Survey Paper: The Development of a Research Network Information System Specification Method

Aldo de Moor *
Infolab

October 30, 1995

1 Introduction

Research networks are complex organizations of professionals that in many ways can be supported by information technology. However, providing such networks with generic and isolated tools and data sets is not sufficient. In order to operate professionally, a research network needs a customized and integrated research network information system. We propose the creation of a method [1] that can be used to systematically develop - and maintain - the high level specifications of research network information systems.

Numerous information system development methods have already been constructed. Before we can start the development of our 'Research Network Information System Specification (RENISYS)' method, the focus of my Ph.D. thesis research, we must first understand how it relates to and distinguishes itself from existing work. To this purpose, this paper presents the results of a literature survey in which related approaches and projects are briefly described and their potential relevance to RENISYS is discussed. These findings are still very tentative, as the complexity of the research topic, which combines work from many widely diverging fields, is high, and

*Infolab B303, Tilburg University, P.O. Box 90153, 5000 LE Tilburg, The Netherlands, E-mail: ademoor@kub.nl. The author wishes to thank Hans Weigand for the inspiring discussions which led to this paper.

[1] There is a lot of confusion about the terms 'method' and 'methodology'. Although there is a subtle theoretical difference, in practice these terms are often used interchangeably. Therefore, we will use both terms without distinction.

the relevant literature is voluminous. We therefore do not claim to be complete in this initial survey. Rather, we sketch some broad contours of the envisaged method and try to manufacture a tool box that can be used in its actual construction. In future papers, we will then focus on a selected subset of the issues discussed here in more depth.

This survey paper consists of five parts. First, it briefly defines what research network information systems are and the role that a specification method can play in creating and maintaining them. After that, the paper lists a set of general criteria that any such method should meet. Subsequently, it gives a preliminary and still very basic outline of RENISYS. The paper then presents the results of a literature survey on potentially significant information scientific theories, followed by an overview of relevant concrete information tools, environments and systems. The paper is concluded by a description of some related information system specification methods, and ends with some initial conclusions and ideas about future work.

2 Toward A Research Network Information System Specification Method

In this section, research networks are presented as having interesting information requirements of sufficient complexity to warrant the development of a special information system specification method. First, the concept of research network is explained, and some interesting network activities are discussed. It is explained why information technology is needed to support the conduct of these activities. This information technology, however, must be appropriately combined into customized and comprehensive network information systems to be really useful. The need for a specification method to systematically construct such systems is explained.

2.1 Research Networks: Complex Information Requirements

The scientific research process is rapidly changing from a relatively solitary operation into a much more collaborative effort [Shade, 1994]. Research networks, which focus on facilitating certain stages of the research process, play an important role in this transformation by acting as catalysts of cooperation. A research network can be considered a professional network: a human network which supports its members in accomplishing shared and individual professional goals. This goal-orientedness is one of the crucial distinguishing

characteristics of professional networks. Research networks can have many roles: coordinating the production of scientific reports, disseminating scientific knowledge to the public, offering advice in societal conflicts, evaluating scientific publications, and so on. Furthermore, these networks are often international, interdisciplinary, and interactive in nature, which only increases their complexity [de Moor, 1994]. [2].

Despite the importance of research networks, their characteristics have so far been insufficiently investigated. This is especially true when compared to the amount of attention that for example the role, composition, and information technological support of business networks has received. In that area, topics such as EDI and Just-In-Time systems have been studied quite extensively over the past few years. This lack of attention for research networks is unfortunate as they form an interesting class of networks from an information scientific point of view, with complex collaboration and discourse patterns. This is due to researchers having very diverse backgrounds and goals, being knowledge- and discussion-oriented and sharing a complicated scientific culture. As the research community is one of the most complex, heterogeneous, and demanding groups of professionals, solutions to informational problems of research networks can most probably successfully be extrapolated to many commercial, educational and other types of professional networks.

Most of current research networks are still in the rather informal, individual-supporting stage. However, the more interesting roles research networks can play are those aimed at furthering shared group goals. This demands that the network fosters strong collaboration, which can be defined as collaboration in which a group synergistically develops and improves a structured artifact more efficiently than would be possible by the same group of people working independently [Johnson and Moore, 1994].

For such collaboration in a network to be successful, it is essential that characteristic work processes are identified that the network can and needs to support [Grudin, 1994]. One important such group work process is the evaluation of scientific output. The traditional, anonymous refereeing process of scientific papers is bound to change. Other means of article evaluation need to be found [Wiederhold, 1995]. Research networks could play a very important role as providing the structural framework for new approaches in this field.

Yet probably the most essential collaborative activity that could be tak-

[2] For a classification of networks and a more in-depth treatment of the research network phenomenon, see [de Moor, 1995]

ing place in research networks is the writing of group reports. To understand its importance, we subscribe to the view that the research literature can be seen as a conversation [Agre, 1994]. However, at the moment this conversation is very non-interactive, due to the long times needed for the preparation and distribution of the average publication. Another problem is that each publication in itself (obligatorily) shows a single point of view, whereas often multiple perspectives being represented in the same article could be very helpful to conducting better quality research. What is needed are dialogic texts, which differ from standard collaborative texts in that they reflect the involvement of multiple authorial voices [Harrison and Stephen, 1992]. This is where research network information systems can play a unique role, by supporting the highly interactive, yet systematic creation of such documents. Therefore, analyzing the activities necessary to accomplish the group report writing process will be an important objective of this thesis.

2.2 The Rationale For Using Information Technology To Support Research Networks

A professional network can be regarded as a globally dispersed and dynamic web of knowledge nodes drawn together on a one-time basis to address a unique problem [Jarvenpaa and Ives, 1994]. Because of the 'openness', 'virtuality' and 'dynamics' of such networks, much more sophisticated applications of information technology are necessary than currently available in order to manage this synthesizing process.

Information technology can play a crucial role in improving the operations and impact of research and other professional networks. It helps them to create, collect, process, and communicate relevant information much more efficiently. It should not only offer them some technical facilities for individual or ad hoc group use, but also actively support strong collaboration. Furthermore, this information technology must not only foster task completion, but support productive personal relationships as well [Kraut et al., 1987].

Some fascinating new ideas about future applications of information technology for supporting research are described in [Foley and Pitkow, 1994]. Of course, the use of this kind of powerful technology also has its dangers and drawbacks, such as those related to information control and information overload. However, if wisely applied, information technology can overcome these hurdles and help minimize their impact. For a more detailed discussion of these issues concerning the appropriate use of information technology in research networks, the reader is referred to [de Moor, 1995].

2.3 Wanted: Integrated And Customized Network Information Systems

A lot of research has been done on how to apply information technologies directly relevant to supporting human networking. Examples are the fields of computerized conferencing systems [Black, 1987, Swanson, 1993], interoperable databases [Brodie, 1992], and the newly emerging branch of information science called computer-supported cooperative work (CSCW) [Grudin, 1994, Ellis and Wainer, 1994]. However, many of these efforts have in common that they describe technical ways that only support limited, abstract information and communication processes, such as decision making or information retrieval. The result is that users have to master a large number of unrelated tools and applications before they can reach at least a minimum level of productivity.

Currently, most research networks use only a few standard information tools and isolated data sets, which do not meet the actual information needs of their users. Furthermore, these tools are not well connected, so that it is not clear how they can cooperate to provide the user with a seamless technological research environment and make the data properly accessible. In [de Moor, 1995] we showed some examples of the insufficient adequacy of current information technology support. One of the major problems facing professional users of information technology is that designers think in tools that support certain aspects of activities, while users need some integrated functionality [van den Broek, 1995]. The important point is that offering a number of isolated tools that are not tailored to the requirements of the users will not do to satisfy the many information needs of a research network.

To better meet the information and communication needs of a research network we must focus our attention on realizing the next stage in the 'information technology for research networks evolution'. Comprehensive, customized network information systems are needed. Such systems combine and configure various tools and data according to the ever changing network information needs.

We are thus not talking about the traditional kind of information system, consisting of predetermined, rigid collections of hard- and software which can perform only a number of standard data-processing jobs. Instead, the information system should be able to be smoothly adapted to the different purposes for which it can be used. In complex environments like the research domain, the users need to be actively involved in this adaptation process. Thus, the network participants need to play many roles, not only using, but also being responsible for at least part of the

evolution of the information system. In other words, the information system consists of a dynamic set of modular, actor-initiated information and communication processes plus accompanying data sets, including those that modify other processes. All of these processes must be made explicitly visible in order to better control and modify them. Our investigation now leads us to the following definition of a network information system:

Network information system: a set of meaningfully combined and configured information and communication processes necessary to support and coordinate the activities of the network participants in their various roles.

2.4 The Need For A Specification Method

In order to develop a useful network information system, we must know about what typical activities and roles occur in research networks, which information and communication processes can be distinguished, and what mechanisms are necessary to specify and relate the various processes to one another, so that complete, high-level specifications of the information system can be given. Due to the complexity of the information system specification process this should be done in a well-structured manner. With current practices of system development, though, adequate research network information systems are difficult to create. Most documentation on how to build such systems limits itself to either technical tool-installation guides or broad-ranged, precondition-satisfying topics like the building of a research community [Agre, 1994] and research network organizational and management issues [BDN, 1992, Pimienta, 1993]. Not much attention, however, is paid to a systematic analysis of network information needs and how they can be translated into exact information system specifications. Thus, a more systematic methodological approach is clearly required here.

3 Criteria For The Specification Method

After having described the major focus of the method, the research network information system, we now turn our attention to some important methodological criteria. The method must at least meet the following requirements to be successful. It must be formal in order to generate complete and consistent specifications, dynamic to generate easily adaptable specifications, context-sensitive to efficiently create specifications, and user-driven to create relevant specifications.

3.1 A Formal Method

Many information system specification methods are of an informal kind: information system concepts are only vaguely described, the various development steps can often be interpreted in many different ways, and system integrity of the different aspect models and their connections is not ensured. Especially the early stages of system development - requirement analysis, specification, and high-level design - are often dealt with in a very unstructured way [Cohen, 1989a]. These problems can be addressed by using a formal specification method, preferably based on some kind of mathematical specification language, so that the completeness and consistency of the specifications can be guaranteed to a much larger extent. Formal specifications can bridge the gap between informal user requirements and overdetailed source code [Lano and Haughton, 1991]. They describe what a system must do, without saying exactly how it is to be done, as is the case with an ordinary programming language [Spivey, 1989]. Completeness and consistency can be much better achieved because formally defined specifications, consisting of with mathematical precision defined requirements and constraints, enable the developer to reason about and prove the properties of the system to be described [Ince, 1988, Krause et al., 1994]. In this way, the satisfaction of many kinds of constraints can be automatically checked and methodological goals investigated.

3.2 A Dynamic Method

Many specification methods, such as those based on the waterfall paradigm, are based on the information system life cycle and aimed at producing relatively stable, if not rigid information systems. They create specifications in subsequent steps and have a well-defined time of delivery. However, due to the extremely dynamic nature of research networks, and because the methods often provide little structured support for recycling steps, it is difficult to keep system specifications up to date using such traditional methods. If research network information systems are to be maintainable and extensible, the specification method should somehow be able to keep their specifications continuously up to date.

An evolutionary or dynamic strategy is the only feasible way for the development of highly unstructured and unstable systems [Orman, 1989]. In other words, the specification process should not be seen as separate from, but instead be directly linked to the system use. This means that the specification method should somehow be linked to the information system,

so that whenever relevant system world changes take place, the method is triggered into action, by for instance querying participants and updating descriptions of the modelled network world entities.

There are many dynamic methods which are not formal, such as rapid evolutionary development [Arthur, 1992]. They are also called structured methods. Despite the large number of structured methods available, we will not concentrate on these informal methods, as they do not enforce complete, consistent and up to date system specifications, resulting in insufficient expressive and analytical power [Cohen et al., 1986].

3.3 A Context-Sensitive Method

Most information system development methods are generic, in the sense that they do not take into account the specific context in which the information system is to be used. However, as we have seen, there are many different types of information system determinants. If the detection of these determinants and their changes is not sufficiently supported by the method, the resulting specifications will not be specific enough to satisfactorily model the information system. If, however, these generic theoretical information system constructs are to be usefully related to actual working situations, it is necessary to combine them with the 'secondary domains' of analysis such as for example all kinds of psycho-social phenomena [Kensing and Winograd, 1991]. Information about possible problem domain, human network and information system constraints must thus to a certain extent be available to the method to guide the specification process. One way in which this can be accomplished is by the use of reference models, which describe determinants that are typical to a certain context and their effects on the information system [van der Rijst and de Moor, 1996].

3.4 A User-Driven Method

Most specification methods are utilized by external system developers, who analyze the organization from their own, limited professional perspective. However, empirical evidence shows that for requirements analysis to be successful, there must be many and direct interaction links between developers and users [Keil and Carmel, 1995]. Furthermore, a one-time analysis carried out by an information expert alone does not do justice to the high changeability and conceptual richness of the research network collaborative activities. Thus, the formal and dynamic method should also be user-driven. Such a systematic evolutionary strategy allows each user to independently

make small contributions at a time to the system, without having to make major system development commitments. It also allows for a highly personalized nature of development efforts, as users are much better able to customize the system to their own needs [Orman, 1989]. Thus, user participation in the specification process is even an epistemological necessity [Kensing and Winograd, 1991]. If not every user can participate, then at least a representative subset of all users should be involved [Grudin, 1994]. This user-involvement does not mean that (complementary) expert analysis can never be necessary, of course: some degree of information system development expertise may be required because many users lack the time and analytical skills to successfully perform an in-depth analysis.

Summarizing, the network information system cannot be specified just by external professionals: its potential uses are simply too many and varying and the subtleties of collaboration too great to be fully describable by analysts who are not domain experts as well. Due to the context-sensitive and dynamic nature of the research network information system specification process, allowing the users, with some degree of expert assistance, to specify and maintain their information system themselves is the only way to keep it complete and consistent. The expert assistance needs to partially be provided by external professionals, and partially be embedded in the method itself in the form of reference models. How exactly these three sources of specification inputs are to be balanced, could be the subject of interesting research.

4 The RENISYS Approach

This section describes some of the basic ideas that need to be taken into account during the development of RENISYS. Together with the methodological requirements, they provide starting-points for giving some initial assessment in this paper of the relevance of the vast literature. First, the underlying, integrating systems view is explained. Then, one of the important innovations of RENISYS is described: the reference framework. The need for such a framework is explained, after which an initial outline is presented. The section is concluded with some ideas about how the framework can be integrated with the rest of the method and applied in the specification process.

4.1 Underlying Philosophy: A Systems View

The structure and behaviour of research networks and their information systems are difficult to model, both because of their complexity and dynamics in goals, activities, organizational structure and information technologies employed. Furthermore, these networks are operating in very turbulent environments, providing them with many, constantly changing constraints and opportunities.

This leads us to the conclusion that in order to create a formal specification method, which allows its users to capture the dynamic and context-dependent nature of research networks, it is not sufficient just to have a shallow 'surface' representation of research networks, as is employed by most traditional specification methods. What is needed is some form of formal 'deep modelling' of these networks by their participants, which takes into account the context of the information systems and can easily model their inherent dynamism. This proposition is supported by the view of [Laszlo, 1992], who says that a review of the relevant characteristics of dynamical open systems, ranging from human individuals to the societies that they form can be very useful to study the impact of information technology on professional processes. Instead of just comparing specialized case studies, a full systems analysis that views these human organizations among other things as information processing systems that interact with their societal environment will be very helpful.

Thus, the research network, instead of being regarded as a rather randomly chosen set of abstract entities plus relations, should be seen as a holistic system, with a complex (1) structure and (2) behaviour, (3) embedded in an environment where it interacts with other systems, and which is (4) evolving over time. By adopting such a holistic systems view when analyzing a research network, it becomes much easier to specify complex information system contexts and their effects on the system specifications. Furthermore, it becomes less difficult to model the current network information needs, and possibly also to more accurately anticipate changes in these needs over time. In that case, we may even be able to speak of an active, specification process-initiating method, which is much more adequate and robust than current, passive methods that can only be successful when fully driven by human beings.

4.2 Why A Reference Framework Is Needed

In order to satisfy its methodological requirements, the specification generator of RENISYS must consist of two main parts: (1) a context-sensitive information system reference framework and (2) a (partially) user-driven, formal and dynamic specification method. For the second part, elements of a number of more traditional dynamic methods can to some extent be used, as described later in this paper. We will focus on the first part in this section.

One of the most important theoretical contributions that we hope to make with RENISYS, is to increase the context-sensitivity of current information system development methodologies. Many traditional methodologies are informal and produce specifications that are very hard to maintain. These problems are addressed by some of the current formal dynamic modelling methods such as DEMO and Semantic Analysis (see section 7). Their great strength is that they are formal: the concepts and techniques that constitute the method allow, at least theoretically, for the complete and consistent definition and modification of research network system concepts and their relations.

The lack of formal approach is a serious deficiency in many of the more domain-dependent network information system development approaches, like proposed in [Jarvenpaa and Ives, 1994, Agre, 1994]. On the other hand, these approaches have the advantage that they identify many of the issues unique to the specific type of network they model. Thus, they help the information system developers to concentrate on potential problems related to the context of the information system, and provide solutions in terms of reengineering activities, organizational structures, and information technologies. This is where the formal dynamic methods fail: they are too generic to be useful for the adequate definition of information systems of complex organizations like business and research networks. They can be said to suffer from the 'expressive versus procedural equivalence' problem [Thagard, 1988]. Even though generic dynamic methods may be expressively equivalent to domain-specific methods, in that their concepts can be intertranslated, this may happen at a procedurally much higher cost. Worse, using a generic method, the system developer may not know what are the right questions to ask at all.

Thus, what we need is a way to combine the formal power of dynamic modelling methods with the context-sensitivity of research network-specific approaches. Surprisingly little attention has been paid to improving the context-sensitivity of tools and methods. True, some tools allow for libraries

of concepts to be specified and reused in other applications. However, this more often than not results in a databases of ad hoc defined concepts, of which the importance and relevance is only known to their creators.

We would now like to introduce the concept of reference models [3]. Such models contain (stereo)typical knowledge about specific aspects of the phenomenon of interest that can be used to guide the development process. Good reference models (e.g. ISO standards), allow professionals to reduce conceptual confusion and provide them with solid common ground for future work. Coherent reference knowledge, created from a systems perspective, and in the form of prefabricated, yet adaptable conceptual models, can be used by developers to better structure the specification process. The models can provide them with rich information system context information and help them to focus on typical specification problems much easier than when building all specification processes from scratch themselves. The RENISYS framework forms the backbone that can contain and combine many different types of reference models. The models should be created by domain experts in collaboration with information system developers.

Several important questions need to be answered before good reference models can be constructed. What information can be considered uniform reference information and what information is to be considered case-specific? What is the relationship between thie static reference information and dynamic, network-generated specifications? How are the reference models structured? How and by whom are the reference models to be used? How can they be embedded within and used by the specification method? Who has the authority to create and modify the models?

4.3 A Basic Outline Of The RENISYS Reference Framework

The RENISYS reference framework will combine many different reference models. This section describes the general outline of the framework in which these models are stored. The framework can not only be used to drive the specification process, but also to systematically store and make accessible the obtained specifications.

The RENISYS reference framework consists of three levels: the problem domain, the human network, and the information system (fig 1). The problem domain level describes the network goals as well as the activities that the network participants need to carry out in order to achieve these goals.

[3] For a more detailed treatment of the role of reference models in RENISYS, see [van der Rijst and de Moor, 1996]

Examples of these goals and activities are the joint writing of an article and the organization of a conference.

At the human network level, the organizational structures are defined in which the activities are performed. The human network consists of four sub-levels: the individual, the group, the network and environment level. Each activity from the problem domain is represented at the human network level as a combination of abstract human information and communication processes (I/C processes), such as decision making, negotiation, and co-authoring.

Finally, the information system level describes the high level specifications of the actual network information system. The method translates each human information and communication process from the human network level into a number of human-machine and machine-machine information and communication processes. Human-machine I/C processes involve interactions where network participants are responsible for triggering computer processes, such as the retrieval of World-Wide Web documents and the filtering of database records. Machine-machine processes are triggered by the computer itself and can be completely transparent to the human information system user. Enabling information technologies can be described in the same process terms and matched with the specified requirements in order to implement the information system. This step, however, does not have priority as we focus on modelling the high-level specifications, not the actual implementation of network information systems.

Much of the context information, both that from the reference models and the actual user-defined specifications, is defined in terms of constraints. Problem-domain specific constraints limit the possible goals, activities and allowable combinations. Network structure constraints help define the network organization. Network dynamics constraints define the permitted interactions between the various types of organizational structures. Information system constraints further modify I/C processes, and can be of three kinds: system preferences expressed by the participants, information system requirement constraints, and enabling technology constraints.

4.4 Applying The Reference Framework

In RENISYS, we make a distinction between the network information system operational and development level. At the operational level, the network structures and procedures are considered fixed, only their instantiations can change. At the development level, these network concepts can be changed if necessary. In other words, the operational level concerns the modus operandi

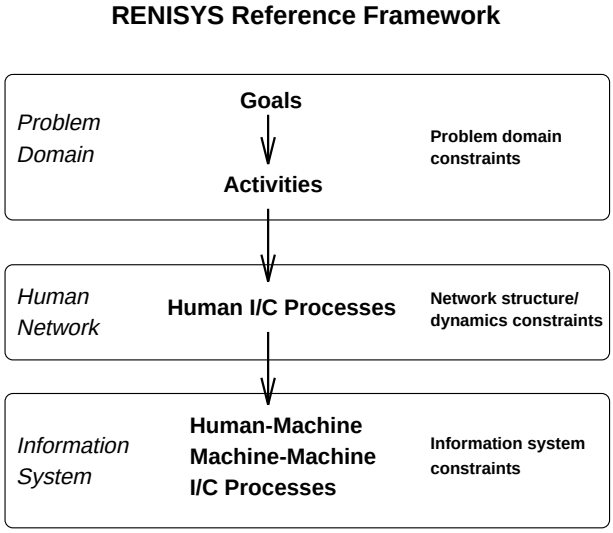

Figure 1: The RENISYS Reference Framework

of the network, whereas the development level is the meta-level at which the operational level is shaped. This means that now system developers and their activities can be specified as well, so that the development process itself can be much better formalized.

Because the method allows the information system to evolve rapidly, we must tightly integrate the specification of the system development processes with the processes occurring during actual system use. In other words, if the information system specification method is to be really useful, we cannot limit ourselves to just modeling the problem domain, human network, and information system; we must also model the processes taking place at the meta-level and especially the links that they have with components from the operational level.

4.4.1 Specification Mappings

Crucial to the RENISYS approach are the mapping processes, which are defined at the development level. Mappings can be classified along two dimensions: the temporal dimension and the complexity dimension. In each of these dimensions two types of mappings are distinguished.

Mappings in the temporal dimension are either instantaneous mappings or sequential mappings. Instantaneous mappings are the mappings that determine relationships between concepts from different subsystems at a

certain point in time. For example, an author in the problem domain can be represented as a group member in the human network. The sequential mapping takes a concept from a specific domain and determines how it and its relationships with other concepts change over time. For example, a 'send file' process at the information system level can be redefined from simply sending a file to sending a file and logging it in a sent-file database as well.

Mappings that can be distinguished according to the complexity dimension are either simple or complex mappings. A simple mapping involves a mapping of one or a cluster of concepts to exactly one other (cluster of) concepts. Sending a document in the human network is translated into mailing a text file in the information system. In complex mappings, however, a concept in one domain is translated into more than one concept in the other domain, a 1:n transformation. Here, one could think of an author in the problem domain consisting of various interacting network groups at the human network level.

It is important that the method enforces procedures related to certain types of mappings, so that for example with sequential mappings no ambiguities can be created during network specification transition, because old, now superfluous concepts have not been redefined in terms of the new ones.

5 Overview Of Relevant Information Scientific Theories

Before we can study actual applications and methods that are of potential use for the development of our method, we must first look into the general information scientific [4] theories that provide the basic conceptual means for their modelling. A large number of such theories exists, of which we will discuss only some of the most relevant ones. Systems theory helps in more naturally defining complex social system structures and behaviour. The language/action approach provides us with a framework for organizing communication processes by viewing the network as consisting of patterns of linguistic action by which people coordinate and structure their work with others. Functional grammar can capture rich context knowledge. Agent-oriented programming stresses the modelling of autonomous subjects and their internal knowledge representations, which is useful for modelling the goals and beliefs of network participants and thus also better anticipating their information needs. Mathematical specification languages specialize in

[4] We use the broad definition of information science, which thus includes concepts from communication, organizational, and other related theories

formally representing information system requirements. Logics can be used to reason about information system concepts and in this way check constraints and produce new information. Computer supported cooperative work studies how information technology can support professional collaboration processes. Together, these theoretical perspectives form a foundation on which the information systems that we are interested in and the methods that create them can be built.

5.1 Systems Theory

Complex human organizations like research networks can be characterized as systems to which the concepts of systems theory and cybernetics can be applied. Such organizations can be seen as systemic wholes which are modelled as hierarchies of interacting subsystems [Scott, 1987]. A professional network is an example of a social-technical system, which is a human-designed, goal-oriented system of persons. If the persons who comprise the network are to a large extent dependent on information technology to carry out their core activities, as is the case here, the system can be called a socio-techno system [Dietz, 1992a]. A basic premise in RENISYS is that there are numerous user-developers, none of whom has a total overview of the research network and its information system. Therefore, the method itself must actively drive and integrate, or at least guide users in the specification process as much as possible. Only when such a holistic view is taken can specification questions like what are the goals and boundaries of the system, how should they change over time, and what support can the information system offer, be partially automated.

In systems theory, a system is viewed as a set of entities with the set of relations that exist between these entities. A relationship exists if a change in the value of the property of an entity causes a change in the value of the other entity. The behaviour of a system is the complete set of values which the properties of the entities (and relations) of the system show over time [Kramer and Smit, 1987].

Cybernetics is an important part of systems theory in our framework. It tries to identify general rules for finding the causalities between the structure and behaviour of systems with the goals accepted by this system [Kramer and Smit, 1987]. An interesting cybernetical approach is research into the behavioural aspects of the system: give inputs, measure outputs, describe the structure and state of the system. This is one function in which RENISYS could excel if it were to adopt a system orientation, and which could, among other things, be used to anticipate specification changes.

Another interesting branch of systems theory may be soft systems methodology, which emphasizes obtaining a shared understanding of complex situations about which there may be considerable debate. This methodology is capable of clarifying activities to meet often only vaguely known organizational goals [Galliers, 1993].

Already, systems theory has been successfully applied in psychology to analyze problems in communication within and between human organizations. Some of the problems successfully modelled are that of metacommunication (beliefs about other participants' beliefs) and the pragmatics of communication. The system models used have allowed for effective intervention in these problems [Scott, 1987]. It is fair to say that, at this stage, much of the systems theoretical research is still too general to be applied usefully in information system development. Still, it would be interesting to investigate how these models can be used to refine and create more refined and robust RENISYS subsystem models at a later stage in our own research project.

5.2 The Language/Action Perspective

Language is a systematic way of arranging symbols, usually to express meaning [5]. Language can have different functions. One of them is to represent a domain. However, it can also be used in a wider sense: '...language is not just a formal system, but in the first place an instrument of social interaction between human beings, used with the primary aim of establishing communicative relations between speakers and addressees [Dik, 1981].'

It is this richer role that language should play in our method. We do not see a research network as a static system of passive elements, but as a set of responsible, interacting actors producing, making use of, and especially communicating numerous types of information. More specifically, we adopt the language/action perspective , which interprets organizations as patterns of linguistic action by which people coordinate and structure their work with others [Kensing and Winograd, 1991]. Thus, each language act is somehow connected with an action in the real world. As this world is ultimately represented at the information system level, each language act can be seen as an organizing device for otherwise isolated I/C processes.

To find out what such a language/action connection looks like, we must realize that every language act is situated in multiple contexts [De Michelis and Grasso, 1994]. One important effect of the use of the lan-

[5] Principia Cybernetica Project, http://pespmc1.vub.ac.be/

guage/action paradigm therefore is to formalize potential differences in interpretation by communicating parties, and in this way help determine which communication processes are sufficiently unambiguous to be automated and which not [Hanseth, 1991].

In describing such context-sensitive specifications of the information system, we clearly need to apply the subjectivist instead of the objectivist view of the world [Stamper, 1992]. The subjectivist paradigm distinguishes not one, but many different world views simultaneously. Central concepts in this paradigm are the respective agents, and the actions that they can perform on certain types of objects, based on the norm structures of the social group to which agents belong. This way of studying networks contributes substantially to the development of information systems that are better able to manage the complexity, diversity and conflicting points of view related to network development and operations. The paradigm is worked out in detail in Stamper's Semantic Analysis, which is described in section 7.4.

Natural language needs to play an important role in the knowledge representation and design of complex information systems [Weigand, 1991]. An important issue in the development of RENISYS is what degree of formality the specification languages used should have. On the one hand, natural language will make it easier to develop a user-driven method. On the other hand, natural language is often ambiguous and very difficult to interpret because of its complexity. However, a natural language-based approach does not necessarily prevent us from making formal abstractions, as long as such a formalism is motivated by linguistic arguments [Weigand, 1989]. Probably we will need to settle for a restricted form of natural language, tailored to the jargon of the specific network community, to represent network knowledge. Here, lexica and lexical management systems will be of great import.

5.2.1 Speech Act Theory / Theory Of Communicative Action

The conceptual foundation of most language/action approaches is Searle's speech act theory. In this theory, the basic unit of expression is the performance of certain kinds of language (or speech) acts instead of, for instance, a sentence. A speech act not only represents the social world, but also constitutes part of its (social) actions [Liu, 1993]. It thus forms a bridge between language and action. Each speech act consists of a proposition, that is a predication of some reference domain, and an illocution, which is the intended effect on the hearer. Each such speech, or illocutionary, act can be classified along twelve dimensions, the most important of which are the illocutionary point, the direction of fit, and the sincerity condi-

tion. Using this classification, five basic categories of speech acts are distinguished: assertives, directives, commissives, declaratives, and expressives. See for an example of using speech acts to model professional organizations [Dietz, 1992b].

The language/action approach in general and speech act theory in particular are not free of criticism. One general difficulty is the lack of locality and situatedness of the conversations [De Michelis and Grasso, 1994]. A major problem with speech act theory, according to Habermas' theory of communicative action, is that it fails to reveal what are the underlying factors that make a speech act work. [Dietz and Widdershoven, 1992]. This theory of communicative action has a better theoretical foundation than speech act theory. Habermas stresses the fundamental importance of orientation towards mutual agreement, and the primary role of regulative speech acts in the coordination of action. According to this theory, a speech act succeeds because the various validity claims which it entails, are accepted. Another proposal for improvement of the theory, especially relevant for our purposes, is to upgrade it into a theory of collaboration in large 'invisible colleges' [Foley and Pitkow, 1994].

5.2.2 The Speech-Act-Based Negotiation Protocol (SANP)

Negotiation is an important process to solve conflicts between agents that are not governed by a central authority, and as such it is a basic communication process in a research network. SANP [Chang and Woo, 1994] is a protocol modelling negotiation which is based on the speech act paradigm. It aims to flexibly automate particular parts of the process, allowing for 'meta-negotiation' about the assumptions and characteristics of a specific negotiation. Rather than basing themselves on Searle's general framework, the authors use a speech act classification developed to model real world natural language expressions. The reason for this was that speech act theory itself is too limited to capture the complexity of negotiation processes. The, partially-ordered, classification on the other hand is very useful for the analysis and construction of interactive negotiation discourses. However, this classification alone is not sufficient to construct a complete negotiation protocol: it does not contain any information about procedural aspects, such as when a negotiation stage is to start, should loop, branch, etc. For this reason, SANP makes use of existing theoretical models from the field of negotiation research.

SANP provides us with a good example of an approach for how to model various kinds of I/C processes in the human network. SANP's approach

clearly shows the importance of increasing the context-sensitivity of the I/C processes comprising our network information system: instead of merely using the few original speech act agent and acts categories, the protocol includes a wide range of ordered natural language expressions plus negotiation process models. SANP describes the negotiation process, in RENISYS similar protocols would need to be used for other human I/C processes as well, such as decision making, inquiry, etc.

An central problem remains the degree of abstraction that protocols such as SANP should have: they should be specific enough to effectively guide participants while at the same time being sufficiently general to deal with a wide variety of, often non-procedural situations. Although SANP is much more context-sensitive than general speech act theory as far as modelling negotiation interactions is concerned, problem domain knowledge is not modelled: users still need to design their own knowledge representation of the application domain. The authors acknowledge that the latter is important and are looking for some 'higher-level control mechanism' that instantiates one protocol for a specific situation. In RENISYS this problem can be addressed by accurately modelling the goals and activities in the problem domain as well as the links between this level of the reference framework and the I/C processes represented in the human network.

5.3 Functional Grammar

Whereas speech act-based language concepts can be used to effectively model patterns of goal-oriented conversations, they are not sufficient to capture the rich semantics of the universe of discourse. This is where functional grammar could play a supporting role. The functional paradigm is very similar to the language/action perspective. In it, a language, rather than as a set of formal rules to be used independently of meaning and use, is seen as an instrument of social interaction. A functional grammar now is the account of such a system of language, containing the rules that govern the structured linguistic expressions used as interaction instruments. The grammar should be driven by pragmatic rules [Dik, 1981]. These rules themselves might for instance be derived from the types of speech acts identified in a particular context. In other words, a combined speech act/functional grammar language system could result in quite a complete implementation of the language/action perspective.

Functional grammar consists of two main components: a knowledge representation language and various expression rules. The representation language is subdivided into four levels: predicates, predications, propositions,

and messages. A predicate, contained in a predicate frame, describes entity, relationship or attribute types. A predication is a predicate frame instantiated at a spatio-temporal location, which denotes a state of affairs of the universe of discourse. A proposition is a complex structure consisting of one or more predications, and designates a fact which can be true or false with respect to a specific state of affairs. A message consists of a frame of a certain illocutionary type which contains a proposition. The message can also have pragmatic functions associated with it. Thus, speech acts can easily be represented at this level. Finally, the expression rules map underlying clause structures onto linguistic expressions. This can be useful for translating to, and possibly from, natural language statements.

5.4 Agent-Oriented Programming

Approaches like speech act theory and functional grammar allow for quite an adequate general description of respectively the network environment/universe of discourse. However, they still do not pay sufficient attention to the representation of network actors, although they should not be seen as black boxes, but as responsible, decision making entities, with their own complex sets of goals, beliefs and strategies.

Shoham proposes a paradigm for agent-oriented programming (AOP) [Shoham, 1993]. AOP is a specialization of the object-oriented programming paradigm, in that its basic entities, agents, have mental components. Examples of such components are beliefs, capabilities, decisions, and commitments. Anything can be described as an agent, but the paradigm is most useful for describing entities of which the structure is mostly incompletely known.

An AOP system consists of three major parts: a formal language for describing mental states, an interpreted programming language in which to create agents, and an 'agentifier' with which to convert neutral devices into programmable agents.

As to the formal language, central to agenthood is the ability to choose among different actions. Actions can be private or communicative, and conditional or unconditional. Agents can commit themselves to other agents to perform certain actions, creating obligations. The mental state of an agent is determined by its history and current actions. The actions are determined by its decisions, which in turn are constrained by its beliefs and prior decisions. Furthermore, an agent can only perform an action if it has the capability to do so.

Based on this formal language, a programming language creates agents.

The language specifies the conditions for making commitments. Both mental and message conditions can be distinguished. These conditions can result in a priori commitments, as well as responses to messages from other agents. The resulting program now is a definition of the agents' capabilities and initial beliefs, a sequence of commitment rules, and temporal constraints.

The agentifier creates a high-level intensional program by processing a low-level process description of a machine in the process language.

This framework can make a useful contribution to RENISYS, since it stresses the importance of actors or agents having mental states being the central entities of the system. Parts of the formal language can be incorporated in the RENISYS specification languages to describe the structure and function of the network actors. However, the current AOP description of the mental state is still rather primitive in that motivation (goal) and authority aspects are paid little attention to. Another problem is that there is no distinction between different kinds of knowledge, which is partially addressed in [Verharen et al., 1994]. Furthermore, in RENISYS the agent concept will not only be used to describe machine entities, but especially to specify human agents, which may add to the complexity, as they are much less deterministic.

Other deficiencies in Shoham's model, from the perspective of RENISYS, are that it concentrates on individual agents, rather than 'societies of agents', including complex agents that consist of coalitions of individual agents. Shoham himself indicates that both social roles and social rules (global constraints) need to be effectively modelled as well to turn this 'micro-theory' into a useful 'macro-theory', which of course is one of the pillars of the systems-oriented RENISYS.

Shoham makes a distinction between a formal, class-describing, and an interpreted, instance-describing language. In RENISYS, it is debatable whether two separate languages are required for this, or that one language suffices. The lexica of the language could contain the formal concepts, analysis modules could use them to model the elements of a (sub)system of a specific research network and store the results in knowledge bases.

5.5 Mathematical Specification Languages

Most specification approaches that are called formal methods are merely sophisticated versions of various mathematical systems [Cohen, 1989b]. This view has two important implications: (1) in a successful formal method, a mathematical notation approach plays an important role and (2) such a

basis is not a sufficient condition for having an adequate method. In other parts of this paper we discuss how such mathematical systems can be utilized in the actual specification process, here we will focus on the structure of the mathematical systems themselves. We will first briefly describe two basic approaches used in mathematical languages: the functional and the object oriented approach. After that we will explain the structure of Z, which is widely regarded to be one of the best or most usable formal specification languages [Lano and Haughton, 1991], and Spec, which concentrates on describing the many different kinds of transformations that are needed.

5.5.1 Functional Approach

In human systems where requirements analysis is distributed and dynamic, and is often done by the users themselves, a functional approach is often preferable [Orman, 1989]. In this approach, the information system is seen as a collection of functions. A function maps domain data items to range items. Each function has a collection of constraints attached to it. Primitive functions can be combined into complex functions by production rules. Essential for such an approach to succeed is that it allows the user to create many small modules which can easily be combined into meaningful programs at different levels of abstraction (which is also important for describing an appropriate systems view).

5.5.2 Object-Oriented Approach

Evolutionary system specification can also be supported by the object-oriented (OO) approach. The basic idea is that objects allow changes to be made at the lowest level without affecting the specification code that calls them [Lano and Haughton, 1991]. Common OO-concepts like encapsulation, inheritance, and polymorphism seem to be well suited for use in modular information systems component specification at different degrees of abstraction. As OO-programming is an area too vast to be surveyed in this initial stage, we will postpone a more in depth investigation of the role of object-orientedness until we have produced a more detailed outline of RENISYS. However, later in this section we will briefly discuss three specification languages that have an object-oriented flavour: Z++, Object Oriented Process Specification, and Spec.

5.5.3 Z

Z consists of two notational languages: a mathematical language and a schema language. The mathematical language is based on typed set theory. Basic mathematical concepts such as set concepts, relations and functions are stored in a library. State schemas describe abstract object classes, event schemas specify the operations that can be carried out on the objects. The schema language tailors these basic concepts into more complex constructs, thus helping the developer to create specifications more easily by identifying concepts common to the domain [Woodcock, 1989].

Z is a declarative rather than a procedural language. One advantage of such an approach is that it focuses on results rather than on procedures describing how to achieve these results. Thus, an advantage of the declarative approach is that one has more degrees of freedom: instead of creating detailed algorithms, one describes high-level specifications, which can still be implemented in many different ways. Another advantage is that it is relatively easy to build the specifications incrementally.

An interesting object-oriented extension of Z is Z++ [Lano and Haughton, 1991]. The problem with ordinary Z is that a change to one of the main data structures often leads to changes in the definitions of most of the operations. The object-orientedness of Z++ helps to address this issue. Z++ allows users to specify their requirements in a restricted natural language. Another feature of Z++ is that it adopts the notion of state preservation (no changes take place unless explicitly specified). Furthermore, a lot of attention is paid to how to handle different types of specification maintenance tasks, such as corrective maintenance, environmental changes, and specification extension.

A similar OO-extension of Z is the Object-Oriented Process Specification [Krause et al., 1994]. One of the main advantages accruing from its OO-focus is that the dynamic behaviour of the complete system can be much easier modelled, which can help to realize a systems perspective on the research network ontology.

5.5.4 Spec

A system based on transformations can be an effective way of recording development histories and verification proof steps, which can be reused to provide alternative implementations [Lano and Haughton, 1991]. Although Z++ already pays some attention to transformations, it has been truly made a core concept in the Spec language [Berzins et al., 1993].

In Spec, a transformation is viewed as a function that maps (schemes of) progams onto other program schemes. The language focuses on meaning-changing rather than on meaning-preserving transformations, which have already been described in formal semantics. Specification approaches such as prototyping depend to a large extent on meaning-changing transformations. A central question which Spec tries to answer is how to ensure their appropriateness.

In Spec, transformations are characterized in terms of the vocabulary, granularity and behavior attributes of a program. The thesis of the language is that every meaning-changing transformation should be decomposed into substeps, each of which preserves two of the three attributes and makes a monotonic change in the third. Prototyping then consists of prototype evolution and production code generation. Due to the changes of specifications meaning, a verification may be necessary, especially with multiple developers, to prove that the proposed decomposition of the system meets the specifications of the entire system whenever the subsystems identified in the decomposition meet their specifications.

One major problem identified is the bad match between primitive transformation concepts and the designer's conceptual universe. According to the authors, getting a better formal understanding of the design space is the key issue for developing better transformation libraries. These libraries can be filled with schemes which are patterns of commonly occurring changes to designs. Thus, RENISYS should not only develop reference models about its different subsystems, but also about how they can be changed, in other words: the development methodology needs its own reference model as well. Such a model could then take its basic inputs from the network descriptions, and show how these can be transformed, both in syntax and in meaning.

5.6 Logics

Logics are important both as formal knowledge representation languages, and as tools to reason about the properties of these languages and the real-world constructs which are expressed in them. As there are numerous logics, it would be impossible to give a comprehensive overview. We will therefore discuss two representative examples: speech-act based deontic logic and computational philosophy of science. The main purpose of deontic logic is to model responsibilities and norms in social systems. We will discuss a speech-act based version of it, as this closely matches the language/action paradigm, our fundamental specification point of view. Another logic more specifically tailored to capturing the essence of scientific reasoning is computational

philosophy of science. It deserves some further examination as it can be used to help describe the basic communication processes of a research network.

Other logics which may be of relevance are modal logic (reasoning about the necessity of knowledge), temporal logic (reasoning about the time dimension of knowledge), and non-monotonic logic (reasoning about how old knowledge is affected by the addition of new knowledge). A good synopsis of these logics and their implementation is given in [Ramsay, 1988].

5.6.1 Speech-act Based Deontic Logic

A work process can be characterized by both the conversations participants are performing and the commitments they have accepted [De Michelis and Grasso, 1994]. Deontic logic can be used to describe the resulting normative systems. [Dignum and Weigand, 1994] propose a combination of illocutionary (speech act) logic with deontic logic to model communication processes. This speech-act based deontic logic can help to integrate the modelling of communication and commitment management. The fundamental reason for the use of speech-act based deontic logic therefore is that the coordination of human behaviour always requires some form of agreement and mutual commitment [Dignum and Weigand, 1994].

Speech-act based deontic logic consists of some basic concepts of illocutionary logic together with those of dynamic deontic logic. Typical deontic operators deal with norms such as obligations, permissions and prohibitions. The logic adopts an agent-oriented view of the information system in which roles are viewed as sets of capabilities and deontic rules.

The information system based on this deontic logic should not only formalize authorized behaviour, but also the management of authorizations. One way to do this is by creating communication protocols or contracts. These are especially useful in organizations where there is little or no hierarchical ordering between agents, as is the case in research networks. The concept is worked out in some detail in [Verharen et al., 1994]. Contracts are sets of related authorizations together with conditions on the relationship between the acts and rules governing the violation of permissions and obligations. In other words, they help to modularize the normative specifications. Contracts can be modelled as complex objects that can be specialized and generalized. Protocols can be partially based on the essential speech acts in these contracts. They can be powerful instruments for supporting research networks, as they allow the users together with the system to set goals and to determine who is responsible for what. However, they should not be constraining the network participants too much by prescribing exactly how

these goals ought to be accomplished.

5.6.2 Computational Philosophy Of Science

Computational philosophy of science [Thagard, 1988] is a logic of scientific reasoning which is considerably richer and more adequate for describing scientific discourse than normal deductive logics. The logic contains rather complex representational schemata. Basic concepts that are represented and connected are observations, laws and theories. Observations are represented in predicate calculus. General laws consist of quantified expressions in predicate calculus plus more complex representations. Theories are even more complex constructs as they generally go beyond what is observed and aim to explain the laws. A theory is not an explicit data structure, but a set of associated structures, and includes a record of its past successes. In this way it can become a flexible yet robust construct. A theory in the view of this logic cannot be rejected merely because it occasionally fails. If it is simpler and explains more important facts it can be accepted as the best explanation.

Computational philosophy of science does not aim to provide a canonical language for doing science, but rather wants to illustrate how scientific knowledge representation and argumentation processes interact. It has two main uses: problem solving and explanation. It uses complex data structures such as concepts and rules to provide considerable control in problem solving and other epistemic processes. Logical approaches it supports are, among others, various kinds of induction and abduction. Interesting is its interpretation of the meaning of concepts: they are not determined by definitions. Instead, the meaning of a concept depends on the rules that have been formulated using it, and on how it is related to other concepts by rules and hierarchical relations. Important in this logic is that it explicitly deals with the problem of 'consequential closure' [Ramsay, 1988], which concerns the difficulty of the user of a logic not being able to understand all the consequences of a set of premises because there is an infinite amount of conclusions to be drawn. Thagard heavily criticizes standard formal logic in this respect, because it does not give any criteria that can help to decide which of all possible inferences should indeed be made in a specific context:

> '[...] formal deductive logic captures so little of what is interesting about reasoning that it would be a grave mistake to take it as paradigmatic [...] The result of the mathematicization of logic has been loss of contact with the great epistemological concern about how knowledge can be made to grow. For the purposes of epistemology and philosophy of science, reasoning, not formal logic is what needs to be studied [Thagard, 1988, page

132].'

To deal with these objections against the classical treatment of reasoning, PI, the inference system which implements computational philosophy, also represents meta-knowledge about the epistemic goals of the reasoning process, such as achieving a particular explanation. An inferential system is now seen as a matrix of four elements: normative principles, descriptions of inferential practice, inferential goals, and background psychological and philosophical theories.

PI is a good example of a sophisticated logical system tailored to modelling scientific reasoning. It pays a lot of attention to both functions of a logic: knowledge representation and inferential capabilities. A research network information system, however, is much more than a reasoning machine. It supports its users in many problem-domain related group activities, of which reasoning is often only a subprocess. Thagard realizes this as he says that one of the most important extensions of PI will be in the modelling of 'group rationality', which will allow for instance the representation of problem solving in groups where individuals play various roles. Yet another problem is that users of a customized network information system cannot be forced into using the rigid and limited concepts of a language like PI. The system must thus provide for some sort of mapping between their activities and a PI-like system. Still, the structures and processes as described here could form part of the core representational framework of sophisticated system knowledge bases and help in guiding the process of group discussion.

5.7 CSCW

A research network information system has many features of a CSCW system, also called groupware. Based on the (a)synchronous form of the interaction and the geographical nature of the participants, Rodden distinguishes four classes of groupware systems: message systems, computer conferencing systems, meeting rooms, and co-authoring and argumentation systems [Rodden, 1991]. Our interest is mainly in the latter type of system, as activities in research networks mainly consist of co-producing scientific texts and structuring the required reasoning processes. An important issue in CSCW systems is the degree of control they exert on their users[Rodden, 1991]. If interaction procedures are too prescriptive, users will feel too constrained to effectively cooperate. Therefore, the system must be a semi-formal system [Lai et al., 1988]. This means that not only the information objects, but also the processing rules of the system must be modifiable by the users, which is

in line with the user-driven development philosophy RENISYS adheres to.

A problem with many groupware systems is that they focus more on supporting individuals in groups than on flexibly coordinating the overall group process [Krasner et al., 1991]. Important coordination tools that can increase the degree of strong collaboration are social protocols that govern group interaction. Empirical research has shown that these protocols can often best be determined by the group's social process, rather than by embedding them in the software beforehand [Krasner et al., 1991]. This will be much easier with dynamic, user-driven specification methods than with their more traditional predecessors. These protocols are examples of process models, which, according to Krasner, should be the basis for the coordination of the cooperation.

So what exactly does this coordination of group processes entail? Coordination can be seen as managing dependencies between activities [Malone and Crowston, 1994]. This means that it is not sufficient to just list the activities in RENISYS. Different types of dependencies need to be identified as well as the coordination processes that can be used to manage them. A language/action approach can be very useful for this purpose. An important dependency among some activities is that they are contributing toward some common goal. Therefore, goal selection and goal decomposition processes, as well as goal conflict resolution processes are important starting points for better coordination of activities. Other dependencies which need to be taken into account are for instance producer/consumer relationships, simultaneity constraints, and managing task/subtask relationships [Malone and Crowston, 1994]. An interesting topic for research is how coordination process primitives can be combined by users to solve particular coordination problems [Malone and Crowston, 1994]. This fits well with the RENISYS approach of defining activities (and their dependencies) in reference models of standard information and communication processes.

CSCW systems are moving the analysis focus from institutional organizations to cooperative work arrangements [Schmidt and Bannon, 1991, Schmidt, 1994]. Rather than trying to describe an organization in legal terms, webs of interdependent activities need to be studied. There can still be an ordering work organization (i.e. research network), but this is now considered as a constellation of deployable resources configured to meet the needs of a set of recurring cooperative activities. This is exactly the kind of approach RENISYS takes: to start with the problem domain activity analysis, and only then shaping the (net)work organization by introducing network structure and dynamics constraints.

Now that we know about the role of the CSCW system, what does it

look like? A fundamental conceptual model of groupware has been proposed by Ellis and Wainer, and will be discussed next.

5.7.1 A Conceptual Model of Groupware

Ellis and Wainer [Ellis and Wainer, 1994] describe groupware systems using three models: an ontological model, a coordination model and a user-interface model.

- **Ontological Model**

The ontological model is very important as it provides the basis for an architecture of the system knowledge bases. It can be used by the knowledge representation system to guide the construction of more detailed classes of objects. The knowledge base can then be better maintained by the system rather than by the user [Mark et al., 1995]. The ontological model describes classes of objects and the user operations. Objects are the data-structures upon which the users operate, and can have relations with eachother. Operations are transformations that act on the objects. Operations either view, create, modify or destroy objects. Users can have operation rights and object access rights. In our general RENISYS framework, we need different ontological models for the problem domain, the human network, and the information system, but also for the (meta) development level.

- **Coordination Model**

The coordination model describes the activities that each user may perform and the way in which these activities are coordinated. Roles can act as an intermediary between actors and activities. An activity is a potential set of operations that an actor playing a particular role can perform with a defined goal. A set of activities and their ordering make up a procedure. An important aspect is the temporal precedence of the activities, in which start and stop-actions play an important role. Coordination can take place at the activity level, where it describes the sequencing of events that make up a procedure, and the object level, that deals with access problems of objects by multiple actors.

The central function of a research network information system is a coordinating one: how to efficiently support the management of the various processes of its users. The degree and kind of coordination which has to take place, depends on the type of information system being developed. In RENISYS, the coordination model would consist of the processes that each actor may perform and the way in which they are coordinated. For this

model to be successful, it must be represented in all three subsystem levels. Note that according to Ellis and Wainer, activities consist of operations, whereas in RENISYS activities are further subdivided in various types of information and communication processes.

- **Actor-Interface Model**

E-W's user-interface model is focused on human-human interaction. Central in this model is the concept of views, of which three types are distinguished. First, it must allow for different views on objects, including meta-objects. Local operations are only visible to the initiating participant. Second, the model must allow for views on the involved participants, both activity-based and background knowledge, such as participant profiles. Third, E-W recognize a contextual view. This is subdivided into a structural context, a social context and an organizational context. RENISYS deals with the last two context view categories in a fundamentally different manner. Instead of giving them a minor, rather informal place in the system framework, as E-W do, RENISYS devotes a complete subsystem, the human network, to the representation of the characteristics of participants as they behave in the network.

This conceptual model is in line with [Schmidt and Bannon, 1991], who distinguish two important aspects of the cooperative mechanisms in CSCW: workflow management (coordination) and the 'common information space' (ontologies plus views). Workflow management should not focus on developing elaborate, hard-wired organizational models and procedures, but much more on supporting users in interpreting these models and changing them when necessary. The common information space does not mean a simple shared database, but rather knowledge bases where participants actively construct, debate, and resolve the meanings of shared objects. This means that the RENISYS context knowledge and specifications must somehow be systematically agreed upon, represented, and modified, based on the activities that the network participants are involved in.

6 Overview Of Information Tools, Environments, And Systems

Before we can look into the methodological aspects of the specification process, we first need to have a general overview of what components a research network information system actually consists of. In the literature, many dif-

ferent potentially relevant kinds of tools, environments and information systems are described. With a tool, we mean a piece of software that embodies a certain amount of information processing functionality. An environment consists of a set of basic software elements that can be used to construct a variety of tools. An information system in our terminology was defined as an integrated and customized collection of information and communication processes (plus data sets). Such a system is should be implemented by an integrated set of tools.

An information tool or system is often classified by the specific core human or machine information/communication process that it supports, such as decision support systems or information navigators. A research network information system is not exactly like one of these types, but consists of a context-dependent mixture of many different types of such components. At any rate, the most adequate typification that can be given is that it is an example of a groupware system.

Based on our discussion of the conceptual model of CSCW, we distinguish two basic types of CSCW system components: workflow management systems and knowledge-based systems. These categories are linked by a third type: issue-based information systems. This category is of great importance as it can be used to support scientific discussion, which is the core process of research networks. Scientific discussion can benefit in many ways from workflow management, while the results of the discussion need to be stored in knowledge bases. This knowledge, in turn, would ideally drive the workflow management (and information system development!) process. In this section, we will describe several interesting of such tools, environments and systems.

6.1 Workflow Management Systems

Workflow (management) systems provide the infrastructure to design, execute and manage business processes or activities in a network. The emphasis is on the management rather than on the automation of the tasks of which these activities are comprised [Abbott and Sarin, 1994]. Thus, central to these systems should be some form of the language/action approach where the conversations between participants initiate and control the flow of activities.

Computers can be applied in groupware in three different ways: to automate actions, support users in their intellectual work, or trigger processes [Kensing and Winograd, 1991]. Especially the latter two types have often been neglected so that most applications currently still concentrate on the

full automation of certain deterministic procedures, ignoring the support of more flexible approaches to organizing work.

Characteristic of workflow management systems is that much less planning functionality is needed than with traditional applications [van den Broek, 1995]. A process definition should therefore only serve as a guide that suggests a default flow of tasks rather than that it provides a rigid plan. A major requirement is that such a definition is flexible, so that 'process maps' can easily be modified to conform better to the current context.

Thus, the specification method using the process models must be able to accommodate the variety of behaviours occurring when performing work. A workflow system must therefore be able to adequately manage both procedural and non-procedural interactions. Procedural interactions can be modelled using conversation-based models, mainly grounded in the language/action perspective [Abbott and Sarin, 1994]. Non-procedural interactions typically occur in the case of exceptions to structured activities. Many of these exceptions cannot be formalized, but the processes in which they are to be handled, should be. Note that not all procedural conversations have to be modelled to obtain proper system specifications. For example, extrinsic conversations that are outside the direct domain of concern can be refrained from [Kensing and Winograd, 1991].

The models of these processes often are built bottom-up: from a set of process primitives more complex processes are aggegrated rather than the other way around [Abbott and Sarin, 1994]. This means that process libraries and composition facilities need to be paid a lot of attention to. One fundamental problem with this process design and execution is the level of support that the network information system (and thus the specification method) must give to its users: the well-known trade-off between guidance versus flexibility. A level of support is needed that is fully general in the sense of being able to support all possible types of collaborative interactions, while also being fully customizable to the particular situation in which interactions take place [Hartfield and Graves, 1991].

An important theme in workflow management is how to deal with changing activities in complex and dynamic professional environments. This is studied in the field of business process redesign (BPR). This approach consists of an important set of principles underlying workflow management that have in common that information technology can only be most effectively used after the supported business processes have been optimized themselves. Some characteristics of BPR are that only a small number of business processes are modified, goals of BPR are defined beforehand in terms of concrete

process improvements, and processes often transcend the individual organization [Buitelaar and Groen, 1994]. BPR has mostly been applied in traditional, hierarchical organizations [Buitelaar and Groen, 1994], so it is not exactly clear how well it can be used to manage process change in flat organizations like networks. Most BPR approaches limit themselves to describing organizational change procedures. A notable exception is the DEMO method (see section 7.3), which views BPR as a way to modify and map formal concepts from the essential to the informational level [Dietz, 1994], and thus it produces results which are much closer to the actual information system specifications.

6.1.1 The Milan Conversation Model

The Milan Conversation Model (MCM) is both a theoretical framework for understanding communication within work processes, and an implementation [De Michelis and Grasso, 1994]. It contains two core constructs: the conversation and the commitment negotiation. The philosophy of MCM is that these concepts are closely related, but that a conversation cannot simply be reduced to a commitment, because it also contains important context knowledge that determines the value of commitments.

The model therefore addresses this problem as follows. Each conversation can not only have (multimedia) documents but also several commitment negotiations attached to it. The conversations themselves can thus be completely unconstrained, all the formalization takes place in the commitment negotiations. The system makes accessible the full record of the negotiation steps, supporting documents, and the free-text conversations in which they are embedded. Note that such a negotiation is a special case of a general negotiation protocol such as SANP.

In this model, the context of a work process is considered to consist of two types: the context related to the temporal connection of events (history), and the context which brings forth the peculiarities of an event (articulation). In MCM, the conversation record represents the history, while the attached documents and commitment negotiations are said to form the articulation context. Note that the definition given of context is different from ours. We view the context as the complete set of determinants of the information system, which itself consists of the information/communication processes. Thus, in our approach, the work processes are part of the context of the information system, whereas for instance the conversations, which belong to the context in MCM, are a key part of our information system. This is in line with the central focus of MCM being the work somebody

is involved in, whereas that of RENISYS is the information system that supports work activities.

6.1.2 OVAL

Oval is an information environment that lets users become system designers by giving them end-user programming tools [Malone et al., 1995]. It builds on the well-known 'spreadsheet for cooperative work', the Object Lens [Lai et al., 1988]. Oval can be used to create radically tailorable information systems. Tailorable means that they can be changed by end users without them having to leave the application domain. Radically suggests that very large changes can be made. Thus, instead of merely changing some parameters, completely new kind of applications can be created. RENISYS definitely is a method which should support the creation of such radically tailorable systems.

One of the key problems in designing these environments is determining the right level of abstraction of the building blocks. Oval distinguishes four such elements: objects, views, agents, and links. Objects represent entities in the world. Views summarize collections of objects and allow users to edit individual objects. Rule-based agents automatically perform tasks for users. Links represent relationships among objects. Oval's primary innovation is the choice of building blocks and the ways of combining them. The main advantages of the environment are (1) that it is simple and intuitive for users to comprehend, as it uses building blocks the users can understand [6], and (2) that it offers a great amount of functionality for creating and modifying a wide range of applications. These claims are supported by some initial experiments, obtained by modelling existing groupware applications [Malone et al., 1995]. Some interesting lessons have been learnt from these experiments: such a general tailoring language allows for a great increase in scope of applications with only a modest increase in complexity, and radical tailorability facilitates the integration of tools. However, the authors admit that much more rigid empirical studies are needed before the usefulness of new applications and the usability of the environment can be assessed.

[6] An environment that aims to achieve the same goals, but is much less ambitious in scope and functionality is the Objection environment, based on CLP (http://www.uni-paderborn.de/fachbereich/AG/szwillus/objection/ constraints.html). Objection is used to specify graphical constraints to build user-interfaces. It addresses the problem of normal widget classes being too application-independent. The environment allows more specific, application-dependent widget classes to be defined that can be used to create new user interfaces in a more elegant and efficient way.

Oval mainly supports the modelling of the information system level, and is not well suited to model the complex problem domain and human network subsystems. The strength of Oval, being so widely applicable, is its weakness at the same time. For guiding its users in how to use the available functionality, some kind of formal, user-driven method like RENISYS could be of great help.

6.1.3 The Cooperator

The Cooperator [7] supports distributed groups of students who are working asynchronously on a joint paper. The core of the system's environment is the document processing facility. A discussion tool called 'issue space' aids group members to establish a shared context. This context is modelled in a graph. A browser can be used to view all the created nodes of the system. An agenda is used for the planning of non-computer mediated group meetings.

Although the issue space is based on the issue-based paradigm, the Cooperator as a whole can be considered to be mainly a workflow management system. The main focus is not on formally representing and reasoning about an epistemologically complex domain, but much more on supporting all, procedural as well as non-procedural, aspects of the co-authoring process. The issue space itself plays a relatively minor role, users of the system can even circumvent it by using the mail-facility.

The system is still under development. However, the prototype already provides us with a good example of a workflow management system with issue-based features that supports a specific research network activity, co-authoring. A full-fledged research network information system would need to be able to deal with and interconnect many more types of activities and related information and communication processes at the human network and information system level. The issue-based information system would have a much more prominent role, and would be connected with sophisticated knowledge bases that can order, store and reason about the knowledge gained during the discussion process. This knowledge could be used both by the workflow management part of the research network information system to better guide interaction processes, and by RENISYS to update system specifications.

[7] http://infolabwww.kub.nl:2080/w3thesis/

6.2 Issue-Based Information Systems (IBIS)

Traditionally, information systems design occurred from either a data, or a decision making perspective. The language/action-based business process model that is used in workflow management systems provides a third way of looking at the information system, and is more suitable for CSCW systems [van der Rijst and van Reijswoud, 1995]. Yet another point of view is of particular import to research network information system design: the issue perspective. Issues can be seen as an organizing principle of collaborative work [Hartfield and Graves, 1991]. Issues are closely related to setting and working out goals, and are thus rooted in the problem domain. An issue is represented in many different entities of the information system, and as such can be seen as providing coherence in otherwise isolated information and communication processes, and giving meaning to otherwise difficult to interpret data.

Central to this concept of issue is that it is shared among team members, in this way driving and organizing collaborative activities. Whereas the language/action perspective is focused on modelling conversational structures and processes, the issue-perspective focuses more on building the context of the actions, instead of on the actions themselves [Hartfield and Graves, 1991]. Issues thus transcend individual conversations. However, this also aggravates the problem of multiple perspectives on information, as now not only different participants in one conversation can have different views, but also there are more conversations in which different views on the same issue can emerge.

To efficiently model the process of group knowledge construction, applications using the issue-based information system (IBIS) paradigm can be used. An IBIS helps to identify questions, develop the scope of positions in response to them, and assists in creating discussions [Kunz and Rittel, 1970]. IBIS support conversations among stakeholders about complex or wicked problems, by structuring the creation and handling of 'issue nets' [Conklin and Begeman, 1988]. Issue nets have three main types of nodes: issues, positions that can be taken by potentially conflicting parties and arguments that they use to support their positions or refute those of their opponents. Many sorts of refinements of nodes and types of links between them can be added to this basic model.

A problem with many IBIS systems is that they do not actively guide the discussion in order to keep it focused on the original goals. There are no stopping rules and it is difficult to determine when a problem has been satisfactorily solved. For an IBIS to be useful, it needs to have access to good

process models of the network activities and I/C processes. For example, in the earlier mentioned SANP, the negotiation protocol provides for loops to allow agents to question eachother's arguments until the exact difference is found [Chang and Woo, 1994]. The usefulness of the IBIS will be further increased if it is somehow actively steered by the workflow management system, but no examples of such connections are known to the author yet.

Many different applications exist that somehow build on the IBIS foundation. Some of them are domain-independent. gIBIS is a graphical hypertext system to be used with rather advanced computer configurations. Its main interface elements are a browser, and a structured node index [Conklin and Begeman, 1988]. HyperIBIS is a text version of an IBIS aimed at the low-end PC market [Isenmann, 1993]. It can distinguish between deontic issues (should?), factual issues (what?), instrumental issues (how?), explanatory issues (why?) and conceptual issues (definitions).

Other IBIS are more tailored to a specific domain. This can considerably improve the modelling efficiency. Some of these systems, closely related to the research domain, will now be described in some more detail.

6.2.1 Scientific Collaboration System

One advanced IBIS that was specifically designed for research purposes is the Scientific Collaboration System (SCS) [Kim et al., 1993]. SCS pays a lot of attention to representing its knowledge in a knowledge base. It uses an ordinary object-oriented DBMS to store this knowledge and make it accessible to its users. Several new types of nodes have been defined, such as hypothesis, claim, and argument. It allows research fields to be modelled as object classes, and organizes these fields in a class hierarchy. Issue nets are then mapped to one or more of these hierarchies.

The system allows its users to make many different types of queries, broadly subdivided into queries on the structure of the topic class hierarchy and queries to retrieve specific issue nets. The first class of queries allows for example to obtain interdisciplinary viewpoints on the same problem, the second class is used to make epistemologically complex investigations of the available knowledge.

There are still many problems related to the creation of better links between SCS and the knowledge base. For example, the development of thesauri and different types of views on the same data based on the level of expertise of the user could significantly increase the usefulness of the system.

6.2.2 CLARE

CLARE ("Collaborative Learning and Research Environment") is a computer-supported knowledge construction environment [Wan and Johnson, 1994] [8]. It provides users with support for comprehending new information, relating it to already known information, and managing different interpretations of the same information. It consists of three main parts: a semi-structured knowledge representation language, RESRA, designed to facilitate collaborative learning from scientific texts, a process model for guiding the use of the representation language, SECAI, and the actual tool itself. The tool will not be discussed here as it only implements the functionality of the other two components).

RESRA ("Representational Schema of Research Artifacts") contains three types of conceptual constructs: node primitives, link primitives, and canonical forms. Node primitives consist of such scientific communication concepts as 'problems', 'method', 'theory', and 'claim'. Links like 'supports' and 'presupposes' allow complex constructs to be built. Canonical forms allow stereotypical graphs to be constructed for various types of research artifacts.

SECAI ("Summarization, Evaluation, Comparison, Argumentation, and Integration") is a valuable research collaboration process model. In the summarization and evaluation stage, individuals make their own representations of a scientific text. In the subsequent stages, a dynamic group knowledge base is created in which consensus and disagreements about issues are stored.

An important question is from what perspective and which level of detail to select the primitives. Moreover, the canonical forms are of great importance to develop a context-sensitive method: they allow knowledge patterns typical to a specific situation to be predefined. They can be the building bricks of more complete reference models. Noteworthy is the authors' stress on finding mechanisms to represent differences in opinion, this being one of the main characteristics of scientific discourse.

The authors also discuss some experimental findings. Remarkable is the focus of users on the first two, individual, knowledge modelling stages, rather than the group discussion stages. Furthermore, they note many errors in the interpretation of the texts. Therefore, for such a tool to be successful, they claim that it is more important to create process-supporting mechanisms than to develop some ideal knowledge representation language.

A major drawback of CLARE is that all the coding and comparing of re-

[8] In our terminology it can better be called a tool as it does not allow for the creation of new information processing functionality.

search conversation needs to be done manually, which means incompleteness and extra inconsistency if different coders are involved. A tool like CLARE can only be really successful if more formal approaches, for example linguistically based, help to automate part of this knowledge construction (and information system specification) process. This brings us to another limitation of the current approach: it only allows post-processing of sources. The tool would be much more powerful if it could also support the individual and group source construction process itself. If that were the case, it could better focus the discussion on achieving certain collaborative goals, something which is not supported now.

6.2.3 AEN

The Annotated Egret Navigator (AEN) is a groupware system designed to support strong collaboration among a group as its members jointly create a structured hypertext document [Johnson and Moore, 1994]. It is closely related to CLARE. Both systems make use of the same general collaborative system development environment EGRET. However, whereas CLARE is meant to collectively interpret existing scientific texts, the focus of AEN is on the construction of complex group documents. The system consists of a hypertext network of nodes and links. Both textual and graphical nodes are distinguished. The main categories of nodes are document and comment, the links offer some standard ways of coupling the nodes.

The primary navigation mechanism is that of the AEN tables of contents, which are used to construct virtual documents and can be defined in various ways. AEN supports strong collaboration, contrary to for instance free-form WWW systems, by providing an authoring mode, locking, and access control to document components. The latter consists of allowing each node owner to determine who has read, write, and annotation access to her work. One of the most fundamental benefits of AEN turns out to be the increased - dynamically defined - access to the intermediate work products of others.

The authors claim that strong collaboration can only occur through an entwining of authoring and learning. The authors realize that substantial knowledge about the state of each user must be known and propagated to other users when needed. This knowledge could for example be used for automatically setting access control rights. However, they do not make clear how these needs are determined. Furthermore, not only knowledge about individual actors is needed; context knowledge about their problem domain, groupings and information system would be very useful as well. A method like RENISYS could help to provide this important information.

6.2.4 Group Decision Support Systems: SIBYL

Group decision support systems (GDSS) are an important category of information systems, because they allow groups of professionals to easier reach consensus on a decision. Most of these systems are 'choice' support systems only, which generally ignore the important problem formulation and alternative generation stages [MacCrimmon and Wagner, 1991]. In research networks, the problem is not so much choosing between actions, but rather working out and structuring different opinions on issues, as is already done in IBIS, be it in a primitive way. Still, especially in the GDSS that focus on the early stages of the decision making process, there is considerable overlap between the two types, as both need mechanisms to represent the goals and motivations of users and structure their arguments. We are thus not so much interested in the choice characteristics of decision making processes, but would rather like to examine the role of GDSS from an IBIS perspective. One system that is typical of such an IBIS oriented approach is SIBYL.

The problem with many generic GDSS is that they do not represent the deliberation process preceding decision making, but only the results of elaborations. The average IBIS can help describe this lacking decision rationale, but is not expressive enough and does not provide enough services to the user. This problem is what the SIBYL decision support system tries to address [Lee, 1990]. The central concept in SIBYL is that of goals. The system represents and manages the goals, the alternatives for reaching them, and the arguments evaluating these alternatives. The system consists of three parts: the Decision Representation Language (DRL) that is used to represent decision processes, a set of services that provides qualitative decision support while using what is represented in DRL, and a user interface (which will not be discussed here).

The DRL uses more or less the standard IBIS concepts of claim, support, deny, etc. Interesting is how the notions of decision problem and goals are represented: a decision problem is the problem of choosing the alternative that best satisfies the goals. Relations between goals and subgoals, such as equivalence (solving the subgoals equals solving the goal), and disjunction (solving either of the subgoals equals solving the goal), as well as the plausibility of an alternative for reaching a specific goal can be represented.

Some typical SIBYL services are dependency management (keeping represented knowledge consistent, viewpoint management (representation of collections of objects that share assumptions), and precedent management. This latter service is particularly interesting as it uses goals to index past decisions. Decision problems are considered to be similar if they share goals.

The service uses the problem indices to determine which parts of the decision graphs are relevant to a certain user, and in this way helps to share knowledge across groups.

6.3 Knowledge-Based Systems

A knowledge base can be considered as a set of propositions about some domain [Weigand, 1991]. One advantage of knowledge bases over normal databases is that they contain rules and constraints that specify the permissible behaviour of the entities stored, thus allowing for a rich semantics [Weigand, 1991]. In many cases, knowledge bases are complete, independent systems, including planning and group discussion facilities. In our view of knowledge bases in the context of groupware, we have already filtered out these workflow and issue management-specific concepts.

Knowledge base theory should build on linguistic theory. To be more precise: the conceptual model of the knowledge base should make use of the deep structure of natural language expressions, and can be described by for instance functional grammar [Weigand, 1989]. When taking such a linguistic approach to information systems development, three important application areas are important: the conceptual modelling, constraint modelling, and communication modelling [Weigand, 1991]. Concept and constraint types can be stored in lexica. Specialized lexica can be derived from more general lexica making use of selection and transformation operations. Constraints can include integrity constraints, behavioral specifications and inference rules. These come on top of domain constraints already described in the conceptual model, and they can thus be considered as a logic that specifies how the conceptual model is to be used. Finally, message base specifications are formed by the set of rules that determine the processing of incoming messages and the generation of outgoing messages.

In RENISYS, conceptual and constraint models stored in lexica will be used to create the specification language of a specific subsystem. Each lexicon describes one of the network subsystems, such as the problem domain. The lexica can be part of a knowledge base, which also stores the actual specifications obtained by the analysis of the information flowing in from the (network conversation filled) message base.

Besides being grounded in linguistics, there is often a need for knowledge bases to be temporal. In many cases the time reference of information items is only implicitly present, but for example for usefully accessing archived data a temporal reference is required [Weigand, 1991]. Especially in information systems supporting professional organizations like research

networks this is often very important.

So what exactly is the role of the knowledge base in a research network information system? Of course, it can be used to store knowledge about and created by the network. Furthermore, knowledge about the information system specification goals and history itself needs to be stored.

Now that we have a knowledge-based system defined like this, we are especially interested in its connection with the workflow management system. One function of knowledge-based systems is to adapt standard workflow processes in case of exceptions [Abbott and Sarin, 1994]. However, we would like to take an even broader view. We are not only interested in how knowledge bases can be used to handle exceptions, but also in how such knowledge bases can be used to help coordinate workflow processes in general. This approach will also close the 'specification loop', creating more opportunities for automatic specification support. Our thesis is that strong links can be created between the knowledge base and the workflow management system. The knowledge based system should get a more proactive role in workflow management, by itself interpreting the state and context of the information system and initiating and coordinating workflow management and information system specification processes, concerning both procedural and non-procedural actions. What mechanisms can be used for this is probably going to be an important research topic in this thesis project.

6.3.1 Comet / Cosmos

Ontologies play a central role in the knowledge based system. Two of their functions are to make the knowledge bases reusable and to define the architecture of the evolving knowledge base [Mark et al., 1995]. Enforceable architectures are relevant domain ontologies which are represented and integrated in the knowledge base construction process. Such architectures form frameworks that guide the construction of more detailed classes. The architecture concept is discussed in some more detail in a later section.

Description logic is a general term for representation languages that allow for the manipulation of structured terms and their taxonomic relationships. An important advantage of such a logic is that it transfers part of the responsibility for the organization of the knowledge base from the user to the system because of the inferences that it allows to be made to determine the ramifications of interconceptual relationships.

Comet and Cosmos are domain-specific systems based on a generic DL-based system, called Loom. Their ontologies can be used as architectures

to guide the knowledge base construction process. In a free form knowledge base the addition of knowledge can take place rather arbitrarily. In this system, the existing ontologies predefine the way new knowledge must be represented. This is very much the idea that RENISYS should incorporate: the reference model plus network information that has already been generated to a great extent determine what needs to be specified next and how this should happen. Cosmos is an improvement of Comet in the sense that it contains more explicit rules about what procedures need to be followed for the addition of new knowledge. In RENISYS, these procedures could be stored in a Development Methodology module, which represents the development level of the method.

Some important lessons were learned during these projects. The ontologies must have a well-defined architectural status, especially when many different people add new knowledge. In dynamic worlds, ontological principles need to be established that help identifying invariants that should not change if the ontology is to remain coherent. It is not clear yet what exactly these principles are. An especially important function of the ontology-as-architecture is that it can help define interfaces with other knowledge bases.

6.3.2 CO4

C04 is a computer environment for the collaborative construction of consensual knowledge, and is therefore also called a coordinated multi-user editor [Euzenat, 1995]. It is dedicated to the incremental and concurrent building of a knowledge base, which organizes, around formalized knowledge, a set of various annotations from which this knowledge originates. The formal knowledge is connected to the informal knowledge in the knowledge base via a hypertext network. The knowledge is stored in an object-based knowledge representation system. The most important innovation of CO4 is the way it deals with the creation of new knowledge. First an individual user proposes a modification of his own knowledge base, which is handled by a confrontation query. This query checks whether the information is not in contradiction with the existing knowledge. It does this by means of the update and revision controller.

The essential characteristic of CO4 is that its knowledge bases are organized in a tree of which the leaves are user knowledge bases and the intermediate nodes are called group knowledge bases. When a user subscribes to a group base she of course can access and comment upon the group knowledge. However, the most interesting process is that of knowledge submission to the group base. After the user has submitted the proposed changes to

the knowledge base, according to some speech-act based submission protocol, the negotiation controller identifies the differences with the consensual group base. Other users can then examine the new knowledge, for instance by tentatively adding it to their private base. These users then either accept the knowledge, reject it, or propose changes for inclusion in the group base. Once consensus has been reached, the knowledge is disseminated to all subscribed bases, making use of a so-called coordination protocol.

One of CO4's strong points is its emphasis on the consistency and global coherence of shared knowledge. It differs from a standard agent-oriented approach in that it assumes that knowledge cannot be shared unless the elaboration process has been shared. That is why CO4's submission protocol can be replaced by other ones if the users so desire. This means that the rules for the group knowledge creation can be changed quite easily. It is not clear however, who is responsible for protocol changes, what causes them, and how the different protocols should look like.

Summing up, this system provides a good example of what can be gained from tightly coupling knowledge-based systems with for example IBIS, in our approach mediated by workflow management systems.

6.3.3 The Principia Cybernetica Project

One of the currently most sophisticated research network information systems with the specific objective of actually being usable by a wide community of researchers, unfortunately still a rarely satisfied constraint in most projects, is the information system being built within the framework of the Principia Cybernetica Project [Heylighen et al., 1991]. This project aims to support the evolutionary development of a system of cybernetic philosophy by an information system that 'allow[s] the construction and publication of structured, non-linear documents by a collaborative group of spatially separated contributors in a hybrid natural and formal language environment' [Joslyn, 1990].

In the development of the knowledge base representing the system of philosophy, cybernetic methods will be used. Studying the approach used here, may thus provide us with valuable insights into how to go about our own systems-theoretically based information system specification method. The current information system mainly consists of a well-structured and extensive WWW-site [9] and offers such facilities as dictionaries and a large collection of research documents provided by participants. As of yet, there

[9] http://pespmc1.vub.ac.be/

are no sophisticated groupware collaborative writing tools available, but the project coordinators plan to start work on them soon.

7 Overview Of Information System Specification Methods

To conclude this paper, we will examine several formal and dynamic information system development methods. RENISYS will to a large extent be based on concepts incorporated in these methods. The descriptions will only be brief, as more detailed analyses are underway in various projects. However, before we start examining the methods, we will consider the concept of information system architectures. As these architectures form the blueprints of any complex network information system, we think they deserve to be treated separately from the methods that create them.

7.1 Information System Architectures

In order for RENISYS to be able to specify an information system, the method must know what kind of system it is to produce. Flexible information (system) architectures can help to guide analysis efforts and meet changing information requirements [Galliers, 1993]. An architecture can be defined as a set of relationships between the essential elements of the complex system, as perceived by the 'problem owners' [Van Waes, 1991]. This means that the architecture focuses on the links that exist between high-level system concepts from the perspective of, in our case, the information system designers. According to Van Waes, an architecture can serve as a means of communication between the various actors in the design process. Moreover, it is important as a basis for the construction, use, maintenance and redesign of the information system. Thus, the architecture is the central conceptual framework that guides the development of the network information system and as such deserves a lot of attention [10].

Instead of consisting of a single layer, the information system architecture should consist of several aspect models. Numerous architectures have been developed to help modelling the information systems of professional organizations. They can be classified as ranging from charting the

[10] The information systems architecture comprises more than the knowledge base architecture described earlier. It not only deals with the structuring of the domain knowledge, but also with clarifying other design and construction aspects of the information system

organizational goals and activities to the very implementation technology-oriented aspects. It is outside the scope of this paper to try and describe all these architectures. However, we will discuss Zachman's influential Information Systems Architecture framework [Sowa and Zachman, 1992], a meta-architecture which can be used to classify the wide variety of other architectures.

7.1.1 The Framework For Information Systems Architecture

The framework consists of a matrix that combines five developer perspectives and six information system aspects from which to create the architectures. The developer perspectives are those of the planner, owner, designer, builder and subcontractor (resp. the scope, enterprise model, system model, technology model and components). The system aspects concern the data, function, location, people, time and motivation aspects (the 'what, how, where, who, when, and why' questions). The resulting taxonomy forms a framework in which all concepts of the information system development can be described and related to one another, at different levels of detail. (Parts of) existing information system representation and development methods and techniques can be mapped to specific cells in the taxonomy using conceptual graphs. This framework can make a valuable contribution to understanding the relations between the numerous, partially overlapping systems development models and techniques.

As the authors indicate themselves, the framework has not been fully worked out yet. No theoretically sufficiently satisfying criteria are given for the current row and column classification. For example, in the developer type subdivision, the distinction between the builder (who develops a technology model) and the subcontractor (who develops components) seems rather artificial. The column classification is semantically elegant, because it is based on the basic question categories available in natural language. However, this also means that too little information system-specific structure is provided by this general classification. The dependence on 'common sense' and the flat structure (all categories are considered equally important) sometimes make it difficult to efficiently map research network specific issues. Using this framework, they often must be artificially divided over different categories. For example, it is not clear where such an important concept as 'the human network' belongs in the classification. Furthermore, the framework sees for example actors as just one of many columns ('who?'), whereas in RENISYS the actors, their goals and their organization form the basic unifying concepts that integrate the system.

This lack of specificity makes the ISA not completely suitable for describing the RENISYS architecture. However, it suggests some types of (sub)architectures that RENISYS could focus on when creating its own research network information system architecture. We are mainly interested in the matrix cells described by the top three developer perspectives (scope, enterprise and system model), and the function (activities), people (roles) and motivation (goals) columns. Typical architectures found in these cells are process flow diagrams, role/deliverable architectures and goal-oriented knowledge architectures. In future work, we will select - and adapt - several of these architectures as the basis for our conceptual model of a research network information system.

7.2 NIAM / ISDM

NIAM (Nijssen Information Analysis Method) is a knowledge representation language that can be used to specify the grammar of an information system [Wintraecken, 1985]. The grammar describes the rules that prescribe the allowed states and state changes of the information base, as well as the meaning of the stored information. NIAM sharply distinguishes between the object system, represented as activity diagrams, and the information system, described in its own models. NIAM considers the activity diagrams as givens, and only models the information system. Note that this is in contrast with especially Semantic Analysis (see below), which does consider the (only) model of the organization as the specification of the information system itself.

The unit of representation in NIAM is the fact-type. A fact is an elementary proposition about a relationship between objects of the world. Each non-elementary proposition can always be decomposed into elementary ones without loss of meaning. Objects can be classified as either lexical (LOTs) or non-lexical objects (NOLOTs). Rules determine which facts are allowed. They specify object-types, fact-types and constraints. Static rules describe the allowable state of the information base at a certain point in time, dynamic rules express what state changes are allowed.

NIAM strongly advocates the use of natural language as a way of stating the problem to be modelled. As this creates a potential conflict with the demand for a formal language, natural language sentences must be expressed in a fixed structure that can only be interpreted in a single, unambiguous way. If they express elementary facts they are called elementary deep structure sentences. A NIAM analysis now consists of describing the organization in natural language, construction of population tables and identification of

LOTs and NOLOTs, identification of elementary sentence types, and addition of the described fact types to an information structure diagram.

RIDL (Relational Idea Laboratory) is a NIAM-based database engineering method. It creates an implementation-independent binary conceptual schema which is translated into a relational schema that can be implemented in traditional databases. It has already been applied in the research network domain to model a scientific conference organization activity [de Troyer et al., 1988]. Another related method, NORM, is an object-oriented variation of NIAM [Verharen et al., 1994]. The original, basic NIAM knowledge representation method has later been extended into a more complete and refined system development method called NIAM/ISDM (Natural Language Based Information Analysis Method/Information System Development Method) [Nijssen and Twisk, 1992].

The strength of NIAM is in its well-worked out entity-relation representation capabilities. Its relevance to RENISYS may be in its power to formally specify the data sets that the RENISYS I/C processes will produce and need as inputs. It does not sufficiently capture workflow management, social norms and linguistic aspects of the research network information system specification process, however. For these, we propose to use approaches such as, respectively, DEMO, Semantic Analysis, and the KISS method.

7.3 DEMO

DEMO is the Dynamic Essential Modelling of Organizations method. A detailed analysis of the potential role of DEMO in RENISYS is given in [de Moor and van der Rijst, 1995], and will therefore be omitted in this paper.

One additional remark deserves to be made here. In DEMO, the fact model constitutes the state space of the object world that the actors communicate about. The action model contains the behavioral rules of the actors. Together these models can be used to check and maintain consistency of the specifications [van der Rijst and van Reijswoud, 1995]. However, it is not clear how this should happen. Automatic methodological support at the system development level seems to be insufficient, whereas this is a precondition for a user-driven method like RENISYS.

7.4 Semantic Analysis

Semantic Analysis is an information system specification method which is developed in MEASUR. This is a research programme, which tries to provide

a range of methods and techniques for requirements analysis, systems design and systems construction which is based on Stamper's semiotic framework [Stamper, 1993, Liu, 1993]. Semiotics provides the idea of the sign as a primitive notion upon which more complex concepts like information and communication can be built. The semiotic framework adds three additional layers to the original syntax, semantics and pragmatics subdivision, namely the physical world and empirics below the syntax, and, as the top layer, the social world. In this view, the organisation is seen as the information system itself. Thus, to model an information system is to formally represent an organisation in which people use signs for business purposes.

MEASUR adopts a radical subjectivist paradigm, in which there is (1) no knowledge without a knower and (2) no knowing without action. The most fundamental concepts of the theory are affordances and norms. An affordance is a universal invariant which constitutes the repertoire of an agent's behaviour. A norm is a social affordance, an affordance which has been accepted by a community as common ground.

The core of the information system specification support provided by MEASUR is Semantic Analysis. This method classifies agents and their affordances, thus permitting the definition of powerful structural constraints on allowable ontological relationships. Both ontological and norm constraints are distinguished. Ontological constraints define the basic 'possible' world. Norms then define the deontic world, how the organization should be and behave. Besides for giving more detailed knowledge about the organization, norms can also be used for reasoning about the organization, i.e. for improving its operations.

Semantic Analysis first defines a problem in natural language, then identifies candidate agents and affordances. These steps are followed by the ontology charting, in which all the groupings are stored in a general semantic model. The last stage is the norm analysis, in which norms are defined as constraints on the realisations of the affordances. The specification language used in these analyses is NORMA (Norms and Affordances), which contains a large number of affordance types.

The ontology charts and norms can be stored in the Semantic Temporal Database. The database consists of affordances, determiners, and particulars. Determiners are invariants of quality and quantity that differentiate one instance, a particular, from another. The database can easily manage the temporal dynamics of the stored information using the database language LEGOL. This language can precisely specify norms so that they can be used as database constraints or to trigger actions. It can also be used as a data manipulation language to build applications on the database.

A major strength of Semantic Analysis is its capability to define complex domain ontologies. It forces ambiguities in concepts used to be resolved by putting them into a context of actions which are already described and understood. However, a major drawback of Semantic Analysis, which it shares with many comparable methods, is that it does not provide enough guidance regarding what are the concepts to be modelled. A related problem is caused by one of its basic assumptions: the users themselves define their own problems, in their own terminology. This in itself is a laudable starting-point, and in line with the RENISYS philosophy, but it ignores the important role that an expert analysis of informational problems can play. RENISYS, with its operational and development level reference framework, could apply a more methodological approach with respect to these problems.

A concluding remark can now be made about the difference between DEMO and Semantic Analysis. DEMO focuses on transactions, during which facts about the world are created, whereas the relationship between these facts and what can be inferred from them is of secondary importance. On the other hand, Semantic Analysis is very interested in exactly describing the way in which facts are interdependent, whereas the processes in which they come to be, receive less attention. RENISYS will try and take the best of both worlds, drawing from DEMO elements from its workflow-like approach and from Semantic Analysis the (normative) knowledge base modelling approach.

7.5 KISS

The KISS (Kristen Information & Software Services) method uses an object-oriented approach to extract information system specifications from natural language statements, which could be helpful for defining procedures to analyze network conversations in RENISYS. A KISS analysis contains three main building blocks: the information quadrant, the subject communication model, and a grammatical analysis.

The information quadrant relates the organizational functions, objects, actions and employees to eachother. It consists first of all of an organizational control model, which indicates who in the organization is responsible for the management of a business process. Second, the quadrant consists of input functions that describe who is authorized to carry out defined actions. Authorizations are given by responsible actors as defined in the previous step. Third, the information architecture defines the structural relationships between objects and actions in the real world. The architecture forms the basis for the information system and the procedures of the organization

to be modeled. Fourth, output functions allow objects to be queried about their current status and thus are used to combine and present the requested information.

The subject-communication model is used for the representation and analysis of the communication processes between the organizational actors. A subject is an object that is responsible for regulating and coordinating functions. A work-flow analysis can be carried out to improve the efficiency of the communication.

The grammatical analysis identifies the initial candidate objects, actions, and subjects by analyzing a NL text. To this purpose, a text containing the specifications is rewritten as structured sentences. Such a sentence consists of a main structure, and some additional, less important parts.

Finally, these KISS models can be implemented in concrete databases making use of various kinds of transformation rules.

8 Summary and Conclusions

In order to create more integrated and customized research network information systems, a specification method is required. RENISYS is such a method. The main difference with other methods is that it is formal, dynamic, context-sensitive, and user-driven. The method needs to draw from many different branches of information science. In order to develop an adequate method, attention must be paid to representational or modelling aspects, the information system architecture, and methodological aspects.

For modelling the many different components of a research network information system, the following theories may provide important foundations. Systems theory can be used to represent the relationships between system entities from complex domains, and to model and possibly predict their behaviour. The language/action perspective offers a flexible way of modelling communication patterns and information needs of user communities, allowing users to determine their work flows themselves, instead of having to conform to rigid prescriptions. Functional grammar offers one language that can be used to represent the rich domain knowledge that is generated in conversations between network participants. Agent-oriented programming allows us to model the motivations, beliefs and other mental concepts of network participants, thus stressing the importance of the (complex) human factor in research networks. Mathematical specification languages provide the means to strenghten the formal qualities of the

method, allowing for such properties as completeness and consistency of specifications to be guaranteed and methodological aspects like transformations and development histories to be adequately represented. Logics are also formal specification languages, but their specific strength is in modelling the reasoning about system world properties, such as determining the commitments of actors or the quality of scientific discourse. CSCW is a way of modelling cooperative work arrangements in organizations such as research networks.

The research network information system which is created using the specification method consists of three main, interrelated subsystems. A workflow management system provides the infrastructure to design, manage and execute the activities in the network. This system is used to create the context for the use of the second component, the issue-based information system. The results of the (scientific) discussions being carried out here are then stored in the third subsystem, the knowledge-based system. The reasoning component of this system regularly analyzes the available knowledge. Some results of this analysis can then fed back into the workflow management system, driving its operations, other results can be used to help users specify their changing information needs and thus update the system specifications. In this way, a kind of self-organizing network information system could be realized.

Finally, the modelling concepts must be incorporated into some methodological approach in order to be able to create and maintain the specifications of the described information system. A method like NIAM shows ways to do the actual data modelling. DEMO can be used to specify the coordination of research network activities. Semantic Analysis focuses more on defining the normative context of the information and communication processes. KISS offers a set of interesting tools and techniques that can be used to create concrete information system specifications from natural language conversations.

Many issues need to be addressed before RENISYS will be ready to be used. A more detailed overview of the role of information technology in research networks, and some core questions that need to be answered in the development of RENISYS, are presented in [de Moor, 1995]. Some initial steps on the way to a concrete implementation of the method are taken in [de Moor and van der Rijst, 1995, van der Rijst and de Moor, 1996].

This paper has highlighted many areas of research that are potentially

relevant to the development of RENISYS. It is not comprehensive, nor, being a mere survey, does it intend to fully integrate the many different approaches. In a future paper, we will give a more detailed and coherent outline of the method. Some empirical observations on the current state of the art of research network information systems will be discussed in an additional paper. A third paper will analyze the typical research network activity of the writing of a group report from an information system perspective.

References

[BDN, 1992] (1992). *Proceedings of the Biodiversity Network Workshop: Needs and Specifications for a Biodiversity Information Network, Campinas, Brazil, July 27-31, 1992.* URL: http://www.metla.fi/bin21/ws92/index.html.

[Abbott and Sarin, 1994] Abbott, K. and Sarin, S. (1994). Experiences with workflow management: Issues for the next generation. In [Furuta and Neuwirth, 1994], pages 113–120.

[Agre, 1994] Agre, P. (1994). Networking on the network. Technical report, University of California at San Diego.

[Arthur, 1992] Arthur, L. (1992). *Rapid Evolutionary Development - Requirements, Prototyping & Software Creation.* John Wiley & Sons.

[Berzins et al., 1993] Berzins, V., Luqi, and Yehudai, A. (1993). Using transformations in specification-based prototyping. *IEEE Transactions on Software Engineering*, 19(5):436–452.

[Black, 1987] Black, J. (1987). Computer conferencing: A powerful tool for the information age. In *Information, Communication, and Technology Transfer*, pages 131–134. Elsevier Science Publishers B.V.

[Brodie, 1992] Brodie, M. (1992). The promise of distributed computing and the challenges of legacy information systems. In Gray, P. and Lucas, R., editors, *Advanced Database Systems - Proceedings of the 10th British National Conference on Databases*, New York/Heidelburg. Springer-Verlag.

[Buitelaar and Groen, 1994] Buitelaar, M. and Groen, U. (1994). Business process redesign: Een nieuwe kijk op automatisering. *Informatie*, 36(6):388–397.

[Chang and Woo, 1994] Chang, M. and Woo, C. (1994). A speech-act based negotiation protocol: Design, implementation, and test use. *ACM Transactions on Information Systems*, 12(4):360–382.

[Cohen, 1989a] Cohen, B. (1989a). Justification of formal methods for system specification. *Software Engineering Journal*, 4(1):26–35.

[Cohen, 1989b] Cohen, B. (1989b). A rejustification of formal notations. *Software Engineering Journal*, 4(1):36–38.

[Cohen et al., 1986] Cohen, B., Harwood, W., and Jackson, M. (1986). *The Specification of Complex Systems.* Addison-Wesley.

[Conklin and Begeman, 1988] Conklin, J. and Begeman, M. (1988). gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, 6(4):303–331.

[De Michelis and Grasso, 1994] De Michelis, G. and Grasso, M. (1994). Situating conversations within the language/action perspective: The Milan Conversation Model. In [Furuta and Neuwirth, 1994], pages 89–100.

[de Moor, 1994] de Moor, A. (1994). The Global Research Network on Sustainable Development: Working together on our common future. GRNSD discussion paper.

[de Moor, 1995] de Moor, A. (1995). Toward a more structured use of information technology in the research community. Submitted paper.

[de Moor and van der Rijst, 1995] de Moor, A. and van der Rijst, N. (1995). Toward a dynamic, context-sensitive research network information system specification method. In Hamel, W., editor, *13th Annual International Association of Management Conference, Vancouver, August 2-5, 1995*, pages 108–117.

[de Troyer et al., 1988] de Troyer, O., Meersman, R., and Verlinden, P. (1988). RIDL on the CRIS case: A workbench for NIAM. In Olle, T., Verrijn-Stuart, A., and Bhabuta, L., editors, *Computerized Assistance During the Information Systems Life Cycle*, pages 375–459. Elsevier Science Publishers, B.V.

[Dietz, 1992a] Dietz, J. (1992a). *Leerboek Informatiekundige Analyse.* Kluwer Bedrijfswetenschappen, Deventer.

[Dietz, 1992b] Dietz, J. (1992b). Modelling communication in organizations. In van de Riet, R. and Meersman, R., editors, *Linguistic Instruments in Knowledge Engineering*, pages 131–142. Elsevier Science Publishers B.V.

[Dietz, 1994] Dietz, J. (1994). Modelling business processes for the purpose of redesign. In *Proceedings of the IFIP TC8 Open Conference on Business Process Redesign*, pages 249–258. North-Holland.

[Dietz and Widdershoven, 1992] Dietz, J. and Widdershoven, G. (1992). A comparison of the linguistic theories of Searle and Habermas as a basis for communication supporting systems. In van de Riet, R. and Meersman, R., editors, *Linguistic Instruments in Knowledge Engineering*, pages 121–130. Elsevier Science Publishers B.V.

[Dignum and Weigand, 1994] Dignum, F. and Weigand, H. (1994). Communication and deontic logic. In [Wieringa and Feenstra, 1994], pages 401–415.

[Dik, 1981] Dik, S. (1981). *Functional Grammar*. Foris Publications Holland. 3rd revised edition.

[Ellis and Wainer, 1994] Ellis, C. and Wainer, J. (1994). A conceptual model of groupware. In [Furuta and Neuwirth, 1994], pages 79–88.

[Euzenat, 1995] Euzenat, J. (1995). Building consensual knowledge bases: Context and architecture. In Mars, N., editor, *Towards Very Large Knowledge Bases - Proceedings of the KB&KS '95 Conference*, pages 143–155.

[Foley and Pitkow, 1994] Foley, J. and Pitkow, J., editors (1994). *Research Priorities for the World-Wide Web*. National Science Foundation. Report of the NSF Workshop Sponsored by the Information, Robotics, and Intelligent Systems Division, Arlington, VA, October 31, 1994.

[Furuta and Neuwirth, 1994] Furuta, R. and Neuwirth, C., editors (1994). *Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, Chapel Hill, October 22-26*. ACM.

[Galliers, 1993] Galliers, R. (1993). Toward a flexible information architecture: Integrating business strategies, information systems strategies and business process redesign. *Journal of Information Systems*, 3(3):199–213.

[Grudin, 1994] Grudin, J. (1994). Groupware and social dynamics - eight challenges for developers. *Communications of the ACM*, 37(1):93–105.

[Hanseth, 1991] Hanseth, O. (1991). Integrating information systems: The importance of contexts. In [Stamper et al., 1991], pages 133–156.

[Harrison and Stephen, 1992] Harrison, T. and Stephen, T. (1992). On-line disciplines: Computer-mediated scholarship in the humanities and the social sciences. *Computers and the Humanities*, 26:181–193.

[Hartfield and Graves, 1991] Hartfield, B. and Graves, M. (1991). Issue-centered design for collaborative work. In [Stamper et al., 1991], pages 295–309.

[Heylighen et al., 1991] Heylighen, F., Joslyn, C., and Turchin, V. (1991). A short introduction to the Principia Cybernetica project. *Journal of Ideas*, 2(1):26–29.

[Ince, 1988] Ince, D. (1988). Z and system specification. *Information and Software Technology*, 30(3):138–145.

[Isenmann, 1993] Isenmann, S. (1993). How to deal with wicked problems using a new type of information system. In Stowell, F. and West, D., editors, *Systems Science: Addressing Global Issues*, pages 367–372. Plenum, New York.

[Jarvenpaa and Ives, 1994] Jarvenpaa, S. and Ives, B. (1994). The global network organization of the future: Information management opportunities and challenges. *Journal of Management Information Systems*, 10(4):25–57.

[Johnson and Moore, 1994] Johnson, P. and Moore, C. (1994). Investigating strong collaboration with the Annotated Egret Navigator. Technical Report csdl-94-20, Department of Information and Computer Sciences, University of Hawaii.

[Joslyn, 1990] Joslyn, C. (1990). Tools for the development of consensually-based philosophical systems: A feasibility study for the Principia Cybernetica project. Technical report, SUNY Binghamton.

[Keil and Carmel, 1995] Keil, M. and Carmel, E. (1995). Customer-developer links in software development. *Communications of the ACM*, 38(5):33–44.

[Kensing and Winograd, 1991] Kensing, F. and Winograd, T. (1991). The language/action approach to design of computer-support for cooperative

work: A preliminary study in work mapping. In [Stamper et al., 1991], pages 311–331.

[Kim et al., 1993] Kim, W., Suh, Y., and Whinston, A. (1993). An IBIS and object-oriented approach to scientific research data management. *Journal of Systems Software*, 23:183–197.

[Kramer and Smit, 1987] Kramer, N. and Smit, J. d. (1987). *Systeemdenken*. Stenfert Kroese B.V., Leiden, 4th edition.

[Krasner et al., 1991] Krasner, H., McInroy, J., and Walz, D. (1991). Groupware research and technology issues with application to software process management. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(4):704–712.

[Krause et al., 1994] Krause, P., Byers, P., and Hajnal, S. (1994). Formal specification and decision support. *Decision Support Systems*, 12:189–197.

[Kraut et al., 1987] Kraut, R., Galegher, J., and Egido, C. (1987). Relationships and tasks in scientific research collaboration. *Human–Computer Interaction*, 3:31–58.

[Kunz and Rittel, 1970] Kunz, W. and Rittel, H. (1970). Issues as elements of information systems. Technical Report 131, Institute of Urban and Regional Development, University of California.

[Lai et al., 1988] Lai, K.-Y., Malone, T., and Yu, K.-C. (1988). Object Lens: A "spreadsheet" for cooperative work. *ACM Transactions on Office Information Systems*, 6(4):332–353.

[Lano and Haughton, 1991] Lano, K. and Haughton, H. (1991). A specification-based approach to maintenance. *Software Maintenance: Research and Practice*, 3:193–213.

[Laszlo, 1992] Laszlo, E. (1992). Information technology and social change: An evolutionary systems analysis. *Behavioral Science*, 37:237–249.

[Lee, 1990] Lee, J. (1990). SIBYL: A tool for managing group decision rationale. In *Proceedings of the Conference on Computer-Supported Cooperative Work, Los Angeles, October 7-10, 1990*, pages 79–92.

[Liu, 1993] Liu, K. (1993). *Semiotics Applied to Information Systems Development*. PhD thesis, University of Twente.

[MacCrimmon and Wagner, 1991] MacCrimmon, K. and Wagner, C. (1991). The architecture of an information system for the support of alternative generation. *Journal of Management Information Systems*, 8(3):49–67.

[Malone and Crowston, 1994] Malone, T. and Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119.

[Malone et al., 1995] Malone, T., Lai, K.-Y., and Fry, C. (1995). Experiments with Oval: A radically tailorable tool for cooperative work. *ACM Transactions on Information Systems*, 13(2):177–205.

[Mark et al., 1995] Mark, W., Dukes-Schlossberg, J., and Kerber, R. (1995). Using ontologies to construct knowledge based systems: Experience with Comet and Cosmos. In Mars, N., editor, *Towards Very Large Knowledge Bases - Proceedings of the KB&KS '95 Conference.*

[Nijssen and Twisk, 1992] Nijssen, G. and Twisk, F. (1992). *NIAM-ISDM: De Methode Werkt Wel!* Kluwer Bedrijfswetenschappen, Deventer.

[Orman, 1989] Orman, L. (1989). Evolutionary development of information systems. *Journal of Management Information Systems*, 5(3):19–32.

[Pimienta, 1993] Pimienta, D. (1993). Research networks in developing countries: Analysis, methodological principles and guidelines for starting.

[Ramsay, 1988] Ramsay, A. (1988). *Formal Methods in Artificial Intelligence.* Cambridge University Press.

[Rodden, 1991] Rodden, T. (1991). A survey of CSCW systems. *Interacting with Computers*, 3(3):319–353.

[Schmidt, 1994] Schmidt, K. (1994). The organization of cooperative work: Beyond the "leviathan" conception of the organization of cooperative work. In [Furuta and Neuwirth, 1994], pages 101–112.

[Schmidt and Bannon, 1991] Schmidt, K. and Bannon, L. (1991). Taking CSCW seriously - supporting articulation work. In Bannon, L., Robinson, M., and Schmidt, K., editors, *Proceedings of the Second European Conference on CSCW*, pages 7–40.

[Scott, 1987] Scott, B. (1987). Human systems, communication and educational psychology. *Educational Psychology in Practice*, pages 4–14.

[Shade, 1994] Shade, L. (1994). Computer-supported cooperative work and academic culture. *Electronic Journal on Virtual Culture*, 2(1).

[Shoham, 1993] Shoham, Y. (1993). Agent-oriented programming. *Artificial Intelligence*, (60):51–92.

[Sowa and Zachman, 1992] Sowa, J. and Zachman, J. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, 31(3):590–615.

[Spivey, 1989] Spivey, M. (1989). An introduction to Z and formal specifications. *Software Engineering Journal*, 4(1):40–50.

[Stamper, 1992] Stamper, R. (1992). Language and computing in organised behaviour. In van de Riet, R. and Meersman, R., editors, *Linguistic Instruments in Knowledge Engineering*, pages 143–163. Elsevier Science Publishers B.V.

[Stamper, 1993] Stamper, R. (1993). A semiotic theory of information and information systems / applied semiotics. In *Invited papers for the ICL/University of Newcastle Seminar on "Information", September 6–10, 1993.*

[Stamper et al., 1991] Stamper, R., Kerola, P., Lee, R., and Lytinen, K., editors (1991). *Collaborative Work, Social Communications and Information Systems.* IFIP.

[Swanson, 1993] Swanson, D. (1993). Toward a policy for managing the use of computer mediated communication in the workplace. *Interpersonal Computing and Technology*, 1(1).

[Thagard, 1988] Thagard, P. (1988). *Computational Philosophy of Science.* The MIT Press.

[van den Broek, 1995] van den Broek, M. (1995). Naar een flexibel werkstroombeheer. *Computable*, 28(22):25–26.

[van der Rijst and de Moor, 1996] van der Rijst, N. and de Moor, A. (1996). The development of reference models for the RENISYS specification method. In *Proceedings of the 29th Hawaii International Conference on System Sciences, January 3–6, 1996.*

[van der Rijst and van Reijswoud, 1995] van der Rijst, N. and van Reijswoud, V. (1995). Comparing speech act based modeling approaches for the purpose of information system development. In *Proceedings of the 3rd European Conference on Information Systems, Athens, June 1-,3-1995*, pages 353–365.

[Van Waes, 1991] Van Waes, R. (1991). *Architectures for Information Management - A Pragmatic Approach on Architectural Concepts and their Application in Dynamic Environments.* PhD thesis, Technical University of Eindhoven.

[Verharen et al., 1994] Verharen, E., Weigand, H., and De Troyer, O. (1994). Agent-oriented information system design. In [Wieringa and Feenstra, 1994].

[Wan and Johnson, 1994] Wan, D. and Johnson, P. (1994). Computer supported collaborative learning using CLARE: the approach and experimental findings. In [Furuta and Neuwirth, 1994], pages 187–198.

[Weigand, 1989] Weigand, H. (1989). *Linguistically Motivated Principles of Knowledge Base Systems.* PhD thesis, Free University of Amsterdam.

[Weigand, 1991] Weigand, H. (1991). The linguistic turn in information systems. In [Stamper et al., 1991], pages 117–131.

[Wiederhold, 1995] Wiederhold, G. (1995). Digital libraries, value, and productivity. *Communications of the ACM*, 38(4):85–96.

[Wieringa and Feenstra, 1994] Wieringa, R. and Feenstra, R., editors (1994). *Working Papers of the IS-CORE Workshop on Information Systems - Correctness and Reusability, Amsterdam, September 26-30, 1994.*

[Wintraecken, 1985] Wintraecken, J. (1985). *Informatie-Analyse volgens NIAM - in theorie en praktijk.* Academic Service.

[Woodcock, 1989] Woodcock, J. (1989). Structuring specifications in Z. *Software Engineering Journal*, 4(1).