4 with z5 of company Z. Y and Z must not obtain the information in x, x1 or x2 at the same time, although each part of the information on its own, or combinations at different times (say more than t a part) are harmless. The information flow is fully defined by a sequence of action, pass and permission relations. Computationally these might be represented as (sparse) matrices but for this text we shall work with the relations.

4.2 *Terminology and Definitions*

In our terminology, we shall stress the dynamical character of protection.

Participants—abstract elements of a protection structure, which can be either subjects or objects. The set of all participants will be denoted by $X = \{x_1, x_2, \ldots, x_n\}$.

An object—a participant, manipulation of which must be controlled.

Subject—an active participant whose manipulation must be controlled.

A participant x_j can simultaneously be a subject with respect to the object x_i and an object with respect to the subject x_k.

Action—certain precisely specified behaviour of participants. A subject *acts* on an object, and an object is *manipulated* by a subject. (Examples of action: read, write, seek, execute, etc.)

Activity—a sequence of actions with some unambiguously specified purpose.

Aim—an a priori specified (required) result of a sequence of actions, which form a particular activity. Note that a specific action can enter as a component into the formation of two or several distinct activities.

Aim controllable by a group of subjects X_t—an aim which can be achieved by a sequence of actions exclusively performed by the group X_t.

Aim protectable by a group of subjects X_t—an aim which cannot be achieved by an activity outside X_t without the specific permission of the group X_t.

It is important to realise that a certain specific action can form two or more distinct activities, or can contribute to the fulfilling of two distinct aims. Hence there may exist two different and often contradictory requirements of the protection in a case where the same action is a component of two distinct activities.

Action matrix—for an action α_i is defined by a relation $R_{\alpha i}(x_j, x_k)$ between participants from $\{X\}$.

Capability—a protected name, a pair $\langle \alpha_i, x_j \rangle$ where α_i is an action and x_j is an object. A subject x_k has the capability $\langle \alpha_i, x_j \rangle$ if it can perform the action α_i on the object x_j.

A subject can pass a capability it holds to another subject. This action must be properly controlled. For this purpose we shall introduce:

Pass—a protected name, a pair $\langle \langle \alpha_i, x_j \rangle, x_k \rangle$ where $\langle \alpha_i, x_j \rangle$ specifies the capability and x_k is the subject holding the pass. A pass signifies that a subject x_k is allowed to pass a capability.

Permit—a protected name, a pair, $\langle \langle \alpha_i, x_j \rangle, x_k \rangle$ $\langle \alpha_i, x_j \rangle$ specifies the capability to which the pass refers and x_k is the subject which holds the permit; $\langle \langle \alpha_i, x_j \rangle, x_k \rangle$ signifies that the subject x_k can give the permission to pass the capability $\langle \alpha_i, x_j \rangle$.

4.3 *Algebraic Models*

An abstract algebraic model used for the investigation of the dynamics of protection structures, is formed by relations expressing the mutual dependencies of subjects and objects as well as relationships of capabilities, passes and permits.

The set A of all actions α_i, which are elements of an activity Z_k is denoted by:

$$A(Z_k) = \{\alpha_1, \alpha_2, \ldots, \alpha_\omega\}.$$

The structure of an action α_1 can be described by the triple of relations $R_{\alpha i}$, $R_{\phi i}$, $R_{\pi i}$:

$$R_{\alpha i} = R_{\alpha i}(x_k, x_l)$$
$$R_{\phi i} = R_{\phi i}(x_k, x_j, x_m)$$
$$R_{\pi i} = R_{\pi i}(x_k, x_j, x_m).$$

The relation $R_{\alpha i}$ defines the subject-object relationship and specifies the capabilities of a set of subjects $\{\text{sub}\} \subset \{X\}$ to perform the action α_i on the set of objects $\{\text{ob}\} \subset \{X\}$.

The ternary relation $R_{\phi i}$ specifies which subject x_k can pass the capability $\langle \alpha_i, x_m \rangle$, $x_m \in \{\text{ob}\}$ to a subject x_j. The ternary relation $R_{\pi i}$ specifies which subject x_k can give permission to copy the pass $\langle \langle \alpha_l, x_m \rangle, x_j \rangle$, $x_m \in \{\text{ob}\}$, $x_j \in \{\text{sub}\}$.

Each ternary relation $R_{\phi i}$, $R_{\pi i}$ can be expressed as a set of binary relations:

$$R_{\phi i}(\text{sub}_1, \text{sub}_2, \text{ob}_m) \equiv$$
$$\{R_{\phi i, \text{ob}_1}(\text{sub}_1, \text{sub}_2), R_{\phi i, \text{ob}_2}(\text{sub}_1, \text{sub}_2),$$
$$R_{\phi i, \text{ob}_\alpha}(\text{sub}_1, \text{sub}_2)\}$$

where $m = 1, 2, 3, \ldots \alpha$; sub_1, sub_2, $\text{ob}_m \in \{X\}$. Similar expressions hold for $R_{\pi i}$.

The relations which have been so far described deal with permissions. However, it seems necessary to introduce structures which can describe the intentions of the participants, as well as the permissions. This can be exemplified by the following example. Let us consider the permission which is described by the transfer rule R1 of Graham and Denning. The rule R1 permits a subject to transfer any capability it holds to any other subject, provided the donor has the corresponding pass (which is realized in the scheme as a copy flag). Without the introduction of some further structures we can investigate only the case where the intention of each subject with the appropriate pass is to give capabilities to all subjects. This limit case describes only the minimal restrictions which are enforced by the permission rules but not the actual state of the protection system in the case that the participants do not reach the limits forced by the permission rules. However, this is required by a user who would like to find out how he should pass his capabilities and avoid some unwanted side effects.

Now we shall introduce a formal definition of a model of protection structures. It will be shown later (section 4.6) that the model can be interpreted as a *hierarchy of sequential machines.*

DEFINITION A model $\mathcal{M}(Z_k)$ of an activity Z_k is composed of the set of triples:

$$\mathcal{M}(Z_k) = \langle R_{A(Z_k)}, \Phi_{Z_k}, \Psi_{Z_k} \rangle$$
$$= \{\langle R_{\alpha i}, \Phi_i, \Psi_i \rangle\}_{i=0}^{i=\omega}$$

where α_i runs over the set $A(Z_k)$ of all actions, which are the elements of the activity Z_k; i.e., $A(Z_k) = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_\omega\}$.

$\Phi_i = \langle R_{\phi i}, R_{\pi i} \rangle$ belongs to the *permission structure* Φ_{Z_k}.

$\Psi_i = \langle R_{\gamma i}, R_{\sigma i} \rangle$ belong to the *intention structure.*

The relation $R_{\gamma i}$ defines the interrelations between the intended passes, and $R_{\sigma i}$ between the intended permits in a way which is analogical to the definitions for the permission structure Φ_i. The difference between Φ_i and Ψ_i is only in the semantics.

In general, changes in the structure can be made by actions ϕ_i which operate on Φ_{Z_k} and which change the $R_{A(Z_k)}$ or by actions ψ_i which operate on Ψ_{Z_k} and change Φ_{Z_k}.

A *trajectory* of $\mathcal{M}(Z_k)$ is an admissible sequence of actions $\alpha_i\alpha_j\psi_i\phi_r\alpha_k \ldots \alpha_j\psi_i\phi_r \ldots \alpha_k \ldots$.

The *dynamics of a participant* is the current state of the vector

$$\mathrm{Dyn}_{Z_k}(x_k) = \{\langle\alpha_i(x_k), \phi_i(x_k), \pi_i(x_k), \gamma_i(x_k), \sigma_i(x_k)\rangle\}_{i=1}^{i=\omega}.$$

Only certain sequences of actions are admissible. The admissibility of sequences must be specified by some additional rules which depend on the type of activity and on the character of actions.

Example continued—the set of all actions $A(Z_k) = \{\alpha_1, \alpha_2, \phi_1, \pi_1\}$

α_1 . . . ***inspect data***

α_2 . . . ***record data***

ϕ_1 . . . ***pass the capability inspect data***

π_1 . . . ***permit to pass the capability inspect data capabilities are defined by the action relations:***

$$R_{\alpha_1} = \{(x, x),(x, x1), (x, x2),(x1, x1),(x1, x3), (x1, x4),(x2, x2),(x2, x5),(x3, x3), (x4, x4),(x5, x3),(x5, x5)\}$$

$$R_{\alpha_2} = \{(x2, x2), (x3, x3), (x3, x5), (x5, x5)\}$$

passes are defined by $\{R_{\phi_1,x}, R_{\phi_1,x1}, R_{\phi_1,x2}\} \equiv R_{\phi_1}$ ***where***

$$R_{\phi_1,x} = \{(x, x1)\};$$

$$R_{\phi_1,x1} = \{(x, x3)\};$$

$$R_{\phi_1,x2} = \{(x, x5)\}$$

permit is defined by $R_{\pi_1,x1} = \{(x3, x4)\}$ ***model of an activity***

$$\mathcal{M}(Z_k) = \{R_{\alpha_1}, R_{\alpha_2}, \Phi_1, \Psi_1\}$$

where

$\Phi_1 = \{R_{\alpha_1,x}, R_{\alpha_1,x1}, R_{\alpha_1,x2}, R_{\pi_1,x1}\} \equiv \{R\Phi_1, R\Pi_1\}$
$\Psi_1 = 1$ ***(universal relation, i.e., every element is in relation to all others). The intention structure in this example is the universal relation, which means that the intention of the participants is to go to the limits which are permitted by the permission structure. (Note that only the passes and permits which are related in the permission as well as in the intention structure can be used—the disjunction of the structures.)***

The trajectory $\alpha_1\alpha_2\phi_1$ ***is the sequence of the following actions:***

(inspect)(record)(modify the R_{α_1} ***according to the pass relation*** $R\Psi$***).***

Let us choose the initial dynamics of the participant $x1$

$$Dyn_{Z_k}(x1) = \{\alpha_1(x1), \phi_1(x1), \pi_1(x1)\}$$

where the ranges of the relations are

$$\alpha_1(x1) = \{x1, x3, x4\};$$

$$\phi_1(x1) = \{(x, x1),(x, x3)\};$$

$$\pi_1(x1) = \{(x3, x4)\}.$$

If the action ϕ_1 ***is applied, it causes the following changes:***

$$\alpha_1(x1) = \{x, x1, x3, x4\}.$$

Now, if the action π_1 ***is applied, then*** $\phi_1(x) = \{(x, x_1),(x, x3),(x3, x4)\}$.

4.4 *A Metalanguage for the Description of Algebraic Models*

Examination of protection requirements in various systems (cf. section 2 for the references) and their formulation in an algebraic form shows that there does not exist an unique universal structure which can capture mutual dependence of the relations of the model and satisfy all applications simultaneously. Indeed, this structure has to change in accordance with the type of protection problem examined.

A host of diverse structures arising in this way accentuates the introduction of a metalanguage for examination of algebraic models of protection. Strong need for such a metalanguage can be exemplified by an example taken from the literature. Graham and Denning[21] (p. 423) argue that "reading is a very powerful operation, as it implies, for example, the ability to read and copy file. . . . In this sense, the 'read' attribute is equivalent to the 'read*' attribute". Yet, this is not implied by the rules R1–R8 of Graham and Denning, and can be avoided by careful design of the operating system. Indeed we can argue that this is an implementation fault that should be avoided. Hence, the examination of foundations of protection models and of the implications of the protection rules is needed, instead of the acceptance of certain "obvious" consequences without proper examination.

What should a suitable metalanguage be? Again, the need to introduce modalities naturally arises in

our attempt to cope with this problem. Necessities and possibilities *in this context* should be clearly distinguished from modalities and topologies introduced inside the algebraic model (cf. section 3). Here, necessity (which is a constraint on the category of the protection structures) implies what the minimal protection rules are, which distinguish protection structures from other algebraic or logic structures. Possibility implies what can be introduced as some additional rules characterising a particular protection structure.

We shall now give a very brief exposition of some rules which our model has to obey, as an illustration and a conclusion to this section.

The symbols &, ∨, [,], will be used only as the symbols of a metalanguage $\mathscr{D}_{\kappa i}$ is the domain and $\mathscr{E}_{\kappa i}$ is the range of the relation $R_{\kappa i}$.

1. $\Box[x_i \in \{X\}]$

   (each participant must belong to the set X).

2. $[R_{\alpha i}(x_i, x_j)] \Rightarrow \Box[[x_i \in \{\text{sub}\}] \,\&\, [x_j \in \{\text{ob}\}]]$

   (every participant in the domain of $R_{\alpha i}$ must be a subject; in the range—an object).

3. $[R_{\phi i}(x_i, x_j, x_k)] \Rightarrow \Box[[x_i x_j \in \{\text{sub}\}] \,\&\, [x_k \in \{\text{ob}\}]]$

   (analogical for this ternary relation).

An analogical rule holds for $R_{\pi i}$, $R_{\gamma i}$, $R_{\sigma i}$.

4. $\Diamond\,[[x_i \in \{\text{ob}\}] \,\&\, [x_i \in \{\text{sub}\}]]$

   (it is possible that a participant is an object and a subject simultaneously).

5. $[\exists\, \text{seq}_\tau \,\&\, \exists \phi_i] \Rightarrow \Diamond\,[\text{seq}_\tau \circ \phi_i]$

   (if there exist two sequences of actions, then it is possible that there exists their composition).

6. $[[x_i \notin \mathscr{E}_{\pi i, xk}] \,\&\, [x_i \in \mathscr{E}_{\sigma i, xk}] \,\&\, [x_i \notin \mathscr{D}_{\phi i, xk}] \Rightarrow \sim \Diamond\,[R_{\alpha i}(x_i, x_k)]$
   $[[x_i \notin \mathscr{D}_{\phi i, xk}] \,\&\, [x_i \in \mathscr{D}_{\gamma i, xk}]] \Rightarrow \sim \Diamond\,[R_{\alpha i}(x_i, x_k)]$

   (if the participant does not belong to the domain of the permission structure then it is not in $R_{\alpha i}$).

7. $[x_i \in \mathscr{D}_{\phi i, xk}] \Rightarrow \Diamond\, R_{\alpha i}(x_i, x_k)$

   (interrelations of the permission structure with the relation $R_{\alpha i}$).

   $[x_i \in \mathscr{D}_{\pi i, xk}] \Rightarrow \Diamond\, R_{\phi i}(x_i, \text{sub}, x_k)$

8. $[x_i \in \mathscr{D}_{\sigma i, xk}] \Rightarrow \Diamond\, R_{\phi i}(x_i, \text{sub}, x_k)$

   (interrelation of the intention structure with the permission structure and with the relation $R_{\alpha i}$).

   $[x_i \in \mathscr{D}_{\gamma i, xk}] \Rightarrow \Diamond\, R_{\alpha i}(x_i, x_k)$

4.5 *Rules for Composition of Actions*

Rules for composition of actions entering into an activity Z_k cannot be entirely arbitrary. The set of admissible sequences of actions is determined by the type of activity and by the objectives of protection. However, it should be noticed, that the rules of composition also depend on the characteristics of a protected system. Let us take as an example the action "read". The previously quoted statement of Graham and Denning ". . . reading implies . . . the ability to read and copy file . . ." means that in the system they had in mind the capability "read" is equal to the capability "read/write" in certain activities. We can, of course, design a monitor which would allow us to introduce the capability "read" without the above mentioned unwanted consequences. From this example we can make some fairly general conclusions, which have impact not only on the design of protection structures as such, but what is more important, on the design of the whole system. That is, elementary actions should be chosen in such a way as to limit the consequences of *uncontrollable transitivity* of actions.

Now we shall introduce an appropriate semantics into our model in order to be able to handle this problem. An action of one participant upon another is called a *direct action* if there is no other participant involved as a mediator. An *indirect action* is an action in which a participant achieves certain aims with respect to another participant through a third participant or through a chain of participants.

Let x_i perform an action α_k on x_j, defined by $R_{\alpha_k}(x_i, x_j)$. We shall abbreviate this by $(x \xrightarrow{\alpha_k} x_j)$. Then we can give the following reduction rules, where the symbol ∘ means the composition of actions:

$$\frac{(x_i \xrightarrow{\alpha_r} x_j) \circ (x_j \xrightarrow{\alpha_r} x_k)}{(x_i \xrightarrow{\alpha_r'} x_k)}$$

a transitive action which composed gives an indirect action α_r'; note that the direct action $(x_i \xrightarrow{\alpha_r} x_k)$ is not always defined.

$$\frac{(x_i \xrightarrow{\alpha_r} x_j) \circ (x_j \xrightarrow{\alpha_r} x_k)}{(x_i \xrightarrow{\alpha_r} x_j) \vee (x_j \xrightarrow{\alpha_r} x_k)}$$

an intransitive action either $(x_i \xrightarrow{\alpha_r} x_j)$ or $(x_j \xrightarrow{\alpha_k} x_k)$ or both.

More generally:

$$\frac{(x_i \xrightarrow{\alpha_r} x_j) \circ (x_j \xrightarrow{\alpha_s} x_k)}{(x_i \xrightarrow{\alpha_p} x_k)}$$

$$\frac{(x_i \xrightarrow{\alpha_r} x_j) \circ (x_j \xrightarrow{\alpha_s} x_k)}{(x_i \xrightarrow{\alpha_r} x_j) \vee (x_j \xrightarrow{\alpha_s} x_k)}.$$

Again similar rules can be given for passes and permits.

Example continued—the action α_1 (inspect data) is not transitive and a corresponding indirect action cannot be formed by a simple composition of two direct actions α_1. For example, taking the subjects x, x2, x5, we get:

$$\frac{(x \xrightarrow{\alpha_1} x2) \circ (x2 \xrightarrow{\alpha_1} x5)}{(x \xrightarrow{\alpha_1} x2) \vee (x2 \xrightarrow{\alpha_1} x5)}.$$

The action α_2 (record data) has different properties. For example, if x3 records data into x5, and x5 into x2 consequently, then x2 owns the data of x3 although x3 cannot write into x2. This is an example of the indirect action α_2'. Take the participants x2, x3, x5 and look at the reduction rules:

$$\frac{(x3 \xrightarrow{\alpha_2} x5) \circ (x5 \xrightarrow{\alpha_2} x2)}{(x3 \xrightarrow{\alpha_2'} x2)}.$$

The indirect action α_1' (inspect data of . . .) can be formed by the composition of α_1 and α_2. For example, if x3 records its information into x5 and x2 inspects x5, then x2 is able to inspect indirectly x3. Let us look at some interesting cases:

for the activity $\alpha_2\alpha_1$ we get:

$$\frac{(x3 \xrightarrow{\alpha_2} x5) \circ (x2 \xrightarrow{\alpha_1} x5)}{(x2 \xrightarrow{\alpha_1'} x3)} \quad \textit{(indirect } \alpha_1')$$

but for the activity $\alpha_1\alpha_2$:

$$\frac{(x2 \xrightarrow{\alpha_1} x5) \circ (x3 \xrightarrow{\alpha_2} x5)}{(x2 \xrightarrow{\alpha_1} x5) \vee (x3 \xrightarrow{\alpha_2} x5)} \quad \textit{(no indirect action)}$$

Following is the result of the activity $\alpha_1\alpha_2\alpha_1$:

$$\frac{(x5 \xrightarrow{\alpha_1} x3) \circ (x5 \xrightarrow{\alpha_2} x5) \circ (x2 \xrightarrow{\alpha_1} x5)}{(x2 \xrightarrow{\alpha_1'} x3)}$$

(indirect action).

4.6 *Hierarchical Structure of the Protection Model and its Description by Systems of Logic and Topology*

The crucial feature of the model $\mathcal{M}(Z_k)$ is the highly specific hierarchical interrelation of its composing structures which forms a *hierarchy of sequential* machines. This static hierarchical structure as well as the dynamics of the model can be expressed in modal or many-valued logics or by general topological structures which can be made mutually interchangeable. It is necessary to distinguish three qualitatively different actions in the sequence of admissible actions: firstly, actions of subjects on objects, as they are enabled by capabilities, secondly, actions of subjects on other subjects which amount to the passing of capabilities, and thirdly, actions of subjects on other subjects which permit the transfer of passes. Hence, three qualitatively distinct levels appear in the dynamics of the whole model, as well as in the dynamics of the individual participants. This becomes obvious if the last statement is re-interpreted in terms of abstract automata.

The relation between subjects and objects which is described by the R_α of the model, represents in these terms a *finite-state automaton*, acceptor, which accepts all admissible sequences of α-actions. The set of all *participants represents states* and the *transitions* are represented by *individual actions* on participants. Similar finite-automata describe the R_ϕ and R_γ components of the model (passes). If the R_ϕ and R_γ both accept an action, which means the passing of a capability, the structure of the R_α will be modified i.e., a new transition added into the R_α automaton. At the same time, if the automata corresponding to R_π and R_σ accept the same action, the permitted passes and intended passes will be modified (i.e.) new transition added into R_ϕ and R_γ automata respectively.

4.7 *Topological Models*

As we stated above (section 3.1), questions about behaviour of participants and about possible violations of protection can be formulated in terms of reachability and controllability in the state-space

of a protection automaton. Reachability and controllability can be discussed in terms of generalised closures in extended topologies[19] which have been shown to be semantic models of some modal logics.[54,55] The considerable advantage of the topological approach consists in the fact that the topological structure "forgets" parts of the automata structure which are inessential to the dynamics of the behaviour of participants. We can look at the behaviour either of mutually suspicious groups of processes, or of several rival groups inside which the member participants cooperate, etc.

We shall use some elements of the theory of generalised (extended) topology in the sequel, the basic definitions have been given by Kohout[19] together with more details and an extensive annotated bibliography on the subject. Closures in generalised topologies offer a tool for investigation of the dynamics of protection as well as of its limit case established for infinite strings of admissible actions.

The basic element of the topological model is the direct action (pass, permit) closure $\mathbf{a}_i(\mathbf{f}_i, \mathbf{g}_i, \mathbf{r}_i, \mathbf{s}_i)$ generated by the action $\alpha_i(\phi_i, \pi_i)$. It is defined as a mapping on the power set of all participants:

$$\mathbf{a}_i : \mathscr{P}(X) \to \mathscr{P}(X)$$

$$\mathbf{a}_i(A) = \mathscr{E}_{x_j \in A}\alpha_i(x_j) = \mathscr{E}_{\alpha_i}(A) \qquad A \subset \{X\}$$

It represents the set of all participants (objects in this case) which can be acted on by the subset A of the set of all participants by a direct action α_i in a particular activity. Similarly, we can define the direct pass and the direct permit closures:

Permission structure	Intention structure	
$\mathbf{f}_{\phi i,xj}(A) = \mathscr{E}_{\phi i,xj}(A)$	$\mathbf{g}_{\gamma i,xj}(A) = \mathscr{E}_{\gamma i,xj}(A)$	passes
$\mathbf{r}_{\pi i,xj}(A) = \mathscr{E}_{\pi i,xj}(A)$	$\mathbf{s}_{\sigma i,xj}(A) = \mathscr{E}_{\sigma i,xj}(A)$	permits

where $\mathscr{E}_{\kappa i,xj}(A)$, is the range of corresponding relations $R_{\kappa i,xj}(x_k, x_1)$ etc., for $x_k, x_1 \in \{X\}$, $\kappa_i = \{\phi_i, \pi_i, \gamma_i, \sigma_i\}$.

Every direct action (pass, permit) closure will be an **A**-topology,[19] i.e., an **A**-axiom will hold:

$$\mathbf{A} : \mathbf{c}_i(A) \cup \mathbf{c}_i(B) = \mathbf{c}_i(A \cup B)$$

$$A, B \subset \{X\};\ \mathbf{c}_i \in (\mathbf{a}_i, \mathbf{f}_i, g_i, \mathbf{r}_i, \mathbf{s}_i).$$

An important closure derived from the direct closure action closure is the **AIOU**-modification[19] of the given **A**-topology. For this (transitive) closure the important **U**-axiom $\mathbf{u}(\mathbf{u}(x_i)) = \mathbf{u}(x_i)$ holds. In terms of control and automata theory it is the *region of reachability* i.e., the limit case of propagation of the effect of particular action or a set of actions. In modal terms, it defines the *possibility* of the existence of the effect of a selected action on the participants which are members of that closure.

Propagation of the effect of a set of actions which is given by a particular trajectory of $\mathscr{M}(Z_k)$ (i.e., by a selected admissible sequence of actions) can be investigated using iterations of the above defined closures. The k-th iteration will be given by

$$\mathbf{c}_i^k(A) = \mathbf{c}_i(\mathbf{c}_i^{k-1}(A))$$

$$\mathbf{c}_i \in (\mathbf{a}_i, \mathbf{f}_i, \mathbf{g}_i, \mathbf{r}_i, \mathbf{s}_i);\quad A \subset \{X\}.$$

Example continued—we shall examine the direct closure* a *for the action* α_1. *We have already shown that the action* α_1 *is not transitive. Hence, the closure* a *will also determine the limit case of propagation of* α_1 *action. We shall list the closures of all singletons of the example

$$\begin{aligned}
\mathbf{a}(x) &= \{x, x1, x2\}\\
\mathbf{a}(x1) &= \{x1, x3, x4\}\\
\mathbf{a}(x2) &= \{x2, x5\}\\
\mathbf{a}(x3) &= \{x3\}\\
\mathbf{a}(x4) &= \{x4\}\\
\mathbf{a}(x5) &= \{x3, x5\}
\end{aligned}$$

a(X) ***is the set of all participants whose files can be inspected by X. Now, let the trajectory of the system be*** $\alpha_1\alpha_1\alpha_1 \ldots \alpha_1\phi_1\pi_1$, ***that is the pass is presented and a permit is given in this sequence. This will cause the following change in the closures from above:***

$$\begin{aligned}
\mathbf{a}(x1) &= \{x, x1, x2, x4\}\\
\mathbf{a}(x3) &= \{x1, x3\}\\
\mathbf{a}(x4) &= \{x1, x4\}\\
\mathbf{a}(x5) &= \{x1, x3, x5\}.
\end{aligned}$$

Let us designate $\alpha = \alpha_1\alpha_1\alpha_1 \ldots \alpha_1\phi_1\pi_1$ ***and*** $\alpha_3 = \alpha_1\alpha_2$ ***then for the trajectory*** $\alpha\alpha_3$ ***we shall list the closures for all participants:***

$$\begin{aligned}
\mathbf{a}(x) &= \{x, x1, x2\}\\
\mathbf{a}(x1) &= \{x, x1, x3, x4\}\\
\mathbf{a}(x2) &= \{x2, x5\}\\
\mathbf{a}(x3) &= \{x1, x3\}\\
\mathbf{a}(x4) &= \{x1, x4\}\\
\mathbf{a}(x5) &= \{x2, x3, x5\}.
\end{aligned}$$

The effect of the new application of the action α_3 ***(i.e., the resulting trajectory*** $\alpha\alpha_3\alpha_3$***) is computed by the second iteration*** $\mathbf{a}(\mathbf{a}(x_i)) = \mathbf{a}^2(x_i)$. ***It will change the following closures:***

$$\mathbf{a}^2(x2) = \{x, x1, x2, x3, x5\}$$
$$\mathbf{a}^2(x3) = \{x, x1, x2, x3\}$$
$$\mathbf{a}^2(x) = \{x, x1, x2, x5\}.$$

For the third iteration (the trajectory $\alpha\alpha_3\alpha_3\alpha_3$) we get changes:

$$\mathbf{a}^3(x) = \{x, x1, x2, x3, x5\} = \mathbf{a}^3(x5) = \mathbf{a}^3(x2)$$
$$\mathbf{a}^3(x1) = \{x, x1, x3, x4\}$$
$$\mathbf{a}^3(x3) = \{x1, x3\}$$
$$\mathbf{a}^3(x4) = \{x1, x4\}.$$

Further iterations (applying α_3) will not change the closure. We can see that we have computed the transitive closure (the* U*-modification of the original topology). This determines the worst case of the security in the system.

From this last computation it can be seen that the requirement on the security specified in the above has been violated so that the competitors can obtain the content of the data files of x, x1, x2 from x5 at once. Hence, the permission structure has to be modified. This can be achieved e.g., by the elimination of the link (x2, x2) in R_{α_2}. Then $\mathbf{a}^3(x5) = \{x1, x2, x3, x5\}$.

4.8 *Modal Logics*

Detailed examination of the meaning of individual closures points at an interesting connection with modalities. For example a closure in the **AIOU**-modification of a topology describing the α-structures determines explicitly the set of subjects, i.e., it determines what is *possible* in certain situations. Similarly, different kinds of *possibilities* correspond to closures in other parts of the algebraic model (in permission and intention structures). It is obvious that, although formally the same in different parts of the model, the closures will express different grades of possibility according to the part of the model in which they appear. Apart from *alethic* modalities, there appear *deontic* modalities of permission and obligation. The intention structures are clearly connected with *aims* of subjects and this leads to yet another type of modality. However, each type of modality is not without relation to other types of modality and for this reason mixed modalities have to be introduced.

The algebraic method of McKinsey and Tarski,[60] further extended by Lemmon,[54,55] provides a formal link between general topologies and modal logics. This approach can be extended to mixed modalities using results on lattices of topologies[61] and modifications of generalised topologies.[63,19] It has been shown by McKinsey and Tarski that there is a close connection between Lewis's S4 modal system and *closure* algebras. A similar connection has been established between *T* and *extension* algebras by Lemmon. In topological terminology, a closure algebra is an **AIOU**-topology and an extension algebra is an **AIO**-topology. The modal and epistemic algebras which were introduced in the study of algebraic semantics by Lemmon[54] are as a matter of fact **A**- and **AI**-topologies respectively.

Example continued—we shall introduce some modalities into our example.

Let the sentence $In(x_i, x_j)$ have the meaning "x_i can inspect x_j". Then a direct action closure will determine the* possibility *of the inspection of the set of participants which are contained in the closure. Because the* a(*X*) *closure is an* AIO*-topology, this type of possibility will be defined by the axioms of an* extension algebra, *which determine the* T*-modal logic. The following are examples of true statements for our example: $\Diamond$ In (x2, x5); $\sim \Diamond$ In (x2, x4); $\Diamond$ In (x2, x2) & $\Diamond$ In (x2, x5) etc. The condition specifying that the security of the system of this example is not violated (i.e., y5 and z5 must not get x, x1, x2) can be expressed as follows: $\sim \Diamond$(In(y5, x) & In(y5, x1) & In(y5, x2) $\vee$ In(z5, x) & In(z5, x1) & In(z5, x2)).

4.9 *Finite-Valued Logics*

Systems of mixed modalities developed for general description of protection structures are in the most general case infinite-valued. On the other hand while describing the structure of a particular protection system we deal with a finite system which has a finite number of elements with a semi-discrete topology. In this case we deal with a hierarchy of finite state automata with strong *structural* constraints. Using the structural theory of automata based on binary logics is not sufficient for the description of such structures. We have to introduce many-valued logic which will be at least a ternary logic, as three types of conditions have to be taken into account:

a) conditions describing capabilities, passes and permits which *must be present* and which are necessary for a proper functioning of the whole system;

b) conditions specifying which capabilities,

passes and permits would violate the protection status of the whole system and therefore *must not be present*;

c) *don't care* conditions specifying which capabilities, passes and permits can be introduced by individual participants.

This leads to a logic with the following interpretation of the values: {necessary, don't care, impossible}: a structural model of protection structures in this form (i.e., as a sequential machine) does not answer global questions about protection structures, while topological models do. In this form it is yet another algebraic description using a logical calculus. In order to investigate the validity of formulas or to determine a set of tautologies in this model we have to introduce a set of distinguished elements in order to define a logic in a given logical calculus. We have also to introduce the value specifying violation of the above conditions. This gives a 4-valued logical calculus with the following interpretation of the logic values: {necessary, don't care, impossible, violation} = $\{n, d, i, v\}$. The set of distinguished elements is then chosen according to the type of question we want to ask (for example to determine if a certain sequence of actions violates the protection when it is executed by any possible participant). For this we have to find out if a certain formula is a tautology of the system with a particular set of distinguished elements.

The logical calculus which is used for the structural description has to be functionally complete as we should be able to describe any given protection structure. In such a functionally complete calculus a suitably chosen set of distinguished elements will define a logic. A functionally complete calculus which forms the base for the definition of a suitable logic presented here, has been designed on the set of constructive rules for the determining of functionally complete systems in the Pinkava algebras.[63,64]

One functor of the calculus is already determined by the semantic considerations, giving the relations between the set of values $\{n, d, i, v\}$. The following table must hold, for an event which is necessary and at the same time impossible is a clear violation of the conditions etc.

$\odot$	d	n	i	v
d	d	n	i	v
n	n	n	v	v
i	i	v	i	v
v	v	v	v	v

Taking this functor we have to establish the remaining functors, which form a functionally complete system. If we define the cyclic shift function by:

x	d	n	i	v
$\vec{x}$	n	i	v	d

and the functor of the $\oplus$-type by:

$$x \oplus y = \begin{cases} d \text{ iff } (x = d) \vee (y = d) \\ n \text{ if } (x \neq d) \,\&\, (y = n) \vee (x = n) \,\& \\ \quad \& \,(y \neq d) \end{cases}$$

the system $\{\oplus, \odot, \rightarrow\}$ will be functionally complete. It will be a P_{2_b} system (Ref. 63, Figure 1).

Other many-valued complete systems of logic can be designed for example taking directly the structure of the Multics protection rings and brackets or of the rings of the Minic.

5 MOTOR SKILLS, CONTROL OF MOVEMENT AND PROTECTION STRUCTURES

The ability to move freely and quickly in the surrounding environment has enormous survival value for both animal and man. This ability has been progressively refined in the evolutionary process. A very complex part of the brain structure has specialised to become the control centre of movement. This control centre has a very complex hierarchical structure with many time-variable dynamic interactions. Problems of resources allocation and protection similar to those of big multi-user oriented computer networks arise in this context. Recent attempts in the field of artificial intelligence to build robots or intelligent arm manipulators show that the control of motor skills is by no means trivial.

In order to demonstrate to a non-specialist the problem of protection in the control of movement, we have to give a few very brief characteristic features of this control. This account is based mainly on the work and researches of N. A. Bernstein, which are succinctly summarised in his important monograph "The design of movement".[65] The attempt to express his theories in a suitable mathematical formalism has played an important role in our formulation of the general protection problem.

The general principle of coordination of movements can be described as "the overcoming of super-

fluous degrees of freedom of the moving organs, i.e., transforming it into a controllable system" (Bernstein[65], p. 33). This process is controlled by a multi-level hierarchical system, where the components of movement are supplied by several interacting levels of control. A proper interaction of individual levels of the hierarchy is crucial for the correct performance of movement. Expressed in general systems terminology, the correct function of the system on the global level is maintained by putting necessary dynamic constraints on mutually interconnected and interacting levels of the time-variable hierarchy of the brain control system. "There is no movement (perhaps with extremely rare exceptions) of which all coordinatory components are supplied by only one dominant level of control . . .. At the commencement of the formation of a new individual skilled movement, almost all corrections are in fact made by the dominant, initiating level, acting as surrogate, but this situation soon alters. Sooner or later each technical aspect and detail of the complex movement to be accomplished finds for itself among *low-lying* levels . . . the one most adequate to it . . .. Thus, gradually, a complex many-levelled structure is formed, headed by the *dominant level* which is adequate to the *purpose* of the act of movement . . .. The process of switching the components of a movement onto low background levels is what is usually called the *automatisation of the movement* . . .. In every movement only one dominant level of it becomes conscious . . ." (Bernstein,[65] p. 36).

Each level of control should be understood to be a functional level of control, that is a program or a process of a different degree of abstraction. We are not concerned here with the localisation of particular functions to individual neural structures of the brain. In computer terminology, we are concerned with individual programs, processes, or virtual machines without their hardware implementation. It is obvious from the above-quoted description of automatisation and of the learning of the proper control of a skilled movement, how the protection problem arises in this context. During the transfer from one level of control to another, certain essential characteristic features of the movement have to be preserved, and also some basic constraints on the hierarchy have to be maintained. Failure to do so results in severe malfunction of the whole system.

We shall illustrate this discussion by a simple example of muscle control on the level of synergies. In the coordination of muscles, an important role is played by *synergies*. Synergies are classes of movement with similar dynamic characteristics, with certain predetermined groups of muscles in action. The learning of some movements consists of organising a synergy which decreases the number of parameters necessary for the control. A new synergy is not composed from entirely new or elementary parts, but it utilises the existing elementary synergies and innate neurological mechanisms. The synergies increase the speed of reaction of the organism as well as decrease demands on information-processing capacity during the synthesis of a particular action or a movement. Again, in computer terminology, they are subroutines, ready as whole units to be triggered off in an appropriate situation.

In the process of learning to form new synergies, certain rules of composition of more elementary synergies have to be obeyed. Also, the transfer of the control of already formed synergies from one level of control to another cannot be done in an arbitrary way. Hence, this leads to similar formalism as in the computing, and we have to introduce relations which correspond to passes and permits, in our attempt to make a mathematical model of this problem. However, the dynamics of the process can be even more complex than that of big computer networks, so that it is not sufficient to have a two-level structure with passes and permits. In most instances, a *further recursion is necessary*. For example c_1 gives to c_2 the permission to distribute the access right of b_1 to a subset of objects $\{d_1, d_2, \ldots, d_k\}$. This is an obvious and necessary generalisation in the context of movement studies. This generalisation leads to a quaternary relation.

Let us look at a simple example of control of movement which shows development of new synergies and the role of training or re-training after selective brain lesions.

Example of movement synergies:

The set of participants is composed of eight elements $\{a_1, a_2, a_3, b_1, b_2, m_1, m_2, m_3\}$, where m_1, m_2, m_3 muscles activated by control centres $\{a_i\}\{b_i\}$. The control centres are not necessarily localised in particular neuronal structures. They may be just elements of the functional hierarchy of control[3] (i.e., they can be viewed as control programs or virtual processes in computer terminology).

The action defined by the relation R is facilitation of function of an object (a muscle, a lower-lying control centre) by subjects from the sets $\{a_i\}$, $\{b_i\}$.

The action of facilitation is then defined by:

$$R_\alpha = \{(a_1, m_1), (a_1, m_3), (a_2, m_1), (a_2, m_2), (a_3, m_2), (a_3, m_3), (b_1, a_1), (b_1, b_2), (b_2, a_2)\}.$$

Passes are defined by:

$$R_{\phi_1} = \{R_{\phi_1,a_1}, R_{\phi_1,b_2}\}$$

$$R_{\phi_1,a_1} = \{(b_1, a_1)\}, \quad R_{\phi_1,b_2} = \{(b_1, a_1)\}.$$

Permits are defined by:

$$R_{\pi_1,b_2} = \{(b_1, b_2)\}.$$

All protection structures (closures, etc.) can be determined in a similar way as in the example given in the previous part. We shall leave this as an exercise for the reader.

A model of this type can be used for assessment of movement disorders. Let us assume, for example, that if all muscles are activated it represents a condition of spasticity. It can be easily seen that if a_3 is able to activate a_1 this spasticity will be present (i.e., a_3 will activate m_2, m_3 and a_1; a_1 will activate m_1, m_3, so that all three muscles m_1, m_2, m_3 will be activated).

Passing the right of access to an object or permitting it to pass this right will correspond to training and development of new synergies and of new movement skills. This is important for studies of rehabilitation after brain lesions. Let us assume that the path from b_1 to b_2 is irreversibly lost after an accident. Further, let us assume that the condition "b_2 can activate itself" corresponds to an important synergy. If the condition passed from b_1 to b_2 (this is attained by appropriate training) before the accident, the synergy may be preserved. If, on the other hand, this transfer had not occurred before the accident, the synergy may be irreversibly lost.

In the application of the concepts of protection structures to the motor system one is limited by the need to *identify* the system, whereas it is specified in advance for computer applications. Thus there is an inherent vagueness which itself needs to be incorporated into the theory. However, these brief notes serve to illustrate the role of protection concepts to the study of motor skills. The concepts themselves are inherent in much of the psychophysiological literature, particularly that of movement disorders. However, a general systems approach to their analysis brings out their similarity to other protection structure concepts in a way that can only be mutually beneficial to all the disciplines concerned.

6 THE RELEVANCE OF PROTECTION STRUCTURES TO GENERAL SYSTEMS

Hitherto, we have shown the use and relevance of protection structures in two applied systems areas. A careful examination of both applications reveals the features they have in common, which can be formulated independently of a particular applied systems area, and which may have a wider relevance. Let us pinpoint informally the characteristic features of systems for which the application of protection structures is relevant, before we attempt to give a more formal presentation.

One characteristic feature is that both systems have a multi-level structure. The individual components of the systems are mutually interconnected, but they do not form a strict, rigid hierarchy, given once and for all. The systems can be artificially decomposed into parts which form hierarchies, or more generally, partially ordered subsets, but the mappings between these parts given by equivalence or coincidence of individual elements, are not order preserving. The interconnection of these artificially decomposed parts into the original system destroys the partial orderings or hierarchies in most instances, but the multi-level character is preserved. The other characteristic feature is that the interconnections between the components of a system are not fixed, they change with time, they have a dynamic character. For example, at one instance they may form a proper hierarchy which may soon disappear again, and then a new hierarchy may appear which cannot be homomorphically mapped by any order preserving mapping onto the previous one.

Protection structures in general systems contexts can be viewed as the constraints which limit the unwanted or the dangerous interactions which would otherwise destroy the most essential functions and characteristic features of a system. If these essential functions were destroyed, the qualitative character of the system would change and the very purpose for which the system may have been created would be defeated.

There has been much heated discussion about the relevance and usefulness of hierarchical systems. The presence and the relevance of hierarchies in various systems has been quickly recognised but attempts at the creation of an efficient hierarchical

systems theory have not been entirely successful. So far only the design of hierarchical controllers has met with certain success.[66,67] In the area of applied computer programming the idea of hierarchies is more controversial. There exists ". . . the contradiction between efficient construction and efficient execution (of computer programs) a contradiction that has been with practical computing since its inception. The strongest argument raised against the "top-down" (i.e., hierarchical) programming approach is that it leads to unacceptable inefficiences. Only recently have we begun to understand that, in the long run, non-hierarchical (or otherwise non-structured) programs become less and less efficient as they are inevitably modified. It would have been nice to gain this experience from hierarchy theory, rather than from bitter experience, but practice cannot wait" (Weinberg,[68] p. 272). In social sciences, some researchers worship hierarchies, some feel that the hierarchies are very rare: ". . . Navies and Churches aside - hierarchies are really much more rare in human affairs than cursory thought implies . . . our age stands in grave danger of making over human life in the image of its theories" (Weinberg,[68] p. 272). Weinberg makes his point, realising that many attempts to utilise hierarchical models fail because the real-life data are fitted to a rather unnatural and artificial hierarchical structure. Indeed, strict hierarchies may be rare and their structure may be variable, or may not survive for long but a multi-level structure is very common. Researchers in Artificial Intelligence have realised the importance of multi-level structures and the disadvantage of strict hierarchies. They emphasised the importance of heterarchies, rather than hierarchies. Experiences with "artificial intelligence" languages (Conniver) prove the point.

Let us summarise our main argument. Multi-level rather than strictly hierarchical systems are very important and protection structures offer the techniques which guarantee that the features essential for the proper functioning, and indeed the very existence, of such systems are preserved. Unambiguously defined restrictions and specific protection leave scope for the independent actions, learning, and adaptation of individual elements of a system and restrict only the "dangerous" actions. This presupposes only the knowledge of the undesired traits of behaviour, instead of a fuller knowledge of possible behaviour, which is needed if we attempt to coordinate the activities of the elements instead, as it has been proposed by Mesarovic, Macko and Takahara.[67]

Protection Structures in the Context of General Systems Theories

Now we shall examine protection structures in the context of General Systems, showing the correspondence of the previously introduced terms to the terms and notions used in the general systems field. Because we are not interested at this instant in showing the exact mathematical correspondence to a particular general systems theory, we choose the language of such general systems theory as suits our purpose best. That is, the one which obeys the canon of the Society for General Systems Research and can be used to "investigate the isomorphy of concepts, laws, and models in various fields, and to help in useful transfer from one field to another". A vocabulary of terms and concepts suitable for our purpose is offered by George Klir's approach.[69]

The participants, objects and subjects can be viewed as elements of an *UC*-structure.[69] The action matrix, related to capabilities, represents directed couplings of the elements (i.e., participants, in our terminology) of the *UC*-structure. The permission and the intention structures, with their passes and permits, and eventually further structures above, in case we require a deeper recursion, represent the *dynamics of change* of directed couplings. It should be realised that the protection structure represents the permitted couplings in its action matrix, where not all actions will be necessarily executed, so that some permitted couplings may not actually exist. Because the algebraic structure of interaction between capabilities must be positive, i.e., one capability cannot cancel another out, there exist certain possible, permanent *UC*-structures, which correspond to the initial minimal action matrix.

Thus, abstract protection structures offer extremely general examples of the concepts of general systems theory. However, they also provide, through actual implementations and schemes discussed in both the biological and computing literature, test cases for the application of specific system concepts. We see them as increasingly important application areas for the evaluation of general systems theory and its practical utility.

7 COMPUTER-AIDED DESIGN OF PROTECTION STRUCTURES

The simple examples developed through the paper demonstrate the great complexity of the dynamics of protection structures. The mathematical models proposed would be an academic exercise, devoid of

relevance to real world protection and security problems, if they were not directly amenable to computer-aided design. However, in practice, it is our search for computer-aided methods for analysis of protection structures, that has largely motivated our choice of mathematical techniques. This section briefly outlines some computational aspects of the mathematical structures previously described.

7.1 *A Metalanguage of Protection Structures and Theorem Provers*

The ultimate aim is to design a machine theorem prover of statements about protection structures. Alethic modalities are sufficient for most practical applications since we are concerned with the "hard" constraints rather than conventions. Semantic studies of logics suitable for expressing scientific, technological and legal problems, especially the recent development of a "Calculus of Problems" [70] indicate that an S4 modal system may be adequate for the study of many protection structures. Recently, very powerful mechanical proof techniques for modal logics have been developed, which may be directly translated into computer programs (Ref. 14, p. 12).

7.2 *Computation of the Dynamics of Protection Structures*

The algebraic model which is formed by a hierarchy of sequential machines, presents the usual computational problems of combinatorial character which are encountered in automata theory.

By forming closures on the protection automata, we select only the information which is pertinent to the given question, reducing enormously the computation required. Dynamics of actions can be comprehensively investigated using iterations and modifications of relevant topological spaces.[62,19] Opting for these methods we eliminate exhaustive search over very large sets and replace it by algebraic manipulation of much smaller lattice structures through iterations in lattices of topologies.

In the case where it is better to represent closures indirectly as possibilities in some modal logics, the techniques referred to in 7.1 above can be used. They are valid for very general modal systems.[14]

7.3 *Finite Logics in Computer-Aided Design of Protection*

In section 4.9 we gave an example of techniques for design of many-valued logics usable for description of protection structures. In order to be able to investigate the validity of logical formulas in such logics we have to establish mechanical reduction rules. Suitable techniques giving reduction rules for finite-valued logics has been given by Surma.[71,72] This is an obvious analogy of the cancellation rules for modal logics with similar consequences for design of machine theorem provers.

The mechanical proof techniques for modal and many-valued logics, which are of very recent origin, and therefore very little known and largely unused in computing, supplied the main motivation for our uses of powerful logics. Their importance can be highlighted by a quotation from Snyder,[14] p. 12:

"*Proving theorems within a given system of logic involves following a straightforward mechanical procedure. . . . The high adventure of seeking clever strategies for deductive proofs, and the concomitant satisfaction of finding such proofs and being able to claim new theorems, are lost in the present set of formal systems. Instead, the adventure of doing logic . . . lies in the development of a variety of systems of logic for a variety of tasks.*"

8 CONCLUSION

We have shown that protection structures have their place in the context of general systems theories. Multi-level structures play an important role in various specialised fields, where general systems theories may be applied. Indeed, the most efficient biological systems have multi-level structures. This applies to the brain structures as well as to control mechanisms on sub-cellular levels (Patee[73]).

The essential characteristic of such systems is that the number of levels and the interaction of individual elements may be variable, in order to suit best a given purpose at a particular instant. The *ST*-behavioural traits may be variable as well, subject to learning and adaptation of individual elements of the *UC*-structure, or to cooperation or competition of these elements. (For examples see Tsetlin.[74]) The future behaviour of individual elements of a *UC*-structure may not be fully known, because of the future learning or competition or "bad intentions" of individual elements, so that it may be difficult to put forward or enforce any principles of coordination (as has been proposed in [67]). Instead, in this case, only such minimal constraints as are necessary for the preservation of the system should be enforced by a protection structure.

According to Yessenin-Volpin[75,76] there exist two different types of incompleteness of algorithms (and, correspondingly, of strategies): a) incompleteness caused by existence of multiple rules or possibilities of actions, without any prescription as to which one should be used; b) incompleteness caused by withdrawal of algorithms or possibilities of action, i.e., enforcing a certain situation on participants without their free choice. In the first instance, it is possible to explore a liberal algorithmic method, in the second instance a despotic algorithmic method, where both of them can be expressed in appropriate formulas of modal logic. Enforcement of coordination without respect for individual independent behaviour of individual participants in a system may lead to enormous rigidity, inefficiency as we witness in some despotic societies. On the other hand, a system without any restricting rules which would preserve the minimal rights of each participant and protect each participant from the malicious actions of others, is an excessively permissive system, the function of which may break down. The natural evolution shows that only such systems efficiently survive which can preserve the maximum of independent action of individual participants or components of the system, simultaneously protecting the minimal rights and necessary actions of each participant. The evolution shows that an extreme in any direction may be fatal for the system.

It is this role of protection structures as the local means of ensuring global performance in systems with many interacting sub-systems that gives them their fundamental importance in such diverse areas as sociology, biology, and computer science. General system theories suffer in some respects from their very generality which makes them appear to be so all-embracing as to have no content. The specialisation of such theories to the case of protection structures offers an opportunity to demonstrate continual diversity of application whilst providing a wealth of important practical examples.

REFERENCES

1. G. J. Popek, "Protection structures". *Computer*, **7**, June 1974, pp. 22–33.
2. L. Kohout and B. R. Gaines, "The logic of protection". *Proceedings of GI '75, 5th Annual Congress of the Gesellschaft für Informatik* (October 1975, Dortmund University).
3. L. Kohout, "A formal representation of functional hierarchies of movement in the brain and its relation to cybernetics". *Proceedings of Second Conference on Recent Topics in Cybernetics*, September 1975, Cybernetics Society, London.
4. "The plan for information society", *Final Report of the Computerization Committee of the Japan Computer Usage Development Institute*, Tokyo, 1972.
5. N. A. Bernstein, *The Coordination and Regulation of Movements*. Pergamon Press, Oxford, 1967.
6. B. R. Gaines, P. V. Facey, F. K. Williamson, and J. A. Maine, "Design objectives for a descriptor-organised minicomputer". Proceedings of European Computing Congress (*Eurocomp 74*), May 1974, Online Ltd., London, pp. 29–45.
7. F. K. Williamson, B. R. Gaines, J. A. Maine, and P. V. Facey, "A high-level minicomputer". Proceedings of IFIP Congress, Stockholm, August 1974, *Information Processing 1974*, North-Holland, Amsterdam, pp. 44–48.
8. B. R. Gaines, M. Haynes, and D. Hill, "Integration of protection and procedures in a high-level minicomputer". Proceedings of *1974 Computer Systems and Technology Conference*, October 1974, I.E.E., London.
9. L. Kohout, "Quantitative techniques for the investigation of human movement disorders". *Technical reports EES-MMS-HMD1-74, UCH-GER-HMD1-75*. Geriatric Research Unit, St. Pancras Hospital, St. Pancras Way, London, U.K. (This research has been supported by a grant from the Medical Research Council.)
10. B. R. Gaines, "Axioms for adaptive behaviour". *International Journal of Man-Machine Studies*, April 1972, **4**, No. 2, pp. 169–199.
11. B. R. Gaines, "Training, stability and control". *Instructional Science*, July 1974, **3**, No. 2, pp. 151–176.
12. Proceedings of a Colloquium on *Modal and Many-Valued Logics*, Helsinki, 1962. *Acta Philosophica Fennica*, **16**, 1963.
13. G. E. Hughes and M. J. Creswell, *An Introduction to Modal Logic*. Methuen, London, 1968.
14. D. P. Synder, *Modal Logic*. Van Nostrand, New York, 1971.
15. M. J. Creswell, *Logics and Languages*. Methuen, London, 1973.
16. R. Purthill, "Meinongian Deontic Logic". *Philosophical Forum*, 1973, **4**, No. 4, pp. 585–592.
17. H. Rasiowa and R. Sikorski, *The Mathematics of Metamathematics*. Warszawa, Poland, 1970.
18. H. Rasiowa, *An Algebraic Approach to Non-Classical Logics*. North-Holland, Amsterdam, 1974.
19. L. Kohout, "Generalized topologies and their relevance to general systems." *International Journal of General Systems*, January 1975, **2**, no. 1, pp. 25–34.
20. B. R. Gaines and L. J. Kohout, "The logic of automata". *International Journal of General Systems*, October 1975, **2**, No. 4, pp. 191–208.
21. G. S. Graham and P. J. Denning, "Protection—principles and practice". *Proceedings of Spring Joint Computer Conference*, 1972, **40**, AFIPS Press, New Jersey, pp. 417–429.
22. R. M. Fano, "The MAC system: a progress report". In M. A. Fass and W. D. Wilkinson (eds.) *Computer Augmentation of Human Reasoning*, Spartan Books, Washington, pp. 131–150, 1965.
23. E. I. Organick, *The Multics System*, MIT Press, 1972.
24. Session 6: A new remote accessed man-machine system. *Proceedings of AFIPS Fall Joint Computer Conference*, 1965, **27**, Part 1, Spartan Books, Washington, pp. 185–247.

25. E. E. David and R. M. Fano, "Some thoughts about the social implications of accessible computing". *Proceedings of AFIPS Fall Joint Computer Conference*, 1965, **27**, Part 1, Spartan Books, Washington, pp. 243–247.
26. A. Westin, *Privacy and Freedom*. Atheneum, New York, 1968.
27. A. Miller, *The Assault on Privacy*. University of Michigan Press, 1971.
28. E. W. Djikstra, "Cooperating sequential processes". In: F. Genuys (ed.), *Programming Languages*, Academic Press, London, 1968.
29. J. B. Dennis, "Modularity". In: F. L. Bauer (ed.), *Advanced Course in Software Engineering*, Lecture Notes in Economics and Mathematical Systems, 1973, **81**, Springer-Verlag, Berlin, 1973.
30. O. J. Dahl, E. W. Djikstra and C. A. R. Hoare, *Structured Programming*. Academic Press, London, 1972.
31. B. W. Lampson, "Dynamic protection structures". *Proceedings of Fall Joint Computer Conference*, 1969, **35**, AFIPS Press, New Jersey, pp. 27–38.
32. R. S. Fabry, "Capability-based addressing". *Comm. ACM*, July 1974, **17**, No. 7, pp. 403–412.
33. R. M. Needham, "Protection systems and protection implementations". *Proceedings of Fall Joint Computer Conference*, 1972, **41**, AFIPS Press, New Jersey, pp. 572–578.
34. D. M. England, "Architectural features of system 250". *INFOTECH*, State of Art Report on Operating Systems, 1972.
35. B. W. Lampson, "A note on the confinement problem". *Comm. ACM*, October 1973, **16**, No 10, pp. 613–615.
36. B. Elspas, K. N. Levitt, R. J. Waldinger, and A. Waksman, "An assessment of techniques for proving program correctness". *ACM Computing Surveys*, June 1972, **4**, No. 2, pp. 97–147.
37. A. K. Jones, "Protection in programmed systems". Ph.D. thesis, Carnegie-Mellon University, 1973.
38. B. C. M. Dougue and G. M. Nijssen (eds.), *Data Base Description*. North-Holland, Amsterdam, 1975.
39. J. W. Klimbie and K. I. Koffeman (eds.), *Data Base Management*. North-Holland, Amsterdam, 1974.
40. R. W. Conway, W. L. Maxwell, and H. L. Morgan, "On the implementation of security measures in information systems". *Comm. ACM*, April 1972, **15**, No. 4, pp. 211–220.
41. J. H. Morris, "Protection in programming languages". *Comm. ACM*, January 1973, **16**, No. 1, pp. 15–21.
42. B. Wegbreit, "The treatment of data types in EL1". *Comm. ACM*, May 1974, **17**, No. 5, pp. 251–264.
43. H. L. Morgan, "An interrupt-based organization for management information systems". *Comm. ACM*, December 1970, **13**, No. 12, pp. 734–739.
44. M. Zelkowitz, "Interrupt driven programming". *Comm ACM*, June 1971, **14**, No. 6, pp. 417–418.
45. E. A. Feustel, "On the advantages of tagged architecture". *IEEE Trans. Comp.*, 1973, **C-22**, pp. 644–656.
46. C. Hewitt, "PLANNER: a language for manipulating models and proving theorems in a robot". *Proceedings of International Joint Conference on Artificial Intelligence* (IJCAI-69), Washington, May 1969.
47. C. Hewitt, "Procedural imbedding of knowledge in PLANNER". *International Joint Conference on Artificial Intelligence*, London, September 1971.
48. G. J. Sussman and D. W. McDermott, "From PLANNER to CONNIVER—a genetic approach". *Proceedings of Fall Joint Computer Conference*, 1972, **41**, AFIPS Press, New Jersey, pp. 1171–1179.
49. C. Hewitt, "Protection and synchronization in actor systems". *Working Paper—83*, Artificial Intelligence Laboratory, M.I.T., Cambridge, Mass., November 1974.
50. C. Hewitt, "Protection and synchronization in actor systems". *Working Paper—90*, Artificial Intelligence Laboratory, M.I.T., Cambridge, Mass., January 1975.
51. R. M. Burstall, J. S. Collins, and R. J. Popplestone, *Programming in POP-2*. Edinburgh University Press, 1971.
52. B. R. Gaines, P. V. Facey, and J. Sams, "An interactive, display-based system for gilt-edged security broking". Proceedings of European Computing Congress (*EUROCOMP 74*), May 1974, Online Ltd., London, pp. 155–169.
53. B. R. Gaines and P. V. Facey, "Some experience in interactive systems development and application". *Proc. IEEE*, June 1975, **63**, No. 6, pp. 894–911.
54. E. J. Lemmon, "Algebraic semantics for modal logics I". *Journal of Symbolic Logic*, June 1966, **31**, pp. 46–65.
55. E. J. Lemmon, "Algebraic semantics for modal logics II". *Journal of Symbolic Logic*, June 1966, **31**, pp. 191–218.
56. S. MacLane, *Categories for the Working Mathematician*. Springer, New York, 1971.
57. J. A. Goguen, "Semantics of computation". In: *Proceedings of 1st International Symposium on Category Theory Applied to Computation and Control*, Mass., February 1974.
58. M. A. Arbib and E. G. Manes, "Foundations of system theory". *Automatica*, 1974, **10**, pp 285–302.
59. A. Eilenberg, *Automata, Languages and Machines*, Vol. A. Academic Press, New York, 1974.
60. J. C. C. McKinsey and A. Tarski, "Some theorems about the sentential calculi of Levis and Heyting". *Journal of Symbolic Logic*, 1948, **13**, pp. 1–15.
61. E. Čech, *Topological Spaces*. Academia, Prague & J. Wiley, *Interscience*, New York, 1966.
62. K. Koutský and M. Sekanina, "Modifications of Topologies". In: *General Topology and its Relation to Modern Analysis and Algebra 1*, (Proceedings of the Symposium Prague 1961). Academic Press, New York and Academia, Prague, 1962.
63. L. Kohout, "The Pinkava many-valued complete logic systems and their applications in the design of many-valued switching circuits". *Proceedings of 1974 International Symposium on Multiple-Valued Logic*, May 1974, IEEE 74CHO845-8C, pp. 261–284.
64. V. Pinkava, "Some further properties of the Pi-logics". *Proceedings of the 1975 International Symposium on Multiple-Valued Logic*, May 1975, IEEE 75CHO959-7C, pp. 20–26.
65. N. A. Bernstein, *O Postroenii Dvizenii*. Moscow, 1947.
66. J. P. Matuszewski and I. Lefkowitz, "Coordination for control in steel processing". *IEEE Transactions on Systems, Man and Cybernetics*, **SMC-3**, No. 2, March 1973, pp. 182–184.
67. M. D. Mesarovic, D. Macko and Y. Takahara, *Theory of Hierarchical, Multi-level Systems*. Academic Press, New York, 1970.

68. G. M. Weinberg, "Review of the book 'Hierarchy Theory: The Challenge of Complex Systems', H. H. Patee, ed." *International Journal of General Systems*, **1**, No. 4, pp. 271–272.
69. G. J. Klir, *An Approach to General Systems Theory*. Van Nostrand Reinhold, New York, 1969.
70. P. Materna, "On problems (semantic study)". *Rozpravy Československé Akademie Věd Řada, Společenských Věd*, **80**, No. 8, pp. 1–62, 1970. (Published by Academia, Prague.)
71. S. J. Surma, "An algorithm for axiomatising every finite logic". In: *Proceedings of the 1974 International Symposium on Multiple Valued Logic*, Morgantown, W. Va., May 1974. IEEE 74CHO845-8C, pp. 315–322.
72. S. J. Surma, "A method of the construction of finite Lukasiewiczian algebras and its application to a Gentzen-style characterisation of finite logics". *Reports on Mathematical Logic*, **2**, pp. 49–54, 1974.
73. H. H. Patee, "The nature of hierarchical control in living matter". In: *Foundation of Mathematical Biology*, **1**, edited by R. Rosen, Academic Press, New York, 1972.
74. M. L. Tsetlin, *Automata Theory and Modelling in Biological Systems*. Academic Press, New York, 1973.
75. A. S. Yessenin-Volpin (A. S. Essenin-Vol'pin), "The ultra-intuitionistic criticism and the anti-traditional program for foundations of mathematics." In: *Intuitionism and Proof Theory* (Proceedings of the Conference at Buffalo, N.Y., 1968), pp. 3–45, North Holland, Amsterdam, 1970.
76. A. S. Essenin-Vol'pin, "On a theory of modalities", (in Russian). In: *Logika i Metodologia Nauki* (4 Vsesojuznyĭ Simposium, Kiev, 1965), Nauka, Moscow, 1967, pp. 56–66.

For Dr. Kohout's biography, see Vol. 2, No. 1, p. 34 of this journal. For Dr. Gaines' biography, see Vol. 2, No. 4, p. 208.