RELEVANCE OF THE X.500 DIRECTORY TO CSCW APPLICATIONS

- Directory support for computer based group communication -

Wolfgang Prinz
GMD
Schloß Birlinghoven
D-5205 St. Augustin, West-Germany
prinz@f3.gmd.dbp.de

Paola Pennelli *
Tecnopolis - CSATA Novus Ortus
Str. Prov. Casamassima km 3
Valenzano (Bari), Italy
pennelli@ibacsata.bitnet

* at present staying at GMD

ABSTRACT

In 1988 the standardization bodies ISO and CCITT released the first international standard of a distributed Directory Service [8]. The purpose of the Directory as it is described by the standard is to supply a global nameserver and an application independent management and information service. But these applications are not exhausting the possibilities of a Directory usage. It is the intention of this paper to present the possibilities and chances the Directory offers to applications in the CSCW area.

Our investigation focuses on CSCW models and applications that support and coordinate communications in groups [15]. First the paper identifies and analyses components which most of these applications have in common. For the analysis we introduce the classification of activity oriented models and conversation oriented models. Then, after a brief introduction into the X.500 Directory Service, it is shown in which way the identified components can be represented by the Directory Service. The paper concludes with a discussion of desirable improvements on the Directory Service.

1 Introduction

A consequence of the growing number of Message Handling System (MHS) users is the need for a directory that supplies information about MHS users. In 1984 this demand led to the constitution of a working group by the standardization bodies ISO and CCITT with the task to define a distributed directory system for OSI applications and users. The result is now available as the CCITT X.500 or ISO 9594 Directory Standard [8].

The standard describes two main purposes of the Directory. The first purpose is to supply an application independent management and information service for information about OSI-applications and their users. The second purpose is to serve as a global nameserver, i.e. for the management of globally unique 'user-friendly names' for objects of the real world which are represented by entries in the Directory.

Even if the idea of a globally unique Directory defined by the X.500 standard is very new and probably unknown to most people, it can be foreseen that the Directory will become a widespread and important support tool for communication systems similar to the success of e-mail systems. Therefore, it is worth to think about the possibilities of a Directory application in the CSCW area at this early stage. This paper presents the possibilities and chances the Directory offers to applications in the CSCW area. We will illustrate how developers and researchers in the area of CSCW can benefit from considering the possibilities of the X.500 Directory.

Our investigation focuses on CSCW models and applications that support and coordinate communication in groups [15]. Henceforth, we will call these models and applications Group Communication Support Systems (GCSS). Examples of such systems and models are CHAOS [4], Lens [11], DOMINO [10], COSMOS-SDL [2] and the AMIGO Activity model [5]. They also include models [5, 6] that consider an application of the Directory for a technical support. An additional application of the Directory Service for the administration of

logical GCSS components avoids a distribution of that information among different information management systems or database applications and a unified management is achived, instead.

2 Components of Group Communication Support Systems

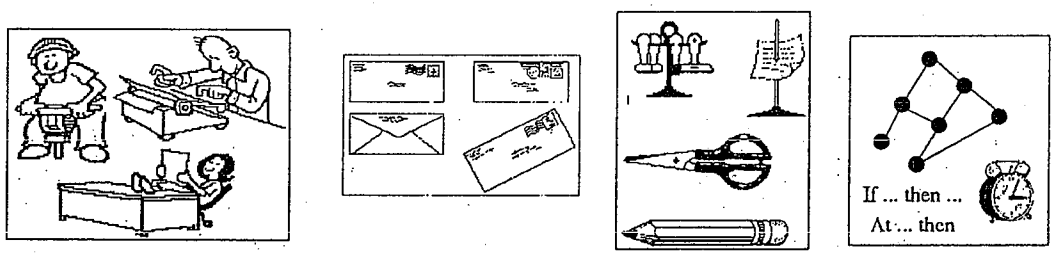
It is the aim of this paper to describe the benefits of a Directory application for group communication support systems (GCSS). To this purpose we should first identify and analyse those components which are common to most CSCW applications and which are suitable for storage and administration by the Directory service.

2.1 The Basic Building Blocks of Group Communication Support Systems

The field of CSCW comprises a large number of different applications. That is why a comprehensive investigation of all kinds of applications within this paper is impossible. Instead, we will focus our considerations on applications which support *asynchronous* communication, i.e. group-communication based on e-mail systems. Henceforth, the term communication should be understood as asynchronous message based communication. Models and applications of that type have been available since several years, and research in that area is still going on. This offers us the opportunity of analysing these applications in order to identify possibilities for a Directory support and to propose an appropriate Directory usage for future developments.

A useful basis for our analysis is a survey of group communication support systems presented in [13]. In this survey we compared models and systems developed in the context of office automation, group communication and conversation analysis (e.g. speech act theory). Since the different terminologies and terms used by the various models may lead to confusion, we used the basic components of the AMIGO Activity Model (AAM) [3] as a comparison schema. This was done by mapping components and elements of the considered models onto the AAM components which served as a platform for a uniform analysis of the different models.

An important result of this investigation is, that the elements of all considered models can be mapped onto the AAM components even if different representation methods are used. This enables us to abstract from concrete realizations and to concentrate on the AAM components illustrated below as the basic building blocks of GCSS:



Roles **Message Objects** **Functions** **Rules**

The *message object* component describes the types and structures of messages that occur in a communication process. An abstract description of the characteristics of the communicators in communication is given in the *role* component. Operations that roles perform (e.g. on messages) are described by the *function* component. It is the purpose of a GCSS to support and coordinate the exchange of messages between roles and to describe the conditions under which roles have to perform which functions. These regulations are described in the *rule* component.

After this identification of the basic building blocks of GCSS we will now have a closer look on each component in order to see whether a representation in the Directory is possible and useful.

Representation methods for the function component range from the simple naming of functions to a procedural description. Within the rule component nets (DOMINO, Officetalk-D, CHAOS) or production rules (PAGES [7], Lens, COSMOS-SDL) are used for the description of the coordination. The Directory data model does not provide means for the representation of procedural descriptions, nets, or rules. Therefore, a representation

of the rule and function component is only possible as a data stream without any semantics for the Directory. Consequently, a Directory representation of GCSS elements that realize these components is not of interest.

For the analysis of the role and message object component we distinguish between **activity oriented** systems and **conversation oriented** systems. We will see that the systems and models[1] of each group use different representations of roles and message objects.

2.1.1 Roles and Message Objects in Activity Oriented Models

Activity oriented systems are designed to support the cooperative execution of a goal oriented task. Characteristic of such tasks is that the regulations which structure the communication process can be described in advance. We will call the communication process described by the regulations an ***activity***. Most of the systems developed as office procedure systems belong to this group. An example for activities, such systems are able to support, is a travel application procedure or a procurement procedure.

The models of this group use the role concept as a means for an abstract description of the function and behaviour of a communicator within an activity. The description is independent of the individuals or automata that fulfill the roles. For example the roles of a date-planning activity [16] are *initiator* and *participant*, for a travel application activity the roles are *applicant, authorizer,* and *travel agent.* Each time an activity is instantiated, a communicator (e.g. a person or an automaton) is assigned to that role, i.e. a new role instance is created. One communicator may play several roles as well as one role can be fulfilled by several communicators. Role assignments may also change during the course of an activity. Several instances of the same activity can be executed concurrently and for each instantiation of an activity another role may or must be chosen.

The lifetime of a role starts with the assignment of a communicator and ends either with a reassignment or the termination of the activity. Within its lifetime a role can communicate with other roles of the same activity instance using instances of message objects.

Message objects of activity support systems mainly represent office forms like a travel application or a procurement form. The messages are typed and their content is structured according to the different office forms.

The definition of the roles and message objects of this group are activity specific. For each activity different roles and message types must be defined and their definition is meaningful only in the context of the activity in which it appears. Since the roles and message objects defined by models of this group are only referable in the activity-specific communication, we call them ***communicational roles*** or ***communicational message objects*** respectively. Addressing of a communicational role in a communication outside of the activity is not possible.

2.1.2 Roles and Message Objects in Conversation Oriented Systems

A user support for everyday conversations which are independent of a specific activity is provided by models and systems we want to classify as conversation oriented systems. For example, these systems automatically arrange the user mail according to its context or they propose appropriate message types for the next communication act within a conversation according to predefined rules. In most cases these rules are derived from the speech act theory, such as in CHAOS and Coordinator [18].

Other systems, e.g. Lens, make the attempt to support the user in filtering and sorting messages as a means to overcome the information overload and to eliminate junk mail.

Common to all of these systems is that they do not apply the role concept. The purpose of the systems is the support of users in individual communication. Therefore, it is not desirable to adopt the role concept as it is used by the activity oriented systems as a means to prescribe the behaviour of a communicator in his/her communication. This technique is only required for activities where communicators must behave in a prescribed way in order to fulfill their task within the activity

Nevertheless, we want to propose an adaptation of the role concept that makes it also useful for conversation oriented models. The adaptation requires a specification of roles which is independent of a particular activity. Roles must be defined as components of the environment in which the conversations between the communicators take place. They should represent the status and function of the communicators within that environment. For the communication applications, we consider this environment to be determined by the organization. Consequently,

[1] The survey considers models as well as realized systems. Henceforth, we will use the terms system and model as synonyms when we mean both, realized systems and theoretical models.

we will call roles of that type ***organizational roles*** to differentiate them from activity specific communicational roles. Examples of such roles are *boss, project-leader, project-member, or secretary.*

In contrast to a communicational role, the assignment of a person to an organizational role is independent of a particular activity. The lifetime of the role and the role communicator assignment depends on the organization in which the org. role is defined. This assignment will not change very often. Henceforth, there will be only one set of role instances, i.e. communicators assigned to a role, in contrast to the communicational roles where each activity instance results in an additional set of role instances. Therefore it is possible to specify the name(s) of the person(s) who fulfill the org. role by the very role description. For communicational roles, instead of names of persons, predicates that express constraints and requirements may be given. The role player has to satisfy them in order to play the role as expected. These requirements can be used for the selection of a role player as well as a for a qualification test of a given person. Alternatively, names of organizational roles can be specified as role players. The role player selection is then restricted to those communicators that are allowed to play the organizational role. In that case a person is bound to a communicational role via the assignment to an organizational role.

Another advantage of the role concept is that persons can be addressed by their role name. In organizations this is useful because in most cases people do not know the names of the people that fulfill the different roles but they know the names of the departments and the roles within that department, e.g. finance dep. → cashier or project xyz → project leader and project members. In the last example, one role name denotes a group of people (project members), i.e. mailing a message to the role project member is like mailing it to a distribution list.

In almost every organization, formal or informal communication regulations exist by which people communicate which each other. These regulations determine who may communicate with whom for which purpose. Such regulations express, for example, that an employee has to consult his/her boss first before s/he negotiates with other departments, or that s/he has to ask the secretary first before s/he goes into the office of the director. Such regulations, which are expressed over the organizational roles people play, do not exist for electronic communication [12]. Therefore it happens, that users violate the communication regulations and etiquette either by intention or in the most cases unintentionally because they simply do not know the regulations. An inclusion of such regulations into communication support systems would increase the level of support and would help the user to behave more adequately, i.e. according to etiquette. A first attempt towards that direction was made by Tsichritzis and Gibbs in [19].

Conversation oriented systems offer the user a predefined, and in some cases extensible, set of structured message types. The different message types are composed according to the forms that exist within an organization. Consequently, we will call such messages ***organizational message types.*** Examples for organizational message types are *short-notice, inquiry, commitment.* Both techniques, the typing and structuring of messages, shall help the users to select an appropriate message type for communication and to fill out the message with the required information. Furthermore, these techniques allow a message type and content based filtering, arranging and sorting of incoming messages.

It should be noticed that the set of organizational and communicational message types is not disjoint. For example, a travel application might be either an organizational message type as well as a message type used within a travel apply activity. Messages, that initiate an activity always belong to both groups. When they are initially sent the activity has not yet been started. After the acceptance by an activity support system, the activity is initiated and from this time on its type can be regarded as a communicational message type.

2.2 Requirements

A comprehensive user support that includes support for cooperative activities as well as a support for individual communication requires a combination and integration of activity oriented and conversation oriented systems. An important integration aspect is that common and interrelated components of the different systems are managed and accessible by only one database management system (DBMS) in order to avoid multiple and inhomogeneous representations of the same information. So far, each system uses its own application specific database or knowledge base system. In the following we will list the most important requirements a general database should fulfill.

a. Roles and message types are objects with specific properties. Therefore, the database should provide means for the storage and management of objects.

b. For the description of organizational roles it is important to express its org. context, i.e. the department to which the role belongs and its hierarchical level. A DBMS that provides means for a hierarchical ordering of entries would be suitable for that purpose.
c. Org. and comm. roles are fulfilled by people or automated services (role players). The binding of a role to its role players is described by naming the role player or by describing possible role players using a predicate. Additionally, bindings from role to role are possible. Therefore, the DBMS should allow the description of relations between entries. Furthermore, it is important that the database stores all employees, services, roles and message types, etc. of an organization. Different and inhomogeneous DBMSs each of them storing a different subset of the organizational information are not practicable for that purpose.
d. Communication systems are mostly distributed systems and organizations are very often locally distributed as well. Consequently, one should choose a distributed database that allows the storage of information at the places where it is produced and managed.
e. The information of large organizations will not be managed by a single authority but by all those authorities being responsible for the information. That entails the need for a distributed management of the database information.
f. Communication and cooperation is not bound to the borders of an organization. A database that additionally provides information about other organizations and their employees, roles, etc. would well support the cooperation of different organizations.

After a brief introduction into the X.500 Directory system in the following chapter, we will show in which way the Directory satisfies the previously listed requirements.

3 The X.500 Directory

A Directory System (DS) is a facility for storing and maintaining information about various objects in a distributed system. Such objects may be users, resources or services available, either within a single network or within several interconnected networks. The Directory is a distributed service offering a distributed administration of all information it contains. A DS hides changes in the network from the user because it enables him/her to refer to an object by name instead of by network address, for example. It also provides a more user-friendly view of the network so that users have an efficient method of referring to network information. In the following we will briefly describe the information model and the services of the Directory. A more comprehensive description of the Directory can be found in [1, 17].

3.1 The Directory Information Model

The information model describes the logical structure of the directory information. Each object of the 'real world' known to the Directory is represented by a so-called *(object) entry*. The collection of all object entries make up the *Directory Information Base* (DIB). An object entry contains a specific set of properties of a 'real world' object which are represented by several attributes of the entry. Each attribute consists of an attribute type, indicating the type of information represented, and one or more attribute values containing the information.

In order to avoid a multitude of different types of entries and to provide a guiding schema for the representation of 'real world' objects by Directory entries, object classes are defined. An object class is an identified family of 'real world' objects which share certain characteristics. The definition of object classes is managed by the *Directory Schema*. The schema consists of a set of rules that specify and control the structure of information. In particular, an object class describes and controls the contents of entries which belong to it in terms of mandatory and optional attributes.

In order to allow a distributed administration of the DIB, the object entries are arranged in a hierarchical structure called the Directory Information Tree (DIT), where the nodes of the tree are the Directory Entries. As the DS operates in a distributed environment, the DIT can be fragmented and allocated to different servers called Directory System Agents (DSAs). The DSAs, each of them only knowing and handling a part of the global directory information, cooperate to perform operations, assuring that a user can navigate through the global DIT and that s/he can access each entry in the DIT from each site.

Each entry in the DIB is labeled with a unique Relative Distinguished Name (RDN). A user can refer to an object entry through its "Distinguished Name". The Distinguished Name of an entry is the sequence of the Relative Distinguished Names of all superior entries and its own RDN. In this way, each object entry can be

reached by a unique name, which is the pathname from the root of the tree to the object entry. An example of a DIT and of Distinguished Names determination is shown in the following diagram:

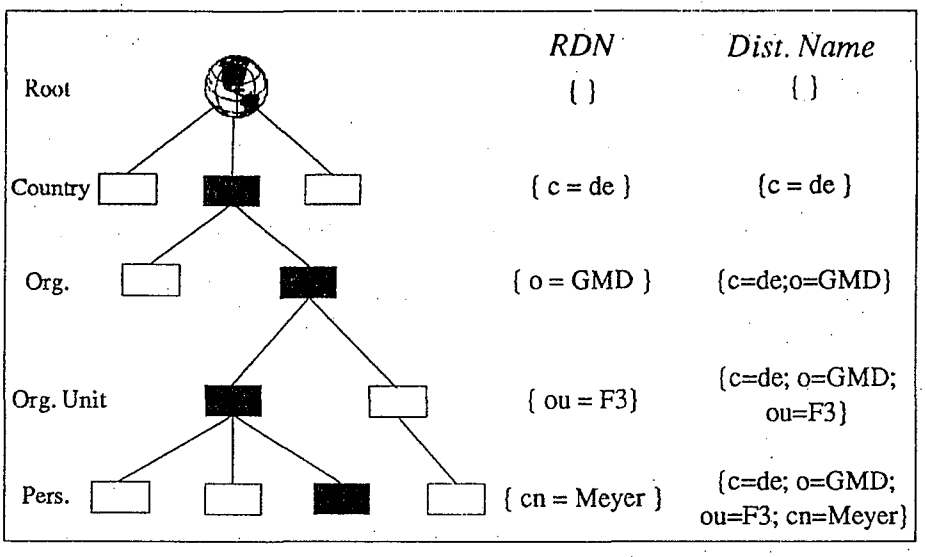


Figure 1: Determination of Distinguished Names

A single object may have several alternative names in the system. These alternative names are called "Alias Names". The alias is another entry in the DIB which points to the object entry and permits to reach a single node in the DIB via another path.

3.2 The Directory Services

According to their purpose the Directory operations are divided into three categories. Each category is described by one *port* that offers appropriate operations. It is distinguished between the Read Port, Search Port and Modify Port.

The Read Port supports the retrieval of information associated with a particular entry. That means that, by the Read Port, the Directory supplies a name to properties mapping service *(white pages facility)*. Additionally, a compare operation is provided that compares a supplied value with the values of a particular attribute type in a specified entry.

The Search Port enables a user to explore the DIB, looking for information associated with a particular object (browsing facility). Moreover, this port provides a yellow pages facility, that consists of looking for all the objects in the DIB, or in a subtree of the DIB, which match a particular set of attributes.

The Modify Port supports all modify operations. These are operations to add, to remove, or to modify an entry or to change the name of an entry.

As shown above, the Directory Service provides a variety of retrieval operations. These operations facilitate the access to and the usage of information contained in the DIB supplying "white pages", "browsing" and "yellow pages" facilities. Moreover, the Directory System offers the basic operations to modify the DIB in conformance with the Directory Schema.

4 Supporting GCSS by the Directory

Before we describe the Directory application for GCSS support in detail, it is shown that the X.500 Directory satisfies the requirements listed in section 2.2. Whenever one of the requirements is affected by the following considerations this will be indicated by refering to the appropriate item marker (a-f).

The Directory is a general and standardized repository of information. It provides all advantages of a standardized database in contrast to a normal database management system. In particular, a normal database is designed to manage its own copy of data and to use different data access services. Sharing data among different applications is often impossible. The Directory, instead, allows the use of a single, coordinated and standardized database and it provides a single set of basic services to manage that data repository. It offers, in a standardized way, the look-up, browsing and yellow pages facilities, so it can be used by any other application which requires the functionality provided by these services.

Real world objects are represented in the DIB according to the Directory data model as object-entries. Since these entries allow the representation of the characteristic properties of a real world object by attributes, requirement (a) is satisfied. Furthermore, the hierarchical naming schema (Fig. 1) enables a hierarchical ordering of entries (b).

The extensibility and flexibility of the Directory data model makes it possible to adapt the object classes in a way that the characteristic objects of an organization can be represented in the Directory. Since each entry is denoted by a unique name, relations between two entries are easily described by assigning the name of the referred entry as an attribute value to the refering entry (c). The type of the attribute determines the semantic of the link. Examples for this technique are given in Tables 1 and 2.

Another important aspect is the distributed management offered by the Directory. The Directory is distributed and provides distributed administration facilities for the information it contains (d,e). In particular, if the Directory is used for the storage of the logical components of the GCSS, such as Roles and MessageType, it will support the handling and distributed management of these components. Moreover, in conjunction with an X.400 MHS as the underlying communication media, the distributed administration of the network and the registering of users involved in a group communication application can be supported by the Directory. In this way, we can note that the use of the Directory System supports the implementation of group communication at two levels. The first level is given by management support for logical components of the GCSS and the second level is represented by the support the Directory offers to the communication service a GCSS is based on in order to store technical communication information (address, etc.).

The information stored in the Directory is organized according to the Directory Schema which is different from a normal database schema because it is open. Open means that each administrative authority establishes that part of the schema which applies to those portions of the DIB administered by the authority (e). So, it is possible to have different rules and object class at different sites. That is very suitable in the context of group communication applications, because the different organizations, represented in the Directory, can thus define their own internal structure in a quite independent way.

Finally, we want to point out the information service that the Directory offers to users and to other services. As the Directory is intended to serve as a world-wide distributed service, any user and service, stored in any site, can use the facilities that the Directory provides for retrieval purposes (f). In particular, a generic user inquiring the Directory can obtain information about his/her own or another organization, about the roles that s/he or another user may occupy and about available services and information required to access them.

These considerations show that the Directory satisfies the requirements listed in section 2.2. The following chapter will illustrate how to represent roles and message types as entries in the DIB.

4.1 Representation of GCSS Components by the Directory

The Directory provides a predefined set of object classes for the representation of 'real world' objects. The following considerations describe in which way these object classes can be used for our purposes and which extensions are necessary.

4.1.1 Representation of Roles

In chapter 2 we distinguished between organizational roles and communicational roles depending on the environment in which they are defined. Accordingly, two different object classes for the representation of roles as entries in the Directory are required.

Previously we defined organizational roles as an abstract means to describe status and function of employees within an organization. An org. role can be described by the properties: name, description, collocation, telecommunication information, function, status, and responsibility of the role and the role player. For the communication context, the function, status, and responsibility of a role can be expressed by listing the message types the role may use and the activities the role may start or be involved in. A more detailed description of the rights and responsibilities of a role can be given using access control specification means. Since an explanation of that technique and its application would exceed the framework of this paper, the interested reader is referred to [14]

Further important pieces of information needed to characterize an organizational role are the names of the employees who can occupy the role. Although one role can be fulfilled by several employees, the DIB will contain only one entry per role. In each entry representing a role there will be an attribute that contains the names of those employees that can play the particular role.

The Directory standard already defines an object class for organizational roles. For the representation of the listed properties this object class must be extended. The following figure lists the attributes of the extended organizational role object class in ASN.1 notation. Additional attributes are typed in *italics*.[2]

```
OrganizationalRole OBJECT-CLASS
        SUBCLASS OF Top
                MUST CONTAIN{
                        commonName}
                MAY CONTAIN{
                        description,
                        localAttributeSet,
                        organizationalUnitName,
                        postalAttributeSet,
                        telecommunicationAttributeSet,
                        preferredDeliveryMethod,
                        roleOccupant,
                        seeAlso,
                        messageTypes,
                        activities,
                        startActivities}
```

The name of a role is stored in the attribute *commonName*. The *local*, *postal* and *telecommunication* attribute sets contain attributes for storing the description of the location of the role. The *organizationalUnitName* attribute stores the department to which the role belongs. The preferred delivery method for messages (e-mail, fax, paper mail) is stored in the corresponding attribute. The *seeAlso* attribute can be used to reference similar roles.

The attribute *roleOccupant* contains the Directory names of the people who are allowed to play that role. The attributes *messageTypes*, *activities*, and *startActivities* store the Directory name[3] of entries that represent either message types or activities. These attributes establish links between different Directory entries.

The following table shows an example of an entry of the OrganizationalRole object class that describes the role of a project leader.

commonName	***Project Leader***
description	Leader of the CSCW project
...	...
roleOccupant	{...; cn=Smith}
seeAlso	Project Member
messageTypes	shortNotice, travelApplication, procurementApplication
activities	travel, procurement
startActivities	travel, procurement

Table 1: Example for an entry of the ***OrganizationalRole*** object class

2 The correct technique for the extension of object classes is the definition of a *subclass*, but for simplicity we have chosen this technique.

3 In following examples we use a descriptive name instead of a Directory name in order to make them easier to read and understand.

4.1.2 Representation of Message Types

In chapter 2 we also distinguished between organizational message types and communicational message types. Organizational message types were characterized as general message types existing in an organization which are used by users in daily work, independently of a specific activity.

The definition of organizational message types includes the assignment of a name, a description of its usage, and information about its organizational context (organizationName, organizationalUnitName for messages types which are used only in an specific department, e.g. the message type *software notice* of the software department). Moreover, the structure of the message type, that is the set of attributes composing the message itself, and the list of roles which can use this message type must be defined.

Based on these considerations the OrganizationalMessageType object class is defined as follows:

```
OrganizationalMessageType OBJECT-CLASS
                SUBCLASS OF Top
                        MUST CONTAIN{
                                messageTypeName}
                        MAY CONTAIN{
                                description,
                                organizationName,
                                organizationalUnitName,
                                structureDefinition,
                                roleList,
                                seeAlso}
```

This object class specifies the message type name and the purpose of that message (description). Apart from the information about the organization and organizationalUnit in which the message type is defined, there are two important attributes *structureDefinition* and *roleList*, which contain a description of the structure of this message type and the names of all roles that can handle it, respectively. The *seeAlso* attribute can be used for references to other message types which have a similar purpose. The following table shows an example of an entry representing an organizationalMessageType:

messageTypeName	***short-notice***
description	This message type should be used to exchange inofficial notices.
organizationName	Fantasia
structureDefinition	Message Attributes: - sender, recipient, date, subject - reference: your letter, telephone-call - with the request for: call-back, for your information, etc. - ...
roleList	all roles
seeAlso	notice, software-notice

Table 3: Example of an entry of the *organizationalMessageType* object class

Communicational message types, however, are message types defined and used within a particular activity. In order to represent these activity specific message types as entries in the DIB, we need to associate with them the message type name, a description concerning the content of the message, the activity in which it is defined and used, the structure of the message itself and, finally, a list of roles that will receive this message type during the course of the activity. The following object class is used to represent communicational message types:

In the previous chapter, we spoke about the Communicational Roles as the roles existing only in the context of a particular activity. They are activity specific and they permit variable roles assignments. So, this kind of roles is characterized by its name, its description and by the activity it is located in. It is also necessary to describe, in a variable way, the people who can occupy the role and to define the kind of communicational message types it can handle and its responsibilities within the activity.

The Directory standard does not define any object class for the representation of communicational roles. So we should now define a new object class to be used to represent these roles. The definition of that object class is as follows:

```
CommunicationalRole OBJECT-CLASS
        SUBCLASS OF Top
                MUST CONTAIN{
                        commonName}
                MAY CONTAIN{
                        description,
                        activity,
                        owner,
                        commMessageTypeList,
                        responsibilities}
```

The *commonName* attribute specifies the name of a role. The *description* attribute allows a general description of the role. The *activity* attribute indicates the particular procedure which the role is defined in. The *owner* attribute contains a description of the person who occupies the role during the activity; that attribute does not contain an employee name because the owner of the communicational role is different for each activity instance. For example, values for that attribute can be organizational roles or descriptions like the following: 'The user who initiates the activity'. The *commMessageTypeList* attribute contains the message types which this role can handle during the procedure. Finally, the *responsibilities* attribute describes the responsibilities of that role within the activity.

The following table gives an example of an entry representing the *initiator* communicational role, defined within a date-planning activity.

commonName	***initiator***
description	Initiator of a date planning activity
activity	Date planning
owner	sender of the date-planning-init message
commMessageTypeList	date-planning-init date-request date-proposal date-decision date-cancelation
responsibilities	decide about a proposed date, authorized to send a date-cancelation

Table 2: Example for an entry of the ***CommunicationalRole*** object class

The proposed object classes should be regarded as an example. The application of the Directory by a specific GCSS might require modifications and extensions to these object classes.

In this way it is possible to represent roles as entries of the object classes organizationalRole and communicationalRole in the DIB. The DIB is organized as a tree structure (the DIT) that represents geographical and/or organizational dependencies. Consequently, it is necessary to determine the place of entries of both object classes within that tree (see Fig. 2). Conforming to the Directory schema entries of the organizationalRole object class are immediate subordinates of entries of the Organization object class or OrganizationalUnit object class. The level at which an entry is added to the DIT represents the status of the role within the organizational hierarchy. An organizational diagram can be chosen as a guideline for the design of the part of the DIT that represents the organization. Communicational roles are defined in the context of an activity. Consequently, they are immediate subordinates of the entry representing the corresponding activity. (see [14] for a discussion about the representation of activities).

```
CommunicationalMessageType OBJECT-CLASS
        SUBCLASS OF Top
                MUST CONTAIN{
                        messageTypeName}
                MAY CONTAIN{
                        description,
                        activity,
                        structureDefinition,
                        roleList,
                        seeAlso}
```

This object class contains a subset of attributes available in the OrganizationalMessageType object class. A new attribute is associated to the entries belonging to the CommunicationalMessageType object class. It is the *activity* attribute. This attribute is peculiar to this object class, since it specifies the particular activity the messageType is defined in. The following figure shows an example of a communicationalMessageType entry:

messageTypeName	***date-planning-init***
description	This message type is used to initiate a date-planning activity bu sending an instance of this message to the date-planning mediator.
activity	date-planning
structureDefinition	Message Attributes: - sender, recipient, date, meeting name - meeting description, date, time - participants
roleList	initiator, participants
seeAlso	date-request, date-proposal, date-decision, date-cancelation

Table 4: Example of a ***communicationalMessageType*** entry

In this way, all message types can be represented as entries in the DIB. The organizational message type entries will be immediate subordinates of entries representing the organization or organizational units, according to its definition context. Entries representing message types are leaf entries in the DIT. Since the communicational message types are only meaningful within a particular activity, the corresponding entries will be immediate subordinates of an entry representing the activity in which they are defined (see also communicational roles).

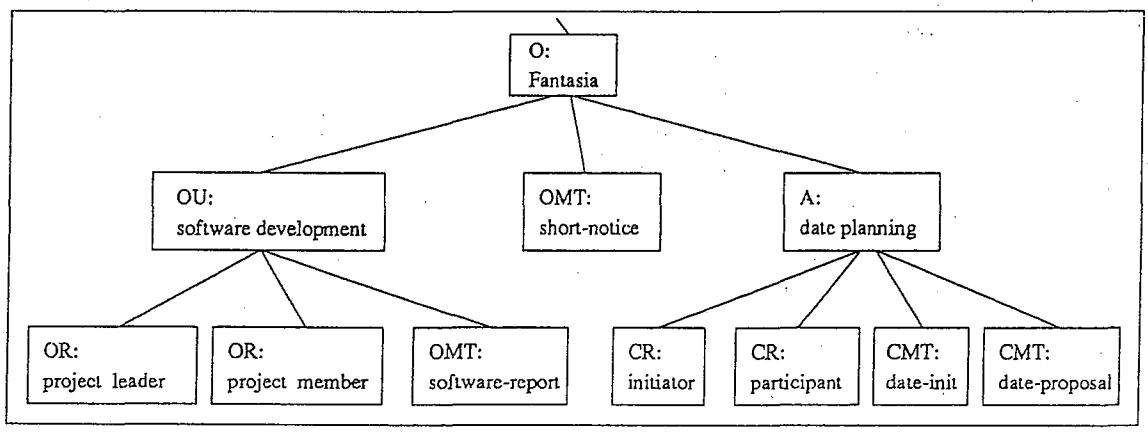


Figure 2: Subtree of the DIT showing entries representing the organization (O), organizational units (OU), activities (A), organizational roles (OR), communicational roles (CR), organizational message types (OMT), and communicational message types (CMT).

Figure 2 shows a subtree of the DIT that illustrates the placement of organizational and communicational roles and message types. The subtree shows an entry representing properties of the organization *Fantasia* and the organizational unit *software development*. Placing the entries for the org. roles *project leader* and *project member* as subordinates of that entry indicates that these roles belong to that specific department. Only within this department the organizational message type *software report* is used. That is why the corresponding entry is placed in this subtree and not directly as a subordinate to the organization entry as done for the org. message type *short notice*. The entry *date planning* represents the date-planning activity. Subordinates of this entry are the communicational roles *initiator* and *participants* and the comm. message types *date-init* and *date-proposal*.

4.1.3 Application of the Directory Information

Above, we explained how to represent the GCSS components by the Directory. In the following we will illustrate how this information can be applied. Three different types of applications can be distinguished: administration support for a single application, application overlapping info-service and user-infoservice.

The application of the Directory for the administration of the roles and message type description releases developers of GCSS from design and realization of their own databases. The distribution of the Directory is especially of advantage for distributed group communication applications, because Directory information is accessible in the same way from different sites. Besides the administrative support the Directory provides possibilities for the description of relations between entries by name-links or predicates. An example for that is the selection and assignment of role-players. A role to role-occupant relation can be described either in static way by naming the role-occupants with their Directory-names (org. roles) or by the description of predicates (comm. roles). That is possible because the Directory allows the representation of heterogeneous objects in a unified and standardized way and because it provides a unique naming schema for all entries. Heterogeneous databases, each of them storing different pieces of information (e.g. about roles or employees) cannot provide this functionality.

The Directory is an application independent database offering its services to different kinds of (group communication) applications. Very often the information used and managed by different applications are the same. For example, O/R addresses, message type or role descriptions can be used by interpersonal mailing services and activity and conversation oriented GCSS as well. Consequently, this information should be stored and managed only by one service, instead of using different data management services. Therefore, the Directory application described in this paper avoids inconsistent data caused by the formation of different application-specific databases.

The Directory offers an infoservice to the users of group communication applications. Users can retrieve all information about the existing roles and message types within his/her organization or of other organizations. In particular, due to "browsing" and "yellow pages" facilities provided by the Directory, it is possible to get information about roles, such as to know who can occupy a particular role and, by "white pages" facilities, to map role names to O/R addresses of users that play a role. For example, in a travel apply activity, a user who wants to send a request, can use the Directory in order to find the employee that s/he has to inquire and the communication information, that s/he needs for sending the request. The Directory offers the same kind of services for message types. That is, when users need to exchange messages, they can retrieve information about available message types and their structure using the Directory facilities. For example, if an employee has to send a short-notice to another employee, s/he can find the form of the message type "notice" inquiring the Directory. Furthermore, we can think of user agents which are able to interpret the Directory information in order to supply the user with an appropriate form on the screen.

5 Open Aspects, Future Work

It is clear that the Directory standard does not meet all requirements of GCSS and that extensions to the Directory are required. We will work out these extensions in this chapter. As we have already seen, there are many relations between entries. For example, the organizational role entries contain the names of entries representing the users that can take on the corresponding role. The communicational role entries have a link to other entries in the DIB, too. This link is realized by the *owner* attribute pointing to possible entries for that role. The Directory standard does not provide support for the retrieval of relations between entries. Thus, a first extension that the Directory needs affects its capability to interpret and to maintain such sorts of relations.

Due to these links existing between entries, the removal of entries where other entries point to leads to an inconsistent DIB. Therefore, in order to maintain the integrity of the DIB, the Directory will have to be extended, including mechanisms for support and management of these kinds of updates. At GMD we are working on solutions for these aspects. Our future plan is to develop an organizational knowledge base which is based on Directory concepts. Currently, we are extending our Directory service, which is used as an information service for users of our message handling service, towards a general information service about our organization for services and users as well.

Another important issue is the security of data in the DIB. We said that the different organizations involved in Group Communication activities are represented in the DIB and that their own information (organizational structure, roles, message types etc.) are available to the Directory users. It can be necessary for an organization to protect and to hide parts of this information from other organizations. Moreover, also inside an organization it is not always desirable to allow every employee the access to every information. Therefore, the Directory has to provide an authorization policy and it must support the specification, enforcement and maintenance of access rights for the purpose of granting and refusing access to information. Current implementations of the Directory service [9] already offer means for access control.

6 Conclusion

This paper has outlined how group communication support systems (GCSS) can apply the emerging X.500 Directory service. For this purpose the basic building blocks of GCSS were identified. Two of these components, roles and message objects, were identified as suitable for a representation in the Directory. For a better analysis of these components we distinguished between activity and conversation oriented models. This distinction results in the definition of organizational and communicational roles and message types, respectively.

The demand arose that a comprehensive user support for co-operative group work can be supplied only when activity and conversation oriented systems are combined and integrated. Such an integration would benefit from a single database for the administration of role and message type description. We listed the requirements for such a database and it could be shown that the Directory satisfies them all. Finally, we illustrated how roles and message types can be represented and in what manner the Directory service and the contained information can be applied.

We hope that this paper will stimulate researchers and developers of group communication support systems an impulse to consider an application of the emerging X.500 Directory service for future developments.

Acknowledgements

We would like to thank our colleagues A. Jerusalem and H. Santo for their helpful comments on this paper and U. Bernhard for the English proofreading.

References

[1] S.D. Benford. *Research into the design of distributed directory services.* PhD thesis, University of Nottingham, Nottingham, GB, October 1988.

[2] John Bowers and John Churcher. Local and Global Structuring of Computer Mediated Communication: Developing Linguistic Perspectives On CSCW in COSMOS. In *CSCW 88: Proceedings of the Conference on Computer Supported Cooperative Work*, Portland, Oregon, Sept. 1988. ACM SIGCHI & SIGOIS.

[3] T. Danielsen. AAM - The AMIGO Activity Model. In U. Pankoke-Babatz, editor, *Computer Based Group Communication, the AMIGO Activity Model*, pages 67–125. Ellis Horwood, 1989.

[4] F. de Cindio, G. de Michelis, C. Simone, R. Vasallo, and A. M. Zanaboni. CHAOS as Coordination Technology. In *CSCW'86 Conference on Computer-Supported Cooperative Work*, Austin, Texas, Dec. 1986.

[5] U. Pankoke-Babatz (Ed.), T. Danielsen, A. Patel, P.A. Pays, W. Prinz, and R. Speth. *Computer Based Group Communication, the AMIGO Activity Model.* Ellis Horwood, 1989.

[6] R.E. Young et al. Interim Report on the COSMOS Project, COSMOS Report. Technical Report 45.5 EXT, COSMOS Project, COSMOS Coordinator's Office, Queen Mary College, London., January 1988.

[7] Heikki Heammaeinen, Reijo Sulonen, and Christian Berard. Pages: Intelligent forms, intelligent mail and distribution. In *IFIP WG 8.4 Working Conference Pisa*, Oct. 1986.

[8] ISO and CCITT. Information Processing Systems - Open Systems Interconnection - The Directory, 1988. ISO 9594- 1-8, CCITT X.500-X.521.

[9] S. E. Kille. The QUIPU Directory Service. In P. Schicker E. Stefferud, O.J. Jacobsen, editor, *Message Handling Systems and Distributed Applications, IFIP 6.5, International Working Conference, Costa Mesa, Oct. 1988*. North-Holland, 1989.

[10] Thomas Kreifelts and Gerd Woetzel. Distribution and error handling in an office procedure system. In G. Bracchi and D. Tsichritzis, editors, *Office Systems: Methods and Tools, Proc. IFIP WG 8.4 Work. Conf. on Methods and Tools for Office Systems*, pages 197–208, 1987.

[11] T. W. Malone, K. R. Grant, K-Y Lai, R. Rao, and D. Rosenblitt. Semi-Structured Messages are Surprisingly Useful for Computer-Supported Coordination. In *CSCW'86 Conference on Computer-Supported Cooperative Work*, Austin, Texas, Dec. 1986.

[12] U. Pankoke-Babatz. Requirements for Group Communication Support in Electronic Communication. In R. Speth, editor, *Message Handling Systems, IFIP 6.5, International Working Conference, Munich.* North-Holland, April 1987.

[13] W. Prinz. Survey of Group Communication Models and Systems. In U. Pankoke-Babatz, editor, *Computer Based Group Communication, the AMIGO Activity Model*, pages 127–180. Ellis Horwood, 1989.

[14] W. Prinz. X.500 Directories in the Office. internal report available from the author, to be published, 1989.

[15] W. Prinz and R. Speth. Group Communication and Related Aspects in Office Automation. In R. Speth, editor, *Message Handling Systems, IFIP 6.5, International Working Conference, Munich.* North-Holland, April 1987.

[16] W. Prinz and M. Woitass. The Date Planning System - Example of a Cooperative Group Activity. In P. Schicker E. Stefferud, O.J. Jacobsen, editor, *Message Handling Systems and Distributed Applications, IFIP 6.5, International Working Conference, Costa Mesa, Oct. 1988*. North-Holland, 1989.

[17] D. Rotondi, A. Pepe, P. Penelli, and M. Waldow. The Directory in heterogeneous computer networks: design and application aspects. In *Proceedings of the VIII - Seminar on Packet switching networks - Leningrad, USSR*, June 1989.

[18] Action Technology. *The Coordinator, Workbook & Tutorial Guide*, June 1987.

[19] D. Tsichritzis and S. F. Gibbs. Ettiquette Specification in Message Systems. In D. Tsichritzis, editor, *Office Automation*. Springer, 1985.