Human Beings in Epos
A contribution to the Tutoring System workpackage

M.F.Verdejo, M.T.Abad

LSI-90-26

Abstract

The main goal of ECOLE, a Collaborative Learning Environment, is to implement a prototype of an environment which makes users able to identify and effectively use EPOS resources satisfying their educational needs. Within this framework, this document addresses the design of a specific channel of interaction among a particular kind of resources in ECOLE: Human Beings.

Topics discussed include the abstract characterization of an agent i.e. the definition of what he knows and the kind of operations he can performs as well as the structure of the communication process , in particular when and why to require an agent for an interaction and how to define the communication.

Particular attention is devoted to the definition of an electronic mail service to support Human-Human interaction. A structure and a typologie of messages is given and the mailing functionalites are specified.

Keywords: Human-Human communication, Mailing Services.

List of contents **Page**

Human Beings in Epos: a contribution to the Tutoring System Workpackage

M.F.Verdejo , M.T.Abad

1. Introduction

The Epos Collaborative Learning Environment (ECOLE) aims to provide users of Epos with an on-line advisor to determine h(is)er needs. WP 2.7.2 will implement this module following an approach where the user through mixed-inititative dialogues together with a set of functions could explore, identify and effectively use an educational network.

The educational network includes diverse types of learning materials, as well as Human Beings resources, and therefore a key aspect is to identify why and when does a user need to interact with other persons in the context of a collaborative learning environment.

Within this framework, WP 2.7.2 is involved in designing and implementing a specific channel of interaction among Humans and in particular supply ECOLE with suitable mailing functionalities. Two complementary approaches have to be taken into account: one to one and group communication. The workload proposed to Telefónica-Technical University of Catalunya is the specification of ECOLE mailing functionalities, mainly focused on one to one communication. This task, as established in [Cas-90b] means to identify some phases in which a user needs helps by Humans and for each of these phases to isolate a set of mailing functionalities which allow the user to search for Humans (a list of attributes describing Humans first of all) and to interact with them designing a suitable form for the mail.

This report presents the results of our study. After this Introduction, Section two describes some general features of Epos, relevant for our purposes. Section three establishes our methodological approach to carry out the task, while Section four focuses on the characterization of Human-Human communication. Section five is devoted to the communication system, first to define the abstract description of agents and then to describe the main components of the electronic mail service. Finally Section six specifies the mailing functionalities provided by means of an user interface. Two annexes are included concerning the application domains that have been used as a first testbed for different aspects of our study.

2. Modelling the problem domain

The entities we want to model, as mentioned before, are educative products, human resources, and some of the components of the Epos system such as the guidance, monitoring or tutoring modules. From now on we want to make the distinction between **agents** and **products** depending on either the active or pasive role they can play. An agent can decide to act, for instance to communicate with other agents, to inspect a knowledge base, to use a product,....while a product is activated only when required by an agent. However an agent can be a human or an artifact i.e. a piece of software. On a first step we will consider that products, as well as curricula and other related material such as teaching plans, are all objects described in a knowledge base called **Curricula Information System** and the agents will be defined in a similar way in another KB, called **Agent Information System**. Our purpose now is to focus on agents . In this section we will outline the different type of agents in order to characterize their main role within Epos.

2.1 Agents in Epos

What kinds of agents do we have in Epos?. We can consider two main classes, **Human Beings** and Artifacts. Furthermore for each class we have the following subclasses:

HB : **Student, Administrator, Author, Manager, Tutor, Producer**

Artifacts: **Guidance System, Tutoring System, Monitoring System**

2.2 General functionalities in Epos

Taking into account diverse Epos deliverables, as well as our own study, we can establish as Epos main functionalities, those set out below:

To give information about available instructional material,teaching plans and curricula

To give administrative information about the use of Epos, conditions and requirements.

To give information about other Epos users.

To provide facilities for designing instructional material, teaching plans and curricula

To provide facilities for updating instructional material, teaching plans and curricula

To activate and control the execution of a product by a user, following a teaching plan

To include a user, and then to monitor and control its activity within Epos

2.3 Main functions for each agent

Although any agent can perform a set of different activities, all of them are related each to their major goal, so it is important to characterize for each agent its main role in Epos.

Student: uses Epos to improve his cognitive abilities and skills

Manager: defines curricula, authorises regulated students acces to Epos

Tutor: provides help to the student along the learning process

Author: creates and designs teaching material

Producer: shapes and introduces instructional products into the market/epos
Administrator: registers and authorizes some classes of users in Epos
G.S: supervises (execute and control) a teaching plan for a user
T.S: provides help and information to a human user within Epos
M.S.: provides and elaborates information about the use of Epos

As has been pointed out before we will use a Knowledge Base Formalism in order to represent what an agent is and what he can do. In this way we will create an Agent Knowledge Base formed by a structured set of objects, each one representing either an abstract or concrete entity. Thus a KB will contain the definition of relevant concepts in the domain, as well as the representation of concrete entities, instances of these concepts.

It is worth noting that any KB formalism provides two components:

. The Knowledge Base: Data Structure where the objects modelling the domain will be represented using a particular language.

. The set of mechanisms operating on the Knowledge Base, i.e the classical operations on a data structure such as creating new objects, including them in the KB accumulatively, inspecting the content,.. however all these operations handle the knowledge represented either in an explicit way or derived by **inference processes.**

We assume in our study two separate KBs, one for curricula and related entities such as teaching plans and the other one for agents. This decision does not have design or implementation consequences. At the moment, it is only a criterion to focus on our problem. Nevertheless it is obvious that objects in a KB will refer to objects in the other KB. This is one of the main points that we have expanded on and it is discussed in further sections.

3. Objectives and methodological approach

Within the Epos framework discussed above, our main objective is twofold:

1. Why and when an agent needs to communicate with a Human Being within Epos
2. How this interaction can be carried out

In more general terms, we have dealt with the following topics:

a) How to define an agent in epos, i.e. what he knows, and what kind of operations he can perform

b) The structure of the communication process, in particular how to require an agent for an interaction and how to define who else he can communicate with

c) How to establish a typology of communications, either related to purpose, content, or both

d) What requirements are needed for a friendly Human-Being communication system within Epos.

Our approach for the whole task will consist of dividing the process into a series of steps, mainly Specification, Prototyping and Evaluation.

The specification phase was carried out from March to July 1990 and it has been divided into the following activities:

1.Problem understanding and characterization
2.Proposing a model
3.Validating the model

To carry out the first step, problem understanding and characterization, we have analysed in parallel the following issues:

a) Agent characterization : communications between them, for what (with examples) and when (in terms of relations to other interactions).
b) A characterization of the communication configuration for each type of functionality, (type of agents involved and links).
c) A characterization of the functionalities for each type of agent.
d) A characterization of communicative acts: preconditions, effects and termination conditions.

The process takes the form of a series of successive refinements.

The second step for the specification phase consisted of the analysis of the above material in order to explore a model for defining human beings and the communication process.

The third step aimed at assessing the model for different sub-domains, one of them related to Telefonica, the other one to an existing educative network.

The problem has now been specified as it is further detailed in next sections. The second planned phase entails selecting features to be demonstrated and building a prototype i.e. a subset of the total system. The aim is to enable appropriate testing and eventually use the results to revise the model.

Finally a third phase will evaluate the prototype, in order to study whether or not the implementation is adequate to meet the requirements, and determine later development iterations to expand the depth and breadth of the coverage.

4. Characterization

4.1. Who can communicate within Epos, and for what purpose?

First of all we will examine any combination of pairs of agents, to establish whether it is reasonable or not within Epos to consider them as potential interlocutors. Then in the case of a positive answer we will discuss different communicative acts, illustrated with relevant examples, in order to characterise types of actions and for each type give its goal and preconditions. This is a general scenario, for particular cases modifications can appear, but they can be defined following the same patterns.

1. Interlocutors: Student, Student

1.a Purpose: *Request / give* personal *information* about
localization, availability, motivations..
activities carried out within epos
available personal resources

Preconditions: to know how to reach one/another

Examples: I am following the curriculum X, do you have some support material for Y?
What is the memory size of your computer?
I am studying X, what tutor do you recommend ?

1.b Purpose: *Propose / accept / reject / change* an *agreement* about
carrying out an activity
share a resource

Preconditions: to know how to reach one/another

Examples: can we do the exercices of the unit X and discuss their solutions ?
can we meet next week to discuss X?
may I use your printer to have a copy of my file Y?

2. Interlocutors: Student, Author. Has no purpose

3. Interlocutors: Student, Manager

Purpose: *Request / give authorization* to became an Epos's user
Preconditions: to know how to reach one/another, S is a regulated student
Examples:I want to obtain the qualification X
I want to do the exam X without following the part Y of the Z teaching plan

4. Interlocutors: Student, Producer. Has no purpose

5. Interlocutors: Student, Tutor

5.a.1Purpose: *request / give information* about
instructional resources

Preconditions: to know how to reach one/another
Examples: what I need to know to follow the course X?
how much time is needed to study subject Y?
what are the books recommended for this subject?
where can I find more exercises about X?
Is there a product to practice X using the Z programming language?

5.a.2 Purpose: *request/give advice* concerning a teaching plan
Preconditions: to know how to reach one/another, T knows teaching plan

5.b.1 Purpose: *propose/accept-reject* an *agreement* about an instructional activity
Preconditions: to know how to reach one/another, T is assigned to S for consultation

6. Interlocutors: Student, Administrator

6.1 Purpose: *request/give authorisation* for entering epos
Preconditions: Student is a free user

7.Interlocutors: Student, Guidance System

7.1 Purpose: *propose/accept - reject* an *agreement* about a teaching plan
Preconditions: S is a regulated student or S has already consulted with TS

7.2 Purpose: to *order/accept* the execution of a teaching plan
Preconditions: a teaching plan has already been selected

8. Interlocutors:Student, Tutoring System

8.1 Purpose: *request/give* information about epos resources
Preconditions: none

8.2 Purpose: advice about a teaching plan
Preconditions: none

9.Interlocutors:Author, Author. Has no purpose

10. Interlocutors: Author, Manager.

10.1 Purpose: request/give information about epos resources
Preconditions: to know how to reach one/another
Examples: The instructional material X is adequate for curriculum Y
What products similar to X are used in the curriculum Y
For how long are students have been working with the material X

10.2 Purpose: *Propose / accept / reject / change* an *agreement* about including tutors
Preconditions: to know how to reach one/another
Examples: this material should be used by a tutor knowing X and with experience of Y

11. Interlocutors:Author, Producer

11.1 Purpose: *Propose / accept / reject / change* an *agreement* for using ideas or products
Preconditions: to know how to reach one/another

Examples: conditions to produce of instructional material
conditions to use part of an instructional material in a product

11.2 Purpose: *Request / give authorization* to use an idea or material
Preconditions: to know how to reach one/another

11.3 Purpose: *request/give information* about instructional material
Preconditions: to know how to reach one/another
Examples: what are the areas covered by your material
what kind of techniques are proposed
what kind of support is available for drill and practice

12. Interlocutors: Author, Tutor

12.1 Purpose: *request/give information* about epos resources
Preconditions: to know how to reach one/another
Examples: some errors are reported related to your material
students find the proposed exercises difficult
questions most frequently asked are about...

12.2 Purpose: *request/give* advice about
teaching plans
epos_resources
Preconditions: to know how to reach one/another
Examples: this material should be used in such and such a way
explanations should be complemented with such and such a practice

13. Interlocutors: Author, Guidance System. Has no purpose
However can be for testing, but in this case the author is like a student

14. Interlocutors: Author, Tutoring System

Purpose: *request/give* information about epos resources
Preconditions: none
Examples: who are the authors for recomended products in this curriculum
how many curricula use product X

15. Interlocutors: Author, Administrator

It is unlikely that cases 15.1 and 15.2 will happen within Epos

15.1 Purpose: *Propose / accept / reject / change* an *agreement* about being included in epos
Preconditions: to know how to reach one/another

15.2 Purpose: *request/give authorisation* for entering epos
Preconditions: Preconditions: to know how to reach one/another

15.3 Purpose: *request/give information* about epos resources
Preconditions: to know how to reach one/another

16. Interlocutors: Manager, Manager

16.1 Purpose: *request/give information* about epos resources
Preconditions: to know how to reach one/another
Examples: how many teaching plans do you have for curriculum X
what kind of prerequisites do you ask students for in

curriculum X

16.2Purpose: *Propose / accept / reject / change* an *agreement* about
use of a product
sharing a resource
including an agent in epos

Preconditions: to know how to reach one/another

Examples: can we share tutor X
can we share the production of X for curriculum Y

17. Interlocutors: Manager, Producer

17.1 Purpose: *request/give information* about epos resources

Preconditions: to know how to reach one/another

Examples: what are the products considered for curriculum Y

17.2 Purpose: *request/give* advice about epos resources

Preconditions: to know how to reach one/another

Examples:other material to complement this product is X

17.3 Purpose: *Propose / accept / reject / change* an *agreement* about
including an instructional object in epos

Preconditions: to know how to reach one/another

Examples: conditions for a new release of product X

18. Interlocutors: Manager, Tutor

18.1 Purpose: *request/give* information about epos resources

Preconditions: to know how to reach one/another

Examples: how many students make questions
do students have the adequate level
do students have any problems

18.2 Purpose: *Propose / accept / reject / change* an *agreement* about
including an agent in epos

Preconditions: to know how to reach one/another

Examples: I would like to be tutor on X

19. Interlocutors: Manager, Guidance System. Has no purpose

20. Interlocutors: Manager, Tutoring System

Purpose: *request/give information* about epos resources

Preconditions: none

Examples:how to proceed to include a new curriculum

21. Interlocutors: Manager, Administrator

21.1 Purpose: *comand* to include an object in epos

Preconditions: to know how to reach one/another

21.2 Purpose: *request/give information* about epos resources

Preconditions: to know how to reach one/another

22. Interlocutors: Producer, Producer

Purpose: *request/give information* about epos resources

Preconditions: to know how to reach one/another

Examples: how to proceed to include a product in epos
what are your conditions to use the product X

23. Interlocutors: Producer, Tutor

23.1 Purpose: *request/give information* about epos resources use
Preconditions: to know how to reach one/another
Examples: It is quite complicated to use product X

23.2 Purpose: *request/give information* about epos resources
Preconditions: to know how to reach one/another
Examples: there is an error reported on product X

24. Interlocutors: Producer, Guidance System. Has no purpose

25. Interlocutors: Producer, Tutoring System

Purpose: *request/give information* about epos resources
Preconditions: none
Examples: which managers could be interested in product X
which tutors are available on subject X

26. Interlocutors: Producer, Administrator

26.1Purpose: *Propose / accept / reject / change* an *agreement* about being included in epos
Preconditions: to know how to reach one/another

26.2 Purpose: *request/give authorization* for entering epos
Preconditions: to know how to reach one/another, if not it is unlikely.

27. Interlocutors: Tutor, Tutor

27.1 Purpose: *request/give* information about epos resources
Preconditions: to know how to reach one/another
Examples: which curricula you are included in
what kind of questions you receive on subject X

27.2 Purpose: *request/give* advice
Preconditions: to know how to reach one/another
Examples: what approach do you use to present X

27.3 Purpose: *Propose / accept / reject / change* an *agreement* about
appointment
substitution
making exams
Preconditions: to know how to reach one/another
Examples: can you substitute me next Monday

28. Interlocutors: Tutor, Guidance System: Has no purpose

29. Interlocutors: Tutor, Tutoring System

Purpose: *request/give* information about epos resources
Preconditions: none
Examples: is there any complaint about me

30. Interlocutors: Tutor, Administrator

30.1Purpose: *Propose / accept / reject / change* an *agreement* about being included in epos

Preconditions: to know how to reach one/another

30.2 Purpose: *request/give authorization* for entering epos

Preconditions: to know how to reach one/another, if not it is unlikely

An important point to consider is **when** a communication can be performed, we see different possibilities:

1. before or after any other activity
2. while perfoming another activity, and in this case:
   2.1 the current activity will be interrupted, waiting for an answer for continuation
   2.2 the current activity will continue
   2.3 the current activity ends

In any kind of situation with human-human interlocutors, a communication operation will be performed before or after any other activity, with only one exception: if help is needed to continue with the current activity. An example of this case is a student asking the tutor for some explanation of a step of a teaching plan, while being supervised by the guidance system. This scenario should correspond either to 2.1 if the communication process can be really on-line , to 2.2 if the student thinks he can continue, or to 2.3 if he must have some feedback. Another typical example concerns an user who needs to know his interlocutor, for a specific purpose, in this case two solutions can be considered: either he should first interact with the tutoring system and establish the communication afterwards (1) or the tutoring system will directly establish this communication for the user and then 2.3 will be the case.

4.2. Classes of comunicative actions

In the above section we have grouped interactions according to the type of interlocutors, the analysis of this material shows that there are three main classes of comunicative actions for a situation with two interlocutors: consultancy, negotiation and order. The first one concerns an agent communicating to another one, using a request, to obtain help: general information, instruction or advice. The second class represents both, communications where an agent will negotiate an agreement, involving propositions, rejections or acceptance actions and communications to request an authorization. Finally the third one is related to the execution of a command, and three subclasses can be established: supervising the student while executing a teaching plan, modifying Epos KBs and notifying information/command.. from one agent to another. The following table describes each class indicating the way the speech act is formulated, the objective and the theme of the communication:

operations for consultancy

speech act	*objective*	*theme*
request / answer	**information**	personal .localization .activity within epos . resources epos_resources .humans .products .curricula .teaching plans
	instruction **advice**	related to a learning unit epos_resources .humans .products .curricula .teaching plans activity

operations to negotiate

propose/accept/reject	**agreement**	activity . appointment . use .. product .. epos .. personal resources .. teaching plan include/modify an entity in epos . curriculum . teaching plan . product . agent make an examination make a substitution
request	**authorization**	activity . use .. product .. epos .. personal resources .. teaching plan include/modify an entity in epos

speech act	*objective*	*theme*
		. curriculum . teaching plan . product . agent

Command operations

proceed	**executing**	a teaching plan
	executing	Epos KBs modification
	executing	notification

5. Communication System

5.1. Introduction

Before going into a detailed study of the requirements for a tool supporting human-human communication in Epos, it would be helpful to outline the facilities offered by the present systems.

Currently available communication systems (for instance EAN, EARN, Mensatex..) provide the user with different services that can be classified into the following four main types:

a) Access to general information included in different classes of information structures

1- Directories: a set of records, describing for each user name, activity, areas of interest,company or institution, country.. A directory can be consulted by means of a special purpose search language. Queries are expressed as combinations of descriptors linked by logic or relational operators. Directories can be updated either by the user or the system administrator.
2- Files and data bases containing references and descriptions about types of products, resources or data.
3- Thematic electronic forums and electronic newsletters

b) Transfer of files

c) Mailing service with two main modes
1-interactive (on-line dialogues)
2- electronic mail

d) Other possibilities such as accessing remote resources

Focusing on the mailing system, we can distinguish two main classes of functionalities : **message transfer** and **user service.** The first one is in charge of receiving and sending messages along an adequate path while the second one offers to the user several facilities to create, store and manipulate messages.

Usual kind of basic operations included in a user service of a mailing system are as follows :

1. Storing messages

A mailing system provides the user with a structure (folder) for keeping received/sent mail, as well as an automatic way of storing messages in them. A folder is a data structure defined as a sequence of messages, usually ordered by date. The set of operations consists of creating and deleting a folder, moreover other commands for consultation are also available, such as printing names of existing folders, printing the directory of a folder, ... Some systems offer another structure for message organization: a conversation, set of messages linked by the system as the user creates them with a reply operation.

2. Composing and editing messages

A message is formed by a set of components among them receiver, subject and content. Other parameters concerning how the message will be handled by the system are usually predefined for each user. The user can define alias of mail addresses to simplify the filling of the receiver field when creating a message, and sometimes this value can be specified in terms of a search operation over a directory. The subject and content of a message are texts created and manipulated by a text editor.

3. Consulting parameters of a message

Other information can also be obtained such as the status of a message, whether it has been received or not, if an answer is required, etc.

We now wish to point out the relevant features of a communication system for human beings in epos. We will start by defining in 5.2 the kind of knowledge needed to characterize humans as epos agents, in particular for modelling their communication functionalities, then we will focus on electronic mail, 5.3 further specifies the structure of a message and the operations that should be available in the mailing system. For these two points we have take into account ideas already discussed in epos deliverables, as well as the results of a detailed work that we have carried out for two case studies: Telefónica Formación and the Educative Network of the Generalitat de Cataluña.

Telefónica Formación is an interesting example to be outlined because its organization presents a quite complex and complete scenario for regulated epos users.

The educational network of the Generalitat de Cataluña is a working communication system, including different educational services, distributed over 4000 nodes, most of them being primary and secondary schools.

Annex one and two describe respectively these case-studies and their formalization with the model proposed.

Next, in 5.4 we further elaborate on message content characterization. For each class of messages we define types according to semantic criteria, and for some of these types we also give their syntactic structure in a formal way. Types can be exploited in different ways, such as designing a structured text editor to compose messages, for consistency checking purposes, etc., and it is a key issue for further extension of epos services, such as an expert front-end, understanding messages in natural langage and providing answers automatically by direct access to knowledge bases or information systems.

This section ends with 5.5, where a synthesis of requirements for the mailing system is presented.

5.2. An abstract description of agents in EPOS

5.2.1. Notation

The notation used to define objects (representing concepts or indivivuals) and their properties is a frame-like one. An object has an internal structure and can be related to other objects. Relations permit the creation of taxonomies and provide a way of defining an object in an incrementaly by structural inheritance. The structure of an object consists of a set of pairs called slot and filler (or attribute and value specification respectively).

5.2.1.1. Syntax, B.N.F. notation.

<object definition> :: = < object name> <Relations> *structure* <Structure>

<Relations >:: = (<relation name> <object *with* restriction> /
<relation name > <disjunction of items> /
<relation name> <enumeration of tuples>)

<Structure>:: = (<attribute name> : <range>)

The range in a conceptual/abstract object is used for two purposes:

1. to indicate what kind of values this attribute can take when instances of this object are created.
2. to define new objects as subclasses of this one, by giving a constraint over the range

<item>::= < object name> / <value>

<range>::= <item> / <disjunction of items> / <set of tuples>

A range can be defined either by reference to an object, by giving a value, as a set of exclusive disjunct items or an explicit set of tuples

<set of tuple> ::= "(" tuple ")"

<tuple>::= "{" item, (item) "}"

<disjunction of items>::= " [" item, (item) "]"

5.2.1.2. Relations with a predefined behavior

We consider three predefined relations, the table below shows their name, inverse and main properties respectively

Name	inverse	properties
is_composed_of	is_part_of	transitivity
subclass	subclasses_are	transitivity,structural inheritance
instances_are	instance	composition with subclass, structural inheritance

5.2.2. Definitions

In order to define an epos agent we need to introduce some other general concepts that we will describe partially. Numbers in the objects below are only given to facilitate reading and reference.

1.Person

structure

name:
localization:
language

A person has a name, a localization and a language, without further specifying any range for these attributes at the moment, because they are not relevant for our purposes.

2. Artifact
subclasses_are
active_artifact
educative_product

An artifact has two subclasses, active_artifact and educative_product respectively.

25. active_artifact
subclass of artifact

26. Educative_product
subclass of artifact
structure
associated_curriculum: (abstract curriculum)

A particular educative product can appear (be used) in a diverse concrete curriculum therefore its definition includes a slot associated_curriculum whose

values have to be instances of an abstract curriculum.

3. Employee
   subclass person
   structure
      employer:
      category:
      charge:
      function:
      workplace:
      professional_cv:

An employee is a subclass of person (inherits name, localization and language) having an employer, a category, a charge, a function, a workplace and a professional curriculum. No further information for these attributes is given for the moment.

4. **Epos_agent**
   subclass [person , active_artifact]
   structure
      affiliation type: [private, company]
      user_type :[student, manager, author, tutor, producer tutoring_system,guidance_system, monitoring_system]
      com_function : (communication_functionality)

An epos agent is a person or an active artifact having an affiliation type (either private or belonging to a company), an user type (one of student, manager, author, tutor, producer, tutoring system....) and a set of communication functionalities.

A communication functionality is defined also as an abstract object, related to functionality, and its structure is explicitly defined by four attributes. The first one, communication_type characterizes the classes of communicative operations that can be performed, the second indicates the way used to communicate, here the electronic mail, the third, user profile, describes the agent from the viewpoint of a particular electronic-mail user, and the last one, configuration, specifies the type of agents participating in the communication process.

40. functionality
   subclass is communication_functionality

41.communication_functionality
   subclass functionality
   structure
   communication_type: message.characterization
   way: electronic_mail
   user_ Profile: electronic_mail_profile
   configuration:
      Who_access: epos-agent
      To_whom: epos-agent

The value of the communication_type slot, for an instance of communication_facility is given by a set of keywords characterizing the purpose and the content of a message, we will discuss further this point on section 5.3.

43. electronic_mail_profile
    subclass communication_profile
    structure
      identification
      default_parameters
      agenda_definition: agenda
      message_storage: data-base
      alias: table of electronic-addresses

The profile indicates the electronic-mail identification for each user, the set of parameters for which a default value is given, the structure of the agenda for noting tasks related to mailing, the data structure for storing messages and a set of alias of electronic addresses. Details about these properties are discussed in section 5.3.

The configuration slot has two components Who_acces, to indicate the agents able to initiate a communication to the user, indicated in the user_profile slot, without having been previously requested, and To_whom to establish possible receivers of this kind of communication from this user.

An human being in epos can be defined as a subclass of person and a subclass of epos_agent with a range restriction:

39. human_epos_agent
    subclass epos_agent *with* user_type :[student,author, manager, producer, tutor]
    subclass person

The objects previously defined constitute the general level of the knowledge base; once they have been introduced, a more specific level can be added to the KB by defining different concepts for each user type.

New objects will be described as usual, by restricting some inherited properties (for instance functionality) of a more abstract object, and/or by adding new slots.

A Manager is a subclass of human_epos_agent,with two main communication functionalities: to consult and to negotiate. Both can be further divided taking into account the theme. For each type of communication the who_acces indicates the type of agents having acces to the manager (without a previous request) for this operation and the to_whom slot describes the type of agents that the manager can have acess to for this operation.

6. Manager
subclass human_epos_agent with user_type = manager
with communication: { consult, information, epos_resources}
who_access: [tutor, author , producer]
to_whom: epos_agent

with communication: {consult, information, personal }
who_access: manager
to_whom: manager

with communication: {negotiation, autorization,activity}
who_access: manager
to_whom: epos_agent

with communication: {negotiation, agreement, activity}
who_access: [Tutor, Author, P, Ad]
to_whom: epos_agent

A tutor is described as a subclass of human_epos_agent, adding some other relevant information: his status, pedagogic evaluation, subject_matter specialization, curriculum where he appears, and its availability.

7. Tutor
subclass human_epos _agent with user_type= tutor
structure
status: [permanent, contractual]
pedagogic.eval
subject_matter_spec
associated_curriculum: abstract_curriculum
consult_timetable

31. Student
subclass human_epos_agent with user_type= student
structure
associated_curriculum: abstract_curriculum

In the same fashion Author and Producer can be defined. From these two levels diverse classifications could be considered to extend and refine the agents to be modelled, for instance further subclassifying each type of users by affiliation, geographic areas, thematic subjects, and so on. By proceeding in this way several conceptual taxonomies can be generated. Instances can be attached to the more concrete objects situated at the bottom, their complete structure being defined by the properties inherited through the taxonomic relationships.

References to the object **abstract curriculum** can also be established at user type level, for instance in the above definition for tutor, the range for associated curriculum attribute is an abstract curriculum. In a similar way, for students,

references to the curriculum they are following as well a their learning profile can be included. These decisions should not be unique for epos, if we are thinking of diverse sub-epos scenario, knowledge granularity and organization should be customized to reflect specific needs and should be decided taking into account a particular study of the problem.

In the current version of the abstract curriculum [HerVer-88] there exists a few attributes concerning authors and tutors (see comments in annex 1). Products are identified by attributes as part of the curriculum. Teaching plans are not described.

We propose to clearly separate and encapsulate product as an object and therefore to eliminate its attributes from the abstract curriculum object, because a product can be used in different curricula but being the same its description should be unique.

In summary the complete information/knowledge system should distinguish the following relevant (both abstract and concrete) objects:

epos_agent (human or artifact)
curriculum
teaching plan
educative_product

These objects have a structured description, and maintain links between them, for instance, as indicated above a tutor with the concrete curriculum he is involved with. In annex two a detailed modelling for a case example is given.

5.3. The electronic mail service

An electronic mail has to be one of the services offered by the communication system to interchange messages between epos human agents, between human agents and epos artifacts, for instance the tutoring system, and perhaps between some type of agents and the information/knowledge bases. This last case depends on the final design of epos architecture.

We will propose here a structured definition of message, based upon the standard one, but with some other properties useful for epos purposes. A typology of messages is established and a first approximation to formalize the content for each type of message is given. This approach is needed for two reasons: to design the structure of the menus for the user interface service, and as a first step for future design and implementation of automatic consultation for some kinds of messages currently handled by human agents.

5.3.1. A message: its structured definition

A message is defined by a set of slots, specifying properties about six main features, namely identification, requirements for an answer, checking parameters, transfer parameters, storage parameters, and characterization of its content.

44. Message
identification
sender: [human_epos_agent,TS , GS , Mailing system]
receiver: (epos_agent)
communication_source: [Message.sender, epos_agent]
source_visibility:[yes, not]
scope : [public, private, (epos_agent)]

answer_requirements
confirm_reception : [yes, not]
answer_claim: [yes, not]
answer_due_deadline:

checking_parameters
status: [pending, send, received, answer_needed, accept, reject]
nº_of_lines:
date_of_issue:
date_of_arrival:

transfer_parameters
storage parameters

content characterization
type: [consult, negotiate, order]
speech_act: [request/inform, proposal/agreement/rejection,
command/accept]
objective: [information, agreement, teaching, advice, notification,
operation]
theme:[personal,epos_resources,epos_cond,external_activity,
com_system_inf,epos_update]
content: [text, formal expression]

Comments to message's descriptors :

The identification block. It is used for two differente purposes, first to formulate the message in terms of logic address-electronic address, and second to inspect or search the message by some of the descriptors

• Receiver: an expression allowing to identify either a user or a group of users.

• Communication_source: identification of the agent that has sent this message. There are two possibilities: the source is the same as the sender or another epos_agent working as an intermediary. In the second case the source_visibility's descriptor denotes who is the sender and will know the receiver (the initiator or the intermediary).

• Scope: If it is public it means that someone different to the sender or receiver can know the content of the message.

Answer_requirements block. Consists of a set of descriptors indicating how the sender will be informed that the message has been received.

• confirm_reception: the electronic mail service notifies the sender that the message has been received by the receiver. This notification does not imply the reading of the message only its reception.

• answer_claim: if the value of this descriptor is affirmative and the receiver does not reply in the answer_due_deadline, the mail service will send the same message automatically to the receiver a finite number of times. If no answer is received, the service will notify the failure of the communication to the sender.

Checking_parameters block, in order for the system to check the right reception of the message.

• status: It shows the status of the message. In some cases, the service will generate a special kind of message related to the specific status of the message or the status of the service. For example, the service could inform the sender if:

- the electronic mail is overloaded (this can be controled when the sender tries to send a message)
- the receiver is unknown (this could be checked when the sender defines the receiver or after the message has been sent)
- the link between the sender and the receiver is broken.
- the receiver rejects the message because the sender is not one of the right interlocutors (or because the receiver doesn't want speak to the sender !)
- the message has arrived at its destination (only if confirm_reception = yes)
- the same message was sent X times (only when answer_required)

In the last two cases it will be necessary to limit the kind of messages than can use confirm_reception and answer_claim because their use could produce net overload.

Communication failure could have a "physical" cause like the receiver does not exist or the link is broken, or a "personal" cause like the receiver rejects the message or he/she does not reply when the message received needs an answer (confirm_reception = yes).

The content characterization block is used to classify messages. This classification will be useful both, to help in producing the message and afterwards for storage and inspection.

These descriptors, their relations and ranges are introduced in the next section which focuses on communication between human agents.

Communications between human_epos_agents

Following the characterization given in section 4.2, the typology for messages between humans agents (tutor,student, manager, author, producer, administrator) in terms of type, speech act, objective and theme is restricted to the ranges defined below.

type: [consult, negotiation, order]
speech act: [request/inform, proposal/agreement/rejection,
command/accept]
objective:[information, arrangement, teaching, advice, notification]
theme :[personal, epos_resources, activity]
epos_resources = [agent, educative_product, curriculum, teaching_plan]
activity: [epos_cond, registration, external_activity]
epos_cond = [epos_resource_use, epos_resource_ update]
external_activity:[appointment,event,.....]

Their combinations are shown in the following table

type	speech_act	objective	theme	reference to 4.1examples
CONSULT	request/inform	information	personal	(1,a) (16,a)
		or	epos_resources	(5,a,1) (10,1) (12,1)
		advice		
			activity	
		teaching	learning_unit	
NEGOTIATION	propose/agree/	arrangement	activity	(1,b) (10,2)
	reject			
	request/agree			(3,a) (11,1)
ORDER	command	teaching	epos_resource_use	
			external_activity	

In this table, only electronic human-human interaction is considered and therefore any kind of communications where one of the participants is an artifact does not appear. Next, we detail the above combinations for each user type.

Message characterization for each type of human_epos_agent

The range for the slot communication_type (belonging to object 41) was specified as message.content.characterization and message.content.characterization has been described as a set of slots of message (44) in terms of type, speech act, objective, and theme. Our objective now is to define the possible combinations of type, objective and theme in order to build a complete catalogue of value sets for each human_agent_type. The receiver types are also indicated in brackets.

Student : S with S,T,M,Ad
- consult, information, personal (S)
- consult, information, epos_resources (T, S)
- consult, teaching, learning_unit (T)
- consult, advice, epos_resources (T, S)
- negotiation, arrangement, epos_resource_use (M,Ad)
- negotiation, arrangement, registration (Ad)
- negotiation,arrangement, external_activity (S)

Tutor : T con T, A, P, S, M, Ad
- consult, information, personal (T)
- consult, information, epos_resources (T, A, S, P)
- consult, advice, epos_resources (A, T, M)
- negotiation, arrangement, epos_cond(M, Ad)
- negotiation, arrangement, personal (T)
- negotiation, arrangement, external_ activity (S, A)
- Order, teaching, epos_resource_use (S)

Author : A con T, P, M, Ad
- consult, information, epos_resources (T, P)
- negotiation, arrangement, external_activity (P)
- negotiation, arrangement,epos_resource_use (Ad, M, P)

Producer : P con A, T, M, Ad
(* alike author *)

Manager : M con M, Ad, T, P, A, S
- consult, information, epos_resources (T, A, P)
- consult, information, personal (M)
- negotiation,arrangement, activity (Ad, M)
- negotiation, arrangement, epos_resource_use (S, T, A, P)

5.3.2. Operations over a message

We establish three different kinds of operations depending on who can perform them :

a) A human agent
b) the electronic mail service
c) the EPOS system

a) **the human agent**, three major kinds:
- to create/modify [compose (receiver, comm_source, confirm_reception, text,..), delete, ...]
- to handle [send, accept, reject, answer, store, classify, ...]
- to inspect [identify receiver, theme, objective, ..., read_the_content]

b) **the electronic mail service**

- when the human agent is creating or modifying a message
The service helps to elaborate the message, checking if the selected value of each descriptor belongs to the defined range and the combination of values are right (type, objective, theme, .. linked with the type of receiver). This will be presented in form of a friendly interface.

- over a message with status = **pending**
The service will fill the nº_of_lines and sender descriptors

- over a message with status = **confirmation_needed**
When the service gives the message to the receiver then build_message (sender, succes , notification) and delete all things related with the message.

- over a message with status = **answer_needed**
This option is more complicated but all standard protocols of communication have defined a systematic behaviour to handle this situation.

c) **the EPOS system** (probably the Monitoring System)
- The Monitoring system will need to extract information to elaborate statistics, administrative reports, etc.

5.3.3. Storage and agenda

Most of the electronic_mail services provide folders to store the user messages. The structure of a folder is a sequence of messages, where the order is usually the receiving or storage date. Some current research projects aim to enhance the storage structure moving towards data base organization. On the other hand it is useful to have an agenda linked to the electronic_mail to record information such as sending, requesting or answering messages on a specific date. The next paragraphs elaborate further on these ideas.

5.3.3.1. Data bases of user messages

It seems adequate to have a structure allowing to search and store messages applying different and flexible criteria for selecting them. We propose using a relational data base to store the user messages. In this way the operations:

store, classify, look_for,print, delete, ... messages

will be expressed like queries in a relational language. The relations and attributes should be the same as the message descriptors. Select sentences with restrictions will provide a powerful way of inspecting messages as for example:

select received messages from student X with type consult and objective teaching

5.3.3.2. Agenda

A user agenda is a data structure containing tasks, more precisely , a sequence of tasks ordered by date: day (0 to 24), month, and year.

The operations on an agenda are: creating and agenda, inserting a task in the agenda, deleting a task from the agenda, searching for tasks in the agenda. A task definition and a detailed description of operations follow.

Task

Three type of tasks can be considered:
Send a message X, with subject Y related to message Z
Appointment with X theme Y related to message Z
Answer required from X with theme Y related to message Z

A task can be defined as a structured object with the following slots:

task
structure
type: [send_message, appointment, answer_required]
interlocutor: epos_agent
references: { theme, message}
status: [finish, pending]

For the first type of task further restrictions can be defined:
send a message to X with subject Y related to message Z

X: should be the identification of an epos_user. If it is already the electronic_address no further manipulations are needed to execute the task.

Y: there are two possibilities. (1) to use the descriptors type, objective and theme, or a free one. In the first case the execution of the task can proceed automatically, in the second one the user will be requested when composing the message

Z: Identification of a message, again two possibilities, first one a date and a receiver, second one a free text. As commented above, in the first case the task can be done automatically while in the second case the user will be required to compose the message.

Some examples follow:

Send message to the tutor

Send message to the tutor, subject the collection of problems

Send message to the tutor, subject my work related to message homework

Send message to X type request, objective authorization, theme activity related to 30-6-90,sender Y.

Operations on agenda:

1. Create an agenda for a user

Parameters: initial_date, task_insertion_procedure,task_delete_procedure, defaults

Task insertion procedure: has to provide the possibility of automatically creating tasks when receiving/sending some kinds of messages

Defaults: indicate the amount of information about tasks given automatically to the user

2. Insert task into agenda

Two modes have to be available, one to create a task when reading a message, related to it (for instance give an answer some time later) in such a way that the editor will automatically fill some descriptors, the others being edited from scratch.

3. Delete task from the agenda

In this case two modes are also considered, automatic deletion, if the date is over, or the task has been finished, and explicit elimination using search operations (find and delete)

4. Search for tasks in the agenda

Different options should be taken into account such as all the tasks in a date, all the tasks in a period of time, all the pending tasks, all the tasks related to a user, etc...

5.3.3.3. Sessions of communication

An interesting idea concerns the notion of session of communication. We propose the following definition : A set of messages interchanged between two interlocutors, with the same subject. Some current services provide a way of linking messages implicitly when using a reply command, but we think this mechanism should be made explicit and more selective.

A conceptual structure is proposed below, a session has a subject, a set of participants, a frequency, an initial message to start the session, a set of messages and a terminating condition. This point should be further investigated.

Session of communication :
structure
subject:
interlocutors:
frequency:
messages
Initial : notification from a to b,
Development: messages from b to a, a to b
End:

This kind of structue is useful for different purposes, for instance:
1. a tutor sending information to the manager about students following a curriculum
2. two students sharing homework during a period of time
3. the development of a negotiation between two epos-agents

5.4. Towards a formal specification for message content

The main purpose of section 4 was to figure out situations where human-human communication in epos is needed. A first classification attending to purpose was given. Our goal now is to refine for each class, the type of message content taking into account semantic criteria. In this way we set out guidelines for the design of an interface. Furthermore this specification can be used for enhancing the production and classification of messages as well as extending epos to include automatic answering processes.

5.4.1. Messages for consultation of information or advice

These are the typical messages asking about knowledge included in epos knowledge-bases and in consequence the Tutoring system should be able to respond, however some class of users should also be able to formulate them directly to the KB.

There are two main types of consulting operations concerning either (1) the identification of an object partially described or (2) the values for some properties of some object.

<content_message_specification > :: = <determine_object >
/ <determine_property >

An example for a student-tutor scenario follow:

a) *Is there a concrete curriculum about natural sciences, with advanced difficulty level and estimated workload no more than three weeks?*
this question can be formulated to the curriculum information system in terms of a search operation for an object partially described, in the style of:

search an object X,
instance of curriculum, with the following properties:
content.subject.area: natural_sciences
content. difficulty_level: advanced
estimated.workload < 120 hours

The syntax for these type of questions is defined by the following BNF rules:

<determine object>::= <Operation1> < quantifier> < element_description>
<property>
<determine property>:: = for < identifier > <Operation2> <argument>
<Operation1>::= search / how many
<Operation 2>::= findvalue
<quantifier>::= one / some / each / the most <criteria>
< element_description > ::= object_reference / <identifier>
<identifier> ::= object_reference / object. property_name = value
<property> ::= < argument> {<logic_operator> <argument>}
<argument>::= < structural_relation > /
< structural_relation > <identifier>
property_name /
property_name object_reference /
property_name <comparative_operator> value

Other examples concerning both types of questions are given in b to h below:

b) *What are the prerequisites for a course to became a communication system specialist*

For X , instance of curriculum
with curriculum.aim = system specialist
findvalue of
target audience. prerequisites

c) *For the operator curriculum, is there educational software available and with what requirements?*

This sentence has to be reformulated as a two step question, first findvalue for a property of an object, and then for each X, object of the first question solution set, find the value of the property

X.organization.hardware.software

d) Who can advise me about language courses?

There are at least two ways of reformulating this question, to select one of them, optimising criteria should be considered.

1. search for a tutor able to advise this user on languages courses
search an instance of tutor
with
comunication = {consult, advice, epos_resource}
configuration.to _whom = *this user type*
subject_matter_spec = languages courses

2. for each curriculum on language, findvalue of associated tutors able to advise this user

Other examples following these patterns:
e) How many modules are in the salesman technician curriculum ?
f) Which course do you recommend to learn the use of the X word-processor ?
g) Who is the responsible for the course X ?
h) How can I change my teaching plan to skip some modules I already know ?

5.4.2. Messages for consultation of instructional activities

In this case we distinguish four main classes; first one, called declaration, corresponds to situations where the student provide the tutor with some comments about his state of knowledge, such as *I don't understand X,* second one, named request, where the student asks for some complementary material for his learning activity, for example *can you give me the solution of x.* The third class groups diverse questions related to the subject-matter, in order to obtain further information from the tutor, and finally the fourth one concerns demands for different kinds of explanation. The proposed classification is further elaborated below.

Class	*Subclass*
declaration	about the state of knowledge or understanding
request	**an instructional object**
	example
	solution
	exercise
	test
	for confirmation
	must I do x ?
information	**what is**
	definition of: concept, property, relation, procedure,action.. in general terms, more detailed than, analogue to,different from,
	which/what
	instances, sets, ...satisfying a partial description
	is it true
	property, relation, action...
	is X an alternative to Y
	What is the importance of X

explanation **why P**

conditions that enable P
facts causing P
justification, finality, motivation of P
consequences of P

How

to proceed
finality or motivation

For what
From where
what if

5.4.3. Messages for consultation of other activities

The general pattern for this class consider two main types of message contents: to inquire about the agent_epos responsible for an activity in epos, and to be informed about the way to proceed in order to perform an activity. Both can contain a statement where the user conveys some information about himself or his specific purpose.

given X
who is responsible for <operation> /
how to proceed for <operation>

where
operation can be
register for a course, an examination,...
obtaining permission
including something or someone in epos

5.4.4. Messages to negotiate

Depending on the activity three main classes can be distinguished, one of them relatedto making an agreement for an external_activity like an appointment, substitute someone, or any kind of event, the other two concern epos resource use and epos resource update respectively.

external_activity make an appointment /perform activity, will be specified by giving
when, for what and where

share a resource/ include a resource will be stated by indicating
which one, for what period and under what set of conditions

5.4.5. Messages to command

This class contain messages where an order to perform an activity is transmitted from and to an human_epos_agent, for instance a tutor indicating to a student some homework to be done, or telling him to attend a meeting.

5.5 Synthesis of requirements

We conclude this section summarizing relevant criteria in order to design or select a communication system for Epos. Requirements can be grouped under three main features: adaptability, user-friendliness, and extensibility.

1. **Adaptability**, in order to customize the diverse communication services to epos, for instance

- facilities to define users and control the way they can operate with different services attending to the organization and restrictions defined in epos
- electronic educative newsletters
- educative forums
- user announcement blackboard
- event calendar
- transfer of documents (administrative, exams, notes,...)
- interactive consultation, and debate

In this paper we have focused on interpersonal communications, but it is worth noting the increasing importance of multipersonal communications, and experience shows that these kind of applications are real and add to the value of the telecommunication services, challenging the educative community for new ways of cooperative learning and communication. Educative forums for instance break the barriers between schools and between school and society. This area merits a deep study in Epos.

2.**User friendliness**. Some general recommendations follow:

- homogeneity : basic keyboard functions should be simple and the same for all services.
- adequate user level: the interaction should be adapted to user profile, offering customized menus for a type of user, so that the interaction could be gradually adapted from a guided style to a more direct command way.

3. **Extensibility**: some services should be extended to enhance current functionalities and to provide a basis for further automatisation, in particular related to the mailing system the following aspects have to be considered:

3.1. Information service

The classical directory, and other sources of educative information currently implemented in files or databases should move to a knowledge base formalism integrating agents, curricula, teaching plans and educative products.The knowledge-base (KB) will be consulted by different epos-artifacts, for instance the tutoring system when carrying out a dialogue with a user, or the mailing system helping to send a message to an appropriate receiver, but it is also possible to allow some types of users to consult directly this KB.

3.2. Messages

3.2.1. To **enrich the structure of a message,** in particular a refinement of the current subject, to distinguish type, object and theme. Some of these properties enhance usual operations, such as:

cataloguing and storing the message by content
checking properties values for a message
designing a content-structured message editor

as well as opening up the possibility of automatic answering for some kinds of messages in the near future.

*3.2.2.*To design a data-base like model for message storage, in order to inspect messages in a flexible way, not restricted to folder and chronological order.

3.3. To introduce an agenda in order to manage tasks related to the electronic mail.

6. An interface for the electronic mailing service

As we have already commented in previous sections, 5.3.3., the interface design must allow the human agent to work in a comfortable, pleasant and simple way with the following elements: the electronic mailing service, the relational data base,which contains objects in message form, and the task agenda of the user.

The interface can be considered initially as an element of the Tutoring System. As such, some of the services and assistance offered to the user by the Tutoring System can be given through the interface, for example, obtaining the addresses of possible interlocutors.

The interface will allow the user, through a series of multiple windows **working simultaneously** and using menus, to work with the relational Data Base, the task agenda and the electronic mailing. The Data Base and the agenda will be created when the user obtains his category as human_epos_agent and in accordance with the characteristics of his associated electronic_mail_profile. On the other hand, a help option at each work level will allow the user to get to know the working of each of the operations. Another option offered to the user is the printing of information obtained or elaborated, where relevant. The interface also includes a structured text editor to compose the content of messages.These features make it possible to create a user interface easy to use. Figure 1 shows the different components and their relationships.

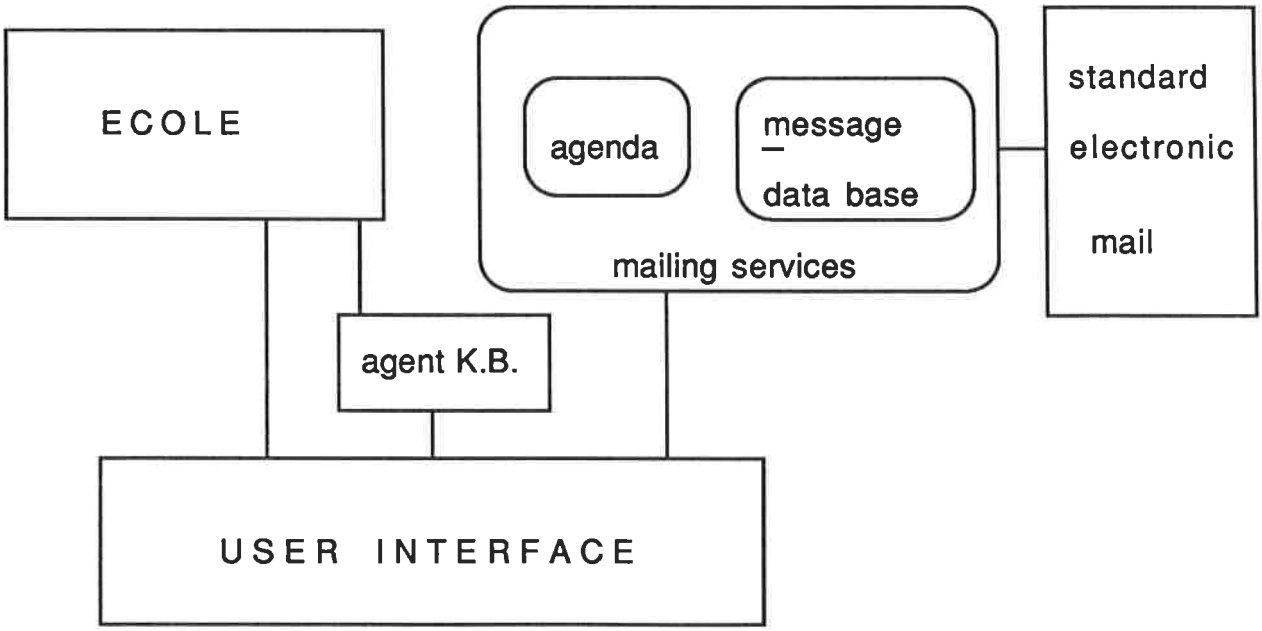

Figure 1: the user interface for ecole and the mailing services

The initial menu which the interface will present to the user is given below. One or more of the options can be selected:

1. Work with the Data Base
2. Work with the agenda
3. Work with the electronic mail
4. General information and user profile
5. Help
6. Dialogue with the tutor
7. Exit

In the following sections we are going to specify all the operations offered by the options 1-4 of this inital menu assuming that, for each of these, there is a help and a print option although these are not mentioned.

6.1. Operations on the relational Data Base (DB)

The DB consists of objects belonging to the message class and they are interrelated by the content of some of the descriptors of these objects. The DB should include at least those relationships which allow the following consultations to be made:

a) all messages with the same sender/sender class
b) all messages sent or received the same day, month, year
c) all messages belonging to a period of time or before a certain date
d) all messages of the same type
e) all messages with the same objective
f) all messages with the same theme
g) all messages with the same status
h) any combination of all previous criteria using the operator and
i) all messages (no restrictions)

The user will have the possibility of defining other criteria for consulting using the information/modification operations of the electronic_mail_profile which we will see further on.

All messages received through the electronic mail will be automatically introduced into the user's DB unless some kind of barrier has been established for certain kinds of messages (messages of a certain type or from one particular sender, etc.). However, the messages sent by the user will not be treated in the same way as those received if the user so wishes.

Assuming the user's DB satisfies all these requirements, the interface will offer the user the following possibilities:

1. **Add** e_message **to the DB**
   input: e_message: message sent by electronic mail from this user
   DBU: DB belonging to the User
   temporary_zone

process: insert the e_message in the DBU establishing all relationships according to the criteria from a)-h) and those which the user may have defined between the new element and those which already existed. An identifier is defined for the message (id_message) which will also know the electronic mail and will be used to update the status of the message. The message is also deleted from the electronic mail.

output: the DBU with the addition of one element and the electronic mail with one element less. From this point any kind of operation to be carried out on the recently included message must be done using one of the operations offered by the DB.

2. **Determine_consult_criterion**

input: DBU: DB belonging to the user
list_c_c: the list of existing consulting criteria

process: the user selects the criterion or combination to carry out the consultation and idicates all the parameter values to carry out the search. It would be useful if the syntactic/semantic parser could indicate possible errors or incoherencies in the values assigned to the parameters of the chosen criterion.

output: consult_criterion instantiated with the values given by the user.

3. **Consult** consult_criterion

input: the present consult_criterion. It can be empty which would mean that there is no criterion in action. By default we can consider the empty consult_criterion as belonging to option i) (all messages).

process: shows the user the criterion it will use to select messages.

output: consult_criterion in action.

4. **Obtain_heads** message_list_cc **of the DB**

input: consult_criterion: the last one selected by the user
DBU: DB belonging to the user

process: consults the DBU and extracts all the message heads of the messages that satisfy the criterion selected by the user as well as their respective identifiers and shows them to the user. If no such messages exists the user is notified.

output: head_lis_cc : list of message heads that satisfy the consult_criterion.

5. **Show** message_list_cc **of the DB**

input: consult_criterion: the last one selected by the user
[head_list_cc: list of message heads that satisfy the consult_criterion]

process: consults the DBU and extracts all the complete messages which satisfy the criterion selected by the user as well as their respective identifiers and shows them to the user. If no such message exists the user is notified. If the previous operation was **obtain_heads**

then it uses the head_list_cc obtained to accelerate the extraction. The operation ends when there are no more messages which satisfy the consult_criterion or if the user so wishes.

output: message_list_cc: list of messages which satisfy the consult_criterion in operation.

6. **Select** message_list_cc **from the DB**

precondition **:** there exists a message_list_cc or a non-empty head_list_cc

input: message_list_cc or head_list_cc
DBU: DB belonging to the user

process: the user selects those heads or messages he can later apply the operations **extract, delete, read** or **reply**. After applying one of these he can make another selection until he decides to end the select operation. Then the message_s_list_id will be destroyed.

output: message_s_list_id: list of the identifiers of the messages selected by the user from those which satisfied the consult_criterion.

7. **Extract** message_s_list **from the DB**

precondition: a message has been selected, if not it has no effect, i.e. message_s_list_id cannot be empty.

input: message_s_list_id: list of identifiers of the messages selected by the user
BDU: BD belonging to the user
destination: the place where the message have to be deposited.
possible destination values: electronic mail, file, agenda, etc.

process: extracts from the DBU all messages whose identifiers are in the message_s_list_id without deleting them from the DBU and copies them in the place indicated by the user.

output: list of messages copied onto the destination place. DBU is not modified.

8. **Delete** message_s_list **from the DB**

precondition: there must exist a selected message, the list of identifiers cannot be empty, if not it does not have any effect.

input: message_s_list_id: list of identifiers of the messages selected by the user
DBU: DB belonging to the user

process: deletes from DBU all messages whose identifiers are in the message_s_list_id and reorganises all the relationships between the messages which are sitll in DBU.

output: DBU from input with less elements and reorganised.

9. **Read** message_s_list **of the DB**

input: message_s_list_id: list of identifiers of the messages selected by the user from those which satisfied the consult_criterion.
DBU: DB belonging to the user

process: shows the content of the selected message. The operation ends because the user so wishes or because the message_s_list_id has

run out. The user can has the choice of visualising the whole message or only its identifier, head an associated text. It can also reply to the message inmediately (see operation 10)

output: list of selected messages

10.**Reply_to** message_s **of DB**

input: message_s: a selected message that has just been read from the DB

process: links up with the operation **compose** e_message of the menu of the electronic mail to allow a reply message to be composed.

output: empty. The reply message is built up using **compose**.

In the operations described up to now we have referred to the term message head. By this we mean the folloving message slots: sender, date_of_issue, date_of_arrival, type, speech_act, objective and theme. The message identifier, the only one in the system, can be included or not in the message object itself.

6.2. Operations on the agenda

According to the definition of agenda and task given in 5.3.3.2., the agenda gives support by default to the folloving set of search operations:

a) all tasks with the same status
b) all tasks of the same type
c) all tasks belonging to the same day or period
d) all tasks before or after a certain date
e) all tasks with the same interlocutor or class of interlocutor
f) all tasks (with no restrictions)

A task will be inserted in the date it must be carried out. There are two methods of insertion :

1. Manual which we will describe below
2. Automatic

   - Every time the mail receives a message for the user requiring a reply (answer_claim=yes) the interface, besides including it in the DB, creates a task of the type answer_required and inserts it in the day of the message is received. The process is the same when the message requiring a reply is sent by the user himself.

   - Every time the mail receives a message with an appointment theme or this kind of message is sent by the user, the interface creates a task of the kind appointment and inserts it in the day of the appointment, if known, or in the day the message was sent or received.

The user can carry out the following opertions on the agenda :

1. **Create** task

   input: a) *newly created task bearing no relation to any of the tasks or messages in the previous DB*

agenda_u: the user's agenda
b) *we are in the mail menu or in the DB menu and want to create a task related to the current message*
message_r: message related to the task to be built up coming from the current operation on the mail or on the DB
agenda_u: the user's agenda

process: a) the user must indicate the date it must be inserted and select one of the three possible types of task: send_message, appointmet or answer_required. Following this he must indicate what the task is related to either with a free text or using the operations **address_list** and/or **select_address** from the information menu. By introducing a free text the value of the descriptor references will be determined.

output: task_c: task created

2. **Insert** task_c **in the agenda**
   input: task_c: task created
   agenda_u: the user's agenda
   process: the day of the task is taken and inserted here
   output: agenda_u with an additional element

3. **Determine** search_criterion
   input: list_cb: list of search criteria (from a) to f))
   agenda_u: the user's agenda
   process: the user is shown the possible search criteria in the agenda and the user chooses one of them.
   output: search_criterion chosen

4. **Consult** search_criterion
   input: chosen search criterion. If the search_criterion is empty, by default criterion f), all noted tasks, is taken.
   process: shows the user the search criterion in action
   output: current search_ criterion

5. **Show** task_list_cb **of the agenda**
   input: chosen search_criterion
   agenda_u: the user's agenda
   process: consults the agenda and takes out all the tasks that satisfy the selected search criteria and shows them to the user. If none are present it gives a warning.
   output: task_list_cb: list of tasks satisfying the chosen search criterion.

6. **Select** task_list_cb **according to criterion**
   precondition: task_list_cb must not be empty
   input: task_list_cb: list of tasks satisfying the search criterion
   agenda_u: user's agenda
   process: the user selects the tasks on which he can later apply the operations **delete, list** or **compose**

output: task_list_s: the list of selected tasks

7. **Delete** task_s **from the agenda**
   precondition: task_list_s not empty
   input: task_list_s: list of selected tasks
   agenda_u: user's agenda
   process: deletes the tasks found in the list. The deletion can be done one at time or on the whole list. In a certain sense this operation could be reduced to modifying the status of the task and placing it in finish.
   output: agenda_u with less elements

8. **List** task_s **from the agenda**
   precondition: task_list_s not empty
   input: task_list_s: list of selected tasks
   agenda_u: user's agenda
   process: shows the user the content of all selected tasks in ascending order by date. They are shown one by one so they can be deleted or sent to the mail menu and a message can be composed.
   output: the same input list with no modifications

6.3. Operations on the electronic mail

Regardless of the electronic mail system chosen, the interface proposed has to allow the user to carry out the following operations taking into account that the messages received and not rejected go directly to the user's DB:

1. **Compose** e_message
   input: a) *if the message is newly created then*
   the default parameter values for the creation of the e_message
   the value of the identification descriptor
   b) *if the message text has already been built with an editor outside the interface then*
   exactly the same as a) option plus
   the name of the file containing the message text
   c) *if the message comes from a copy of an existing one in the temporary storing area of mail messages then*
   a completely built up message (e_message_t_s)
   d) *if the message is the reply to another existing one in the DB which has been selected then*
   exactly the same as in a) plus
   the values of the sender descriptors and the characterization content (type,speech_act, objective and theme) of the message that has just been selected in the DB (one belonging to message_s_list)
   e) *if the message is related to a task in the agenda then*
   exactly the same as in option a) plus
   task_s of the agenda

process: a) the head of the message is created filling in some of the message descriptors with the content of the default parameters and with the user identification value. Then using the operations of information/modification of the electronic_mail_profile **address_list** and **select_address** the receiver descriptor value is filled. Then the user decides the message's descriptor type and in function of this value the user is offered the possible values for speech_act,objective and theme in this order. In message characterization for each type of human_epos_agent the valid combinations are found. Finally a structured text editor allows him to fill in the content of the message. The characteristics of this editor are described in 5.4.
b) the process is the same as in option a) except that the structured text editor does not participate in the process
c) the user will be able to modify any of the descriptors of the message that is already built up with or without the structured text editor.
d) a similar situation to c) as the head is fully defined and the only thing left is to create the reply message which can be done using the structured text editor or not.
e) the user creates the message using all the information contained in the agenda task. When the message has been sent the task status automatically becomes finish.

output: the built e_message

2. **Store** e_message_t
   input: the newly composed e_message
   temporary_zone where the message has to be deposited
   process: stores the e_message in the temporary_zone and assigns an identifier to it. Any later operation on this message (**send**, **delete**, **add** to DB, etc.) has to be done through the operation **select**. The status value of this message will be put on stand-by for sending.
   output: id_e_message_t: identifier of the message included in the temporary zone
   temporary_zone with one additional element.

3. **List_cab** e_mensaje_t
   input: temporary_storage_zone
   process: shows the user a list (id_e_message_t and cab_e_message_t) of all messages stored in the temporary_zone.
   output: list_e_message_t: list of all messages stored in the temporary_zone

4. **Select** e_message_t
   precondition: list_e_message_t not empty, i.e. after the operation 3 has been carried out
   input: list_e_message_t: the list of all the messages stored in the temporary_zone

temporary_storage_zone

process: the user selects the messages on which he wishes to aplpy one of the following operations : **send, delete, read, copy** or **add** to DB

output : list_e_message_t_s: list of all the messages stored in the temporary_storage_zone which have been selected up to now.

5. **Send** e_message

precondition: an e_message has just been created or one has been selected from the temporary_zone, i.e. list_e_message_t_s is not empty.

input: e_message: a fully composed message
temporary_zone and
DBU: user's DB

process: checks for possible errors in the message in question and sends it if none are found. If not the user is notified and the message is not sent. If the user has decided that all the messages he sends have to go to his DB, the recently sent e_message will disappear completely from where it was and will be placed in the DB. If not it will there with the value of sent status and with its corresponding identifier. In this way the mail can modify the status of the message when necessary.

output: depending on the input situation, the temporary_zone will be plus or minus one element and/or the DB plus one element.

6. **Read** e_message_t_s

precondition: there exists a non empty list_e_message_t_s, i.e. messages have already been selected from the temporary_zone.

input: list_e_message_t_s: list of messages selected from the temporary_zone
temporary_zone

process: shows the user the content of the messages selected from the temporary_zone in the order they appear in the list_e_message_t_s without modifying the content of the list, so that the operations **send, copy, delete**, etc. can later be applied. The operation ends when requested by the user.

output: same as input.

7. **Delete** e_message_t

precondition: there exists a non empty list_e_message_t_s

input: list_e_message_t_s: list of selected messages
temporary_zone

process: deletes from the temporary_zone and from the list the message where the operation has been applied. Elements can be deleted one by one from a list or all at once.

output: list_e_message_t_s less the number of elements deleted
updated temporary_zone

8. **Copy** e_message_t
   precondition: there exists a non empty list_e_message_t_s
   input: list_e_message_t_s: list of selected messages
   temporary_zone
   process: creates a new message, an exact replica of the one being copied, and automatically applies the operation **store** to include it in the temporary_zone.The only difference between the original and the copy is the associated identifier. By then applying the operation **select** and **compose** onto the copied message the same message can be sent to different people. The original message does not undergo any changes.
   output: the same list_e_message_t_s as in input
   temporary_zone with a new element
   id_e_message_t: identifier of the message copy

6.4. Operations of information/modification of the user's electronic_mail_profile

Here the set of operations are described which allow the user's electronic_mail_profile to be consulted and any modifications made (object n. 43). These make the task of creating a message and giving information to the user on what has happened since his last work session easier.

- relating to the ALIAS descriptor

1. **List_addresses** d_user
   input: table of electronic_addresses taken from the alias descriptor of the user's
   electronic_mail_profile
   process: shows the addresses the user can send his messages to from the table of electronic_addresses
   output: same as input

2. **Select_addresses** list_d_user_s
   input: table of electronic_addresses taken from the alias descriptor of the user's electronic_mail_profile
   process: the user selects one or more addresses from the table. The selected addresses will make up the descriptor for receiver of the message to be built or in the process of being built or they will be later deleted
   output: list_d_user_s: list of addresses of selected users

3. **Insert_addresses** d_user
   input: table of current electronic_mail_addresses
   d_user: electronic_mail_address [and alias [and explanatory text]]
   process: adds the new electronic_mail_address togeher with its alias and an optional explanatory text to the table of
   electronic_mail_addresses.

A check will be made for repetition of addresses and alias as well as the syntactic correction of the address

output: table of electronic_mail_addresses with one new element

4. **Delete_addresses** d_user
   input: table of electronic_mail_addresses
   list_d_user: list of addresses selected from the table
   process: deletes all addresses contained in the list from the table
   output: updated table of electronic_mail_addresses

- Relating to IDENTIFICATION descriptor

1. **Consult** user_identification
   input: user's electronic_mail_profile
   process: consults the value of the identification descriptor of the user's electronic_mail_profile and shows it to the user
   output: empty

- Relating to MESSAGE_STORAGE descriptor

1. **Define/activate** warning_input_DB
   precondition: there must exist a DB
   input: information which will be given to the user every time a work session is started on the DB. For example, from among all the warning_input_DB descriptors the user can select the following: no. of messages received that day, no. of messages rejected, total no. of stored messages, etc.
   process: the user selects which descriptors of the object warning_input_DB should be visible at the beginning of a session with the DB. From then on this information will be displayed to the user.
   output: the object warning_input_DB instantiated

- Relating to DEFAULT_PARAMETERS

1. **Insert_auto** e_message **in DB**
   input: user's electronic_mail_profile
   process: shows the user the value of the automatic_insertion slot of the default_parameters of the electronic_mail_profile belonging to the user. The user can modify this value. If the value is YES then whenever the user sends a message it will disappear immediatly from the temporary_zone and will be passed on to the DB (see operation 5 of mail). If the value is NO, then it is up to the user to make the transfer manually from one place to the other applying the **select** operation from mail and then the **add** operation of the DB
   output: user's electronic_mail_profile with or without modifications

2. **Reject** e_message
   input: user's electronic_mail_profile
   process: the user is shown the working restrictions to reject messages. The value by default is the non-existence of restrictions. The user can also describe which combinations of values of certain descriptors of messages received are not allowed or are forbidden. When the mail receives a message for the user which contains one of the combinations given in this operation, the system will give a warning which it will leave in the user's DB indicating that a message has been rejected. Two different kinds of messages can be rejected: messages from one particular user or user class, and/or certain types of message. Rejection can be of two kinds: implicit, the user class it belongs to (for example, communication between an author and a student makes no sense, and if this is attempted for any reason the interface will not allow the message to be composed), or explicit, built by this operation.
   output: user's electronic_mail_profile with or without modifications

3. **Def_consult_criterion** list_c_c **of the DB**
   input: list_c_c: list of existing consult criterion. By default the content of this list is the relationship described in 6.1. from a)-h)
   process: the user defines more complex processes for access to and recuperation of information in the DB, using a programming language or some specific tool.
output: updated list_c_c

4. **Default_composition** e_message
   input: user's electronic_mail_profile
   process: the user consults and can define the values by default of any of the message descriptors which will be used by the **compose** operation of the mail. This is the descriptor list whose value can be defined by default: sender whit the result of the **consult** user_identification, communication_source with the same value as sender, source_visibility to yes, scope with private, confirm_reception to no, answer_claim to no and answer_due_deadline to 3.
   output: user's electronic_mail_profile with or without modifications

5. **Delete_auto** task **in agenda**
   input: user's electronic_mail_profile
   process: shows the user the value of the automatic_delete slot of the default parameters of the instance of the user's electronic_mail_profile. The user can modify this value. If the value is yes this means that the moment a task written in the agenda takes on status=finish it will be automatically deleted. If the value is no then the user must delete the task from the agenda himself.
   output: user's electronic_mail_profile with or without modifications

- Relating to the AGENDA_DEFINITION descriptor

1. **Define/activate** warning_input_agenda
   precondition: there must exist an agenda
   input: the information displayed to the user each time a working session with the agenda is begun. For example, from all the warning_input_agenda descriptors the user can select the following: list of dates with tasks to be done, description of the tasks to be done that day, number of created tasks since the last session, description of the tasks which have not been carried out in the given time, etc.
   process: the user selects which descriptors of the object warning_input_agenda should be visible at the beginning of a session with the agenda. From then on the information chosen will be displayed to the user.
   output: instance of the warning_input_agenda

Annex I. Case Study: Telefónica Formación

I.1. Description

I.1.1. The Training Program

The Training Program of Telefónica is an annual plan where all the courses, either for entrance into the Company or promotion within the Company or to acquire a specialization are defined.

A Training Program is formed by a promotion plan and a specialization plan, and they are valid for one year. The program is organized by regional areas, so that a program is defined as the set of regional programs.

A promoting plan is organized into lines, and for each of them diverse courses are specified. A course corresponds roughly to the idea of professional curriculum in [VedHel-89], but including four levels of description: curriculum, modules, subjects matters, and learning units.

The lines considered are: stores, crafts, phone operators, administration, commercial, computer applications for bussines and administration, external plant and internal plant. Each line is organized as a hierarchy of a maximun of five categories , from the bottom to the top: auxiliary, operator, head of team, technician and engineer, however the three first lines do not include the two top level categories.

A specialization plan is structured by areas, and within an area diverse courses are defined, each one corresponding to a thematic curriculum, divided into subjects and learning units. The areas considered follow: network, commercial, administration, office computer applications, leaders, teachers and monitors.

I.1.2. Agents: their characterization and functions

Central Manager

Description :S/he is an employee of The Training Service of Telefónica belonging to the central organization of Telefónica

Functions: authorize the annual training program created by the cental designer
authorize modifications to the program suggested by provincial managers
decide and call for regular meetings with provincial managers
notify the annual training program to provincial managers
consult with provincial managers

Provincial Manager

Description : s/he is a Telefonica employee belonging to the provincial organization of Telefonica, being in charge of instructional tasks and in some cases director of a provincial school.

Functions: in charge of administering a provincial training plan
configurate a provincial training plan
negotiate modifications to the plan with the central manager
authorize a student or a tutor as epos_agents
consult with students
negotiate with tutors
consult with tutors

Central Designer

Description: s/heis an employee of the Training Department of Telefónica belonging to the Central organization of Telefónica

Functions: course design
consult the central manager
propose the annual training plan to the central manager
order the scripts for instructional material from the authort
order instructional material from a producer
coordinate between provincial managers and tutors

Tutor

Description:s/he can be or not a Telefónica employee
In case of being an employee, s/he can be a full or part-time teacher s/he has associated some subject-matters of the curriculum

Functions: give instruction to students
solve doubts about a subject-matter
propose activities to students
consult the provincial manager and the central designer
negotiate with the provincial manager and the central designer

Student

Description: s/hecan belong to another company or being related to Telefonica in two different ways: is a current employee or a candidate for entering the company.
in case of being an employee, there are two subclasses: promotion and specialization
- a candidate has to be in the promotion path
- an employee from another company, has to be in the specialization path

Functions: perform an instructional activity
consult doubts with the tutor
consult the manager on administrative matters
negotiate with the provincial manager

Author

Description: s/hecan be or not a telefonica employee
in case of employees, they are not usually from the central service

Functions: write didactic material
Consult with the designer
Negotiate with the designer

Producer

Description : s/he can be or not a telefonica employee
in case of employees, they are usually from the central service

Functions: Produce didactic material
Negotiate with the designer

I.2. Modelling the domain

In this section the knowledge needed to represent the training program as well as the agents is defined as a set of structured objects following the formalism described in section 5.

I.2.1. The program and its relationship with the abstract curriculum

8.Program

structure
validity :
domain:
scope:

A program has a validity (period of time), a domain of application (company, country,..) and a scope (for instance regional, national,..)

10.Telefonica program
subclass program with validity : annual
domain: telefonica
subclasse_are Telefónica provincial program

11. Telefonica provincial program
is_composed_of promotion plan, specialization plan
subclass Telefónica program
with scope: telefonica-region

A provincial plan is a Telefónica program, and then with annual validity and applicable to telefónica, whose scope is one of the regions defined in Telefónica. It consists of two plans, promotion and specialization respectively.

12. Promotion plan
subclasses are line_promotion_plan

13. line_promotion_plan
subclass promotion plan
with line: telef-line
structure
associated_to_curriculum: abstract_curriculum

Any instance of the object *line promotion plan* will have as value for the slot a*ssociated to curriculum* a reference to an instance of the object *abstract curriculum*

14. telef-line : [stores, crafts, phone operators, administration&commercial, computer applications for busines and administration, network and plant]

15. Specialization plan
subclasses are specialization_area plan

16. Specialization_area plan
subclass specialization plan
structure
area:
associated_curicula :

34. area :[network, commercial_administrative, computers, leaders, teachers, monitors]

27.abstract curriculum
subclass instructional design
is_composed_of module

The object *abstract curriculum* corresponds to the definition given in [VedHel-89], it is worth noting that all the attributes specified in [VedHel-89], are inherited here from the object instructional design, defined below.

28. module
subclass instructional design
subclass_are [module1, course]

We distinguish here the possibility of identifying module as a set of course (module 1) or as one course, in this way the levels for curricula can be four or three respectively.

32. module 1
    subclass of instructional design
    is_composed_of course

33. course
    subclass instructional design
    is_composed_of learning unit

29. learning unit
    subclass instructional design

30. instructional design
    structure
        identification
        objectives
        content
        audience
        didactic feautures
        organization

I.2.2. Telefonica_epos_agents

The type of agents to be represented are: central manager, provincial manager,tutor,student, author and producer. As indicated by Telefonica, the designer should not been considered. Definition of the rest of the agents are given below, following the notation introduced in section five.

5. Company
   Instances_are : Telefónica

One of the instances of companies is Telefónica. A company should be also a structured object but for our purposes this is enough.

17. epos_telefonica _agent
    subclass epos_agent with affliliation: Telefónica
    subclass_is human_epos_telefonica_agent

18. human_epos_telefonica_agent
    subclass epos_agent, person

An human epos agent is defined as an epos agent and a person, inheriting slots from both.

19. Central Manager
subclass employee with employer: Telefonica, workingplace: Central Service
subclass human_telefonica_epos_agent with user_type = manager
with communication: { consult, information, epos_resources}
who_access: provincial manager
to_whom: provincial manager

with communication: {consult, information, personal }
who_access: provincial manager
to_whom: provincial manager

with communication: {negotiation, authorization,activity}
who_access: provincial manager
to_whom: telefonica_human_epos_agent

with communication: {negotiation, agreement, activity}
who_access: provincial manager
to_whom: telefonica_human_epos_agent

20. Provincial Manager
subclass employee with employer: Telefonica, workingplace: provincial service
subclass human_telefonica_epos_agent with user_type = manager
with communication: { consult, information, epos_resources}
who_access:tutor
to_whom: [manager,tutor]
with communication: {negotiation, authorization,activity}
who_access: [manager,tutor, student]
to_whom: [manager, tutor, student]
with communication: {negotiation, agreement, activity}
who_access: manager
to_whom: manager

A tutor is described as a subclass of human_epos_agent, adding some other relevant information: his status, pedagogic evaluation, subject_matter specialization, curriculum where he appears, and its availability.

7. Tutor
subclass human_epos _agent with user_type= tutor
structure
status : [permanent, contractual]
pedagogic.eval
subject_matter_spec
associated_curriculum: abstract_curriculum
consult_timetable

21. Tutor epos telefonica

subclass tutor with subject_matter_spec: one of the telefonica spec
subclass telefonica_epos_human_ agent
with communication: { consult, information, epos_resources}
who_access:student
to_whom: [student, provincial_manager]
with communication: command, execute,activity}
who_access:
to_whom: student
with communication: {negotiation, agreement, activity}
who_access: provincial manager
to_whom: provincial manager
structure: course_given: [abstract_curriculum]

31. Student
subclass human_epos_agent with user_type= student
structure
associated_curriculum: abstract_curriculum

22.Telefonica_epos_student

subclasses are contract_student internal_student
subclass student

structure
registration_number
type: [contract, internal]

23.Internal_student

subclass telefonica_epos_student with type:internal
with communication: {negotiation, autorization, activity}
who_acces: manager, provincial_manager
to_whom: provincial_manager
with communication : consult
who_access : tutor
to_whom : tutor
with communication : {consult, information, personal}
who_access : student
to_whom: student

subclasses_are promoting, specializing
f_type: [promoting, specializing]

24.internal_specializing_student
subclass internal_student with f_type : specializing
employee with employer :Telefónica

35.promoting_internal_student
subclass internal_student with f_type : promoting
subclasses own_student, outside_student

36.own_student
subclass promoting_internal_student
employee with employer: Telefónica

37.outside_student
subclass promoting_internal_student
subclass of candidate

38. Candidate
subclass person
structure
job
call number
company
prof_requisites

Annex II. Case Study: the educational telematic network of Catalonia

II.1 Description

II.1.1 Services

The Educational Telematic network of Catalonia is an institutional project of the Department of Education, covering 6.000 teaching institutions. The following services are available:

- Electronic mail: a vehicle of communication that guarantee privacy of access to the messages

- Questions and answers: a notice board on which to place requirements or proposals for solution. Public consultation.

- Electronic news bulletin: is an informative newspaper organised by sections. Any user can read it.

- File transmission : to distribute notes, programs, ...

- Teledebate: a public discussion. All the contributions are recorded and made available in an organized way for access or future reference.

- Databases: educational resources available to support concrete goals

- Team games

The network is made up of a central computer, some terminal computers and a transport network

II.1.2. Agents in the network

Central Manager
Local Manager
Teacher
Student
Individual_user
Producer
Author (not integrated into the network)

The relationships between agents, services and operations is summarized in the table below:

service	adminis	board		e mail		data base		file trans		teledeb		news	
oper / agent		CR	CO	S	R	CR	CO	S	R	CR	P	CR	CO
c manag	*	*	*	*	*		*		*	*			*
I manag	*												
teacher		*	*	*	*		*		*	*		*	*
student				*	*		*		*	*			*
producer		*	*	*	*	*	*	*	*			*	*

CR: CREATE, CO; CONSULT, S:SEND, R:RECEIVE,P:PARTICIPATE

II.2. OPERATIONS Model

Operation Class	Service
CONSULTING	
personal information	electronic-mail
resources	notice board, electronic news
advice	elec_mail,notice board, data_base
NEGOTIATION	
appointment	electronic_mail
authorization	electronic_mail
DEBATE	
organize	teledebate
participate	

References

[Cas-90]

Casati ,D. Epos collaborative Learning Environment. Delta 7002 WP 2.7.2 Deliverable, February 90.

[Cas-90b]

Casati,D. Minute of the meeeting Sip-Technical University of Catalunya, Wp 2.7.2, March 1990

[Cast-89]

Castells,J. The Educative Telematic Network of Catalunya. Programa de Informática Educativa. Generalitat de Catalunya.1989.

[Epos-89]

Epos Technical Annex, Delta Project 7002, March 1989

[Fal-90]

Falcone,C. Groupware for educational environments. Delta 7002 WP 22 Deliverable, appendix A.

[TFF-88]

Telefónica Formacion. Catálogo descriptivo de cursos.PGN 02001. Abril 1988

[TF-85]

Telefónica. XIII Convenio Colectivo de Telefónica. 1985.

[VedHel-89]

Vedelaar H, Helwig L: The abstract curriculum; a curriculum model for Epos.DIDA*EL 89-20.