Partially Shared Views: A Scheme for Communicating among Groups that Use Different Type Hierarchies

Jintae Lee

Thomas W. Malone

90s: 89-066

March 1988

Sloan WP#: 2052-88

Management in the 1990s
Sloan School of Management
Massachusetts Institute of Technology

Management in the 1990s

Management in the 1990s is an industry and governmental agency supported research program. Its aim is to develop a better understanding of the managerial issues of the 1990s and how to deal most effectively with them, particularly as these issues revolve around anticipated advances in Information Technology.

Assisting the work of the Sloan School scholars with financial support and as working partners in research are:

American Express Company
British Petroleum Company, p.l.c.
BellSouth Corporation
CIGNA Corporation
Digital Equipment Corporation
Eastman Kodak Company
Ernst & Young
General Motors Corporation
International Computers Ltd.
MCI Communications Corporation
United States Army
United States Internal Revenue Service

The conclusions or opinions expressed in this paper are those of the author(s) and do not necessarily reflect the opinion of the Massachussetts Institute of Technology, the Management in the 1990s Research Program, or its sponsoring organizations.

Acknowledgement

This research was supported in part by Wang Laboratories, Inc.; Xerox Corporation; General Motors; and Bankers Trust Company. We thank Kevin Crowston, Kum-Yew Lai, Wendy Kackay, and David Rosenblitt for their comments that helped identify and elucidate problems that we otherwise would not have seen.

Abstract

Many office systems are based on various types of messages, forms, or other documents. When users of such systems need to communicate with people who use different document types, some kind of translation is necessary. In this paper, we explore the space of general solutions to this translation problem and propose several specific solutions to it. After first illustrating the problem in the Information Lens electronic messaging system, we identify two partly conflicting objectives that any translation scheme would satisfy: preservation of meaning and autonomous evolution of group languages. Then we partition the space of possible solutions to this problem in terms of the set theoretic relations between group languages and a common language. This leads to four primary solution classes and we illustrate and evaluate each one. Finally, we describe a composite scheme that combines many of the best features of the other schemes. Even though our examples deal primarily with extensions to the Information Lens system, the analysis also suggests how other kinds of office systems might exploit specialization hierarchies of document types to simplify the translation problem.

Computer-based office systems often use various types of messages, forms, and other documents to communicate information. When all the people who wish to communicate with each other use exactly the same types of documents, translation problems do not arise. However, when people want to communicate with others who use different kinds of documents, some kind of translation is necessary. The needs for such translation seem likely to become increasingly common as more and more diverse kinds of systems are linked into heterogeneous networks.

We have been particularly concerned with one instance of this problem that arises in the context of template-based communication systems (e.g. Malone et.al., 1987b; Tsichritzis, 1982). The problem is how users of such systems can communicate with other users who have a different set of templates. Other examples of the problem may arise when users of different word processing systems wish to exchange documents, or when different companies wish to use Electronic Data Interchange (EDI) standards to exchange business forms such as purchase orders and invoices.

In this paper, we will explore the space of general solutions to this translation problem and propose several specific schemes for solving it. Our primary goal has been to design extensions to the Information Lens system (Malone et.al., 1987a; Malone et.al., 1987b) that allow different groups to communicate with each other when (1) the groups use some, but not all, of the same types of messages, and (2) the message types used by each group may change over time. In addition to these extensions to the Information Lens system, we have been pleased to find that the analysis has implications for other kinds of document interchange as well. One of the most important general lessons of our analysis is that several novel and attractive translation schemes are possible when the document types are arranged in an *inheritance hierarchy* with certain types of documents being regarded as *specializations* of others.

In the first section of the paper, we illustrate the problem as it arises in the context of the Information Lens system. In the second section, we state the problem more precisely in terms of the objectives that we want its solution to satisfy. In Section 3, we explore the space of possible solutions by suggesting a dimension along which to partition the space and examining a representative solution for each class. In Section 4, we propose a new scheme that combines most of the desirable features of the solutions we explored. Finally, we conclude by hinting at the implications that this research might have for more general contexts.

1. Illustration: Inter-group communication in the Information Lens system

1.1. Description of the Information Lens system.

The Information Lens is an intelligent system for supporting information sharing and coordination in groups and organizations. It helps people filter, sort, and prioritize electronic messages they receive; it helps them find useful messages or other documents they would not otherwise have seen; and it supports common actions people may take on receiving certain kinds of messages.

More specifically, the Lens system enhances the usual capabilities of an electronic mail system with four important optional capabilities, that individual users may or may not choose to use: (1) Senders can compose their messages using structured templates that suggest the kinds of information to be included in the message and likely alternatives for each kind of information. (2) Receivers can specify rules to automatically process their incoming messages on the basis of the same dimensions used by senders in constructing messages. In some cases these rules filter or classify incoming messages into folders; in other cases, the rules may automatically take actions such as replying to or forwarding the

messages. In still other cases, the system takes no automatic action but simply suggests actions that a person might take on receiving a message of a certain type. (3) Senders can include as an addressee of a message, in addition to specific individuals or distribution lists, a special mailbox (currently named "Anyone") to indicate that the sender is willing to have this message automatically redistributed to anyone else who might be interested; and (4) Receivers can specify rules that find and show messages addressed to "Anyone" that the receiver would not otherwise have seen.

In addition to electronic mail, bulletin boards, and conferencing, this basic framework supports a surprising variety of other applications including task tracking and simple calendar management. The system is described in much more detail elsewhere (Malone et.al., 1987a; Malone et.al., 1987b).

For our purposes here, a central feature of the system is that it depends on a set of *semi-structured message templates* or "message types." For instance, Figure 1.1 shows a sample template for a meeting announcement message. The template contains fields for Time, Place, and Date, as well as the usual fields found in all messages (such as To, From, and Subject). The display-oriented editor uses pop-up menus to suggest alternative values for different fields, but users can put any value they desire in any field. A similar editor is used to create rules for automatically processing incoming messages of a given type based on the values of the same fields used by senders to compose messages of that type.

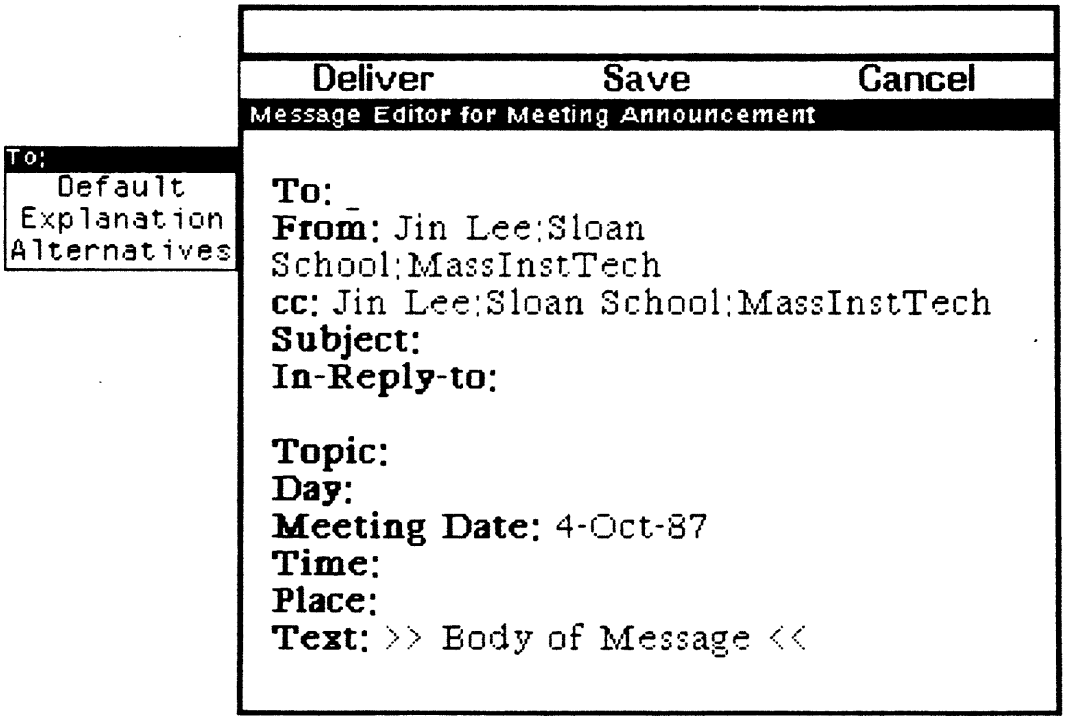


FIGURE 1.1 A template for a meeting announcement message is displayed in a message editor.

The set of message types currently in use by our research group at MIT is shown in Fig. 1.2. As shown in the figure, the message types are arranged in a network with some templates designated as subtypes of others. The subtypes of a given template can automatically *inherit* from the *parent* template the field names and other properties (such as alternatives for field values, and rules for processing incoming messages). Any subtype may also, in turn, add new fields or override any of the property values inherited by the parent (Fikes and Kehler, 1985). For example, the Action Request message type has the field Action Deadline in addition to the usual fields such as To, From, and Subject. Its child, Bug Fix Request, inherits these fields and has additional fields such as Bug Name and Severity.

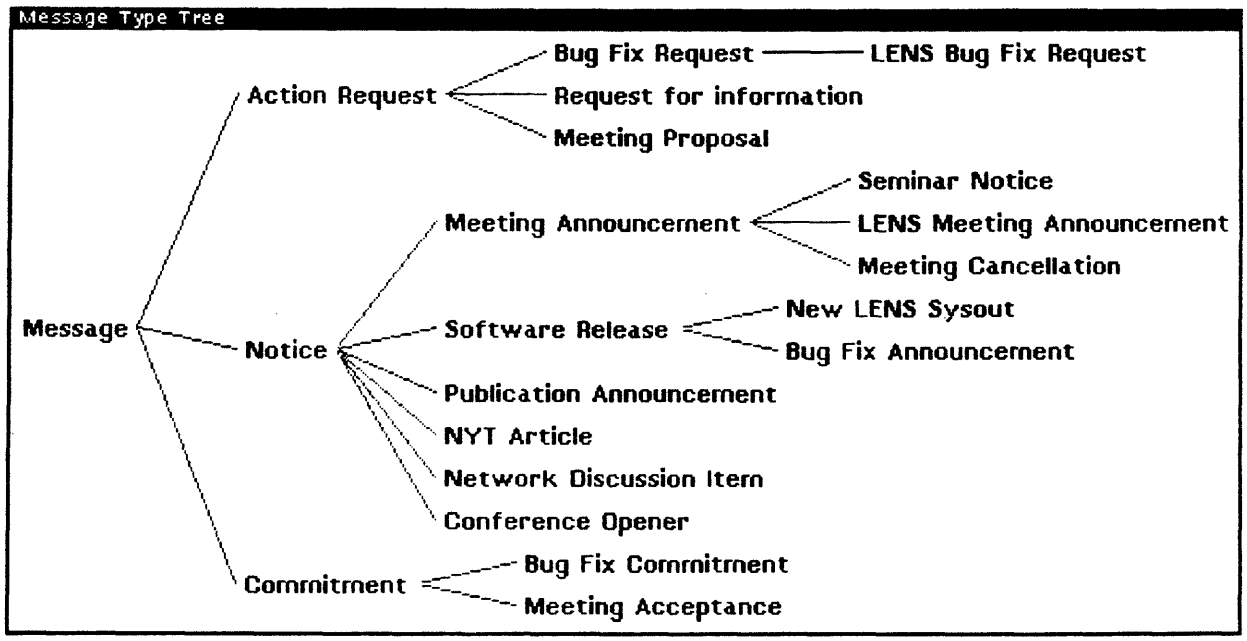


FIGURE 1.2 The message type hierarchy used at MIT

We will refer to this network of message types as a *type hierarchy* even though it need not be a strict hierarchy. The *multiple inheritance* capabilities of the underlying knowledge representation system that we use (Stefik et.al., 1983) allow one message type to inherit properties from more than one parent.

1.2. Communicating with other groups.

In the current version of Lens, all users of Lens in a local group share the same network of message types. Lens users are also connected to national electronic mail networks, however, so they often communicate with other people who do not use Lens at all, and they sometimes communicate with users of Lens at other sites that use a different set of message types. (We currently have one other active test site for Lens, in addition to our research group at MIT.) The problem that arises in this situation is how to handle messages to and from other sites.

Our current solution to this problem is the following: When a Lens message is sent--even within the local group--it is sent as an ordinary message with only the standard mail header fields in its header. All the other fields specific to this message type are sent as the first few lines of what the mail transport system regards as the body of the message. The body of the message also includes a field called "Message Type" and another one called "Text" that contains all the remainder of the message. Thus any electronic mail recipient can read a message from a Lens user. If the message is one of the specialized types with extra fields, these fields (including the field names followed by colons) will simply appear in the body of the message.

When a Lens user receives a message, the message is first checked to see whether it contains a field called Message Type and whether the name in that field is one recognized by the local group. If so, the message is treated as being of that specialized type. If not, the message is treated by Lens as the most general message type of all--Message. Thus Lens users can receive messages from any connected electronic mail user. These messages are not processed by any of the special purpose rules set up for particular kinds of messages (such as Meeting Announcements or Bug Fix Requests). They can, however, be processed using rules that check fields present in all messages (such as To, From, Subject, and Text). Thus the system degrades quite "gracefully," taking advantage of as much of the Lens functionality as possible, even when communicating with non-Lens users.

With the current scheme, Lens users can also receive and process specialized message types from Lens users at other sites--or even from non-Lens users who simply send messages with the appropriate field names typed into the body of the messages. When the sending site and the receiving site have exactly the same definitions for a given message type, this communication works as desired. However, there are two kinds of problems that can occur:

(1) If the sender and the receiver use the same name for what are actually different message types, then the incoming message may have either extra fields, missing

fields, or both. In these cases, the extra fields are inserted in the text of the message and the missing fields are simply treated as being empty. Even if fields with the same names are present, the sender may be using these fields in ways that are different from what the receiver expects.

(2) If the sender and receiver use different names for what are in fact the same message types (e.g., Bug Report and Bug Fix Request), then they can read each other's messages satisfactorily, but none of the automatic processing that could otherwise have been applied to the messages will be invoked.

No disasters occur in either of these cases, but the Lens system fails to be as helpful as it might be. The problem we will explore in the remainder of this paper is how to design systems like Lens that can retain as much functionality as possible in communication between groups without imposing excessive burdens on the different groups to coordinate their changing document type definitions.

2. The Problem

We formulate our problem more precisely as follows:

Let there be some number of *groups* A, B, C, etc. Each group has a set of document *types* that are shared by all members of the group. For instance, the types shared by members of group A might be denoted by a_1, a_2, etc. We refer to the set of document types used by a group as the *language* of the group. We will be particularly concerned with languages that are *type hierarchies*-- that is, languages in which some document types are specializations of others (their "parents") and automatically inherit properties from their parents.

There are also some number of *translation schemes* for translating types from one language to another. For instance, $T_{XY}(x)$ might be a scheme that translates the types used by group X into types used by group Y. The problem is to formulate translation schemes that (1) preserve the original "meaning" of a given type of document as much as possible while (2) allowing different groups to create or change their type definitions with as much autonomy as possible. Let us explore these two objectives in more detail.

2.1. Preservation of Meaning

The question of what a document, or any other expression, "means" is, of course, a complex philosophical and linguistic issue (eg. see Putnam, 1975; Jackendoff, 1983). For our purposes here, we will oversimplify greatly and ignore most of these complexities. Following (Barwise & Perry, 1983), we will not regard the "meaning" of an expression as having some (potentially falsifiable) correspondence with "reality". Instead, we will focus only on the actions that a receiver might wish to take on receiving an "expression" (eg. a document). Thus, for our purposes, two documents that evoke the same actions in the same situation, have the same "meaning". To preserve meaning of an expression, therefore, means to translate the expression in such a way that the receivers of the translated expresion can perform the same range of actions they would have wished to perform had they received and "understood" the original expression. We intentionally leave undefined the term "understood" and appeal only to an intuitive sense of what it would mean for receivers to "understand" the original expression.

In the context of Lens, for example, we can interpret this criterion as follows. Take the simple scheme T_{AB} that translates any type unknown to a group into the most general type the group has, say Message. Suppose someone in Group A sends to someone in Group B a message of type Action Request, and suppose that Group B does not have the message type Action Request. The operations that the members of the group want to apply on objects of the generic type, Message, such as Reply To, Forward, etc. are preserved under this scheme. However, the operations specific to Action Request messages such as Making a Commitment would not be available because, for all the system knows, the object received is now an object of type Message no matter what the original group may have intended it to be.

Now suppose further that group B has a message type called Request and that another translation scheme T'_{AB} allows translation of all objects of type Action Request to this type, Request. This translation scheme would preserve the meaning of the original type to the extent that the operations that the group has defined on Request objects overlap with the operations that the group would have defined over the set of objects that the other group classify as Action Request. If the two sets of operations are the same, then the meaning of the type is fully preserved, despite the differences in how the groups name it. If the two sets of operations do not have much in common, then the meaning of the type is not well-preserved. One can imagine situations where this definition is problematic--e.g., receivers cannot understand the document as its senders did--but is useful enough for our purposes.

One way to ensure full semantic preservation is to require all groups to share the same set of types. However, such a requirement has its costs. The next objective captures the tradeoffs involved.

2.2 Autonomous evolution of group languages

Each group would, in general, like to be able to create or change their type definitions according to their own needs with as few constraints as possible from the other groups. We separate three aspects of this objective: maximizing expressive adequacy, minimizing the need for consensus, and minimizing the need for propagation of changes.

2.2.1. Maximize expressive adequacy. Each group would like to be able to express the distinctions that are useful for its purposes. Requiring all groups to share the same set of message types makes it difficult to meet this objective of expressive adequacy because the different goals and contexts of different groups often lead them to make conflicting distinctions. For example, a manufacturing division of a company might want to make many detailed distinctions in their messages about different parts and subassemblies of many different products while the marketing division of the same company might want to group products in a different way and make detailed distinctions about various market segments.

2.2.2 Minimize need for consensus. Each group would also like to be able to change its type structure while requiring as little consensus as possible from other groups. In general, type structures are not static, but evolve as needs change. The less the need for consensus from other groups, the more easily a group can change its type definitions to meet its current needs.

2.2.3 Minimize need for propagation of changes. When one group makes a change, other groups might have to know about the change so that they can reflect this change in their translation schemes. For example, if one group renames a type, all the groups that maintain a dictionary for translating types from that group would have to know about the change. In general, the fewer of these updates a translation scheme requires, the more desirable it is on this objective.

In the next sections, we will discuss how these different objectives interact when we evaluate different translation schemes against these objectives.

3. The space of possible translation schemes

There are a number of possible translation schemes that achieve, to different degrees, the objectives we discussed above. In this section, we will describe several general solutions to the translation problem, evaluate each solution with respect to the objectives discussed above, and provide specific examples of how each solution could be implemented for the Lens system.

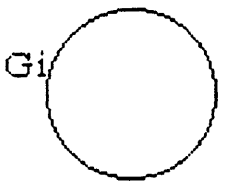
We partition the space of possible translation schemes by using a dimension that centers around the notion of common language and its relation to group language. Even though there are other ways of categorizing possible solutions, we have found this dimension useful in placing the schemes that seem practically important. First we define a *common language* for a set of groups as a language that all the groups in the set use to communicate with one another. In order for all groups to be able to communicate using the common language, each group must either use the common language itself or be able to translate between the common language and the group language.

Given this definition of common language, we characterize the space of possible general solutions to the translation problem in terms of the set theoretic relationships between the group languages and the common language (if any). If we consider only "pure" cases, that is cases where all the group languages have the same relationship to the common language, then, as shown in Figure 3.1, there are six possibilities. The first possibility is that (1) there is *no common language*. If there is a common language, then it may be (2) *identical* with the group languages, (3) *non-overlapping* with the group languages, (4) a *subset* of the group languages, (5) a *superset* of the group languages, or (6) *partially overlapping* with the group languages.

The first four of these possibilities represent interesting practical solutions to our general problem, and we will describe them in the remainder of this section, using the terms (1) *no common language*, (2) *identical group languages*, (3) *external common language*, and (4) *internal common language*. The fifth and sixth possibilities and "hybrid" cases where some group languages have one relationship to the common language and other groups have another relationship can all be analyzed in terms of the first four "pure" possibilities. We discuss these special cases briefly at the end of the section along with several other possibilities.

(a) No Common Language

$C = \phi$

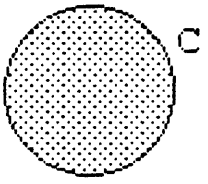


Gi: A Group Language

C: A Common Language

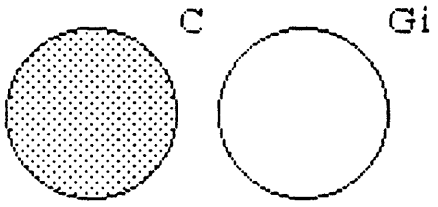
(b) Identical Group Languages

$C = Gi$ for all i.



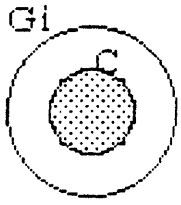
(c) External Common Language

$C \cap Gi = \phi$



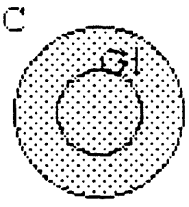
(d) Internal Common Language

$C \subset Gi$



(e) Superset

$C \supset Gi$



(f) Intersection

$C \cap Gi \neq \phi$
$C \neq Gi$
$C \not\subset Gi$
$C \not\supset Gi$

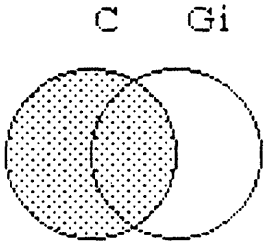


FIGURE 3.1 Possible Relations between Common Language and Group Language.

3.1 No Common Language

In a scheme of this type, there is no common language that is shared by groups that wish to communicate. Instead, each pair of groups must be able to have pairwise translations made into and out of each other's languages. This is, in a sense, the situation that actually prevails in the real world of natural languages such as English, French, and Japanese. In the world of computer-based documents, this scheme implies that there are translating programs for each pair of groups. The most general form of such programs would involve "dictionaries" that provide translation rules for each type from a source language into some type in the target language. Type and map derivations in Heimbigner and McLeod's work (1985) provides an example of such a dictionary in the context of information sharing among autonomous databases.

In Lens, this solution could be implemented by letting each group have a dictionary for each other group with which it communicates. Whenever a message is received from one of these groups, Lens would use the dictionary to translate the incoming message into a type understood by the receiving group. This translation could involve much more than just changing the type name. It might, for example, include moving and transforming values within the fields of the message. For instance, the following rule might be contained in a dictionary for messages that group B receives from Group A:

IF the message type is Information Request
    and the topic field contains Lens,
THEN convert the message type to Lens-related Request,
  and map the fields as follows:
        ChangeDateFormat (Action Deadline) -> By When,
        Subject -> Topic.

The rule says that if the received message is of type Information Request from Group A and the value of its topic field is Lens, then group B will treat it as a message of type Lens-related Request; change the name of the field 'Action Deadline' to 'By When' and change the name of the field 'Subject' to 'Topic'. Furthermore, the function ChangeDateFormat will be applied to the value of Action Deadline in the original message so that it will appear in the format used by Group B, say YY/MM/DD.

Evaluation. This scheme is very flexible in the sense that it can provide case-specific translations in fine-grained details. In this way, the scheme can fully preserve meaning provided that the following condition is met:

If there is a one-to-many mapping from a Group A's type, X, to Group B's types, Yi, then there should be other characteristics of documents of type X that allow unambiguous selection of the Yi to which X should be mapped.

In the above example, the rule that translates Group A's Information Request type to Group B's Lens-related Request is only a partial map because only some of the messages of Information Request type are mapped to the type in B. To be able to translate all the Information Request messages properly, there has to be more than one type in B to which Information Request is mapped. Hence, there has to be some characteristics in such messages that allow the translation scheme to determine as which types in B the messages should be translated. In the example, the value Lens in the Topic field served as such a characteristic.

This scheme also satisfies some of the objectives of autonomous evolution fairly well. Since the group is not constrained to have any part of its hierarchy shared with other groups, the expressive adequacy is not limited. Since setting up translation rules within a group does not affect any other group, no consensus is required. However, when a group changes its type structure, the change will, in general, have to be propogated to all other groups with which this group communicates so that the other groups can change their translation rules or dictionaries. A problem with this scheme, therefore, is that it could be quite costly to set up and maintain. Each group needs a dictionary for all the other groups with which it communicates. If there are n groups communicating with one another, there would, in general, need to be $n(n-1)$ dictionaries. Also, whenever a new group joins, all the other groups would have to set up a dictionary for that group, and the new group would have to set up dictionaries for all the existing groups.

Applicability. This scheme would be appropriate when the groups already use their own langauges and when the efforts to build translation rules and dictionaries for the other groups are justified. Such efforts may be justified, for example, if the number of groups that need to communicate may be small or if the groups themselves are stable and their type hierarchies do not change often.

3.2 Identical Group Languages

In this scheme, the common language is the same as any group language. That is, all groups are required to use the same language and communicate through that language. Examples of this scheme include the adoption of global standards such as Dewey Decimal Classification System among libraries or Computing Reviews Classification Scheme used for categorizing literature in computer

science. In the context of Lens, adopting this scheme means that all the groups share the same message type hierarchy.

Evaluation. In this scheme, meaning is fully preserved because the types and the operations defined on them are the same for all the groups. However, expressive adequacy is quite limited and the need for consensus is quite large. Since all groups have to agree on changes, the only changes that are easy for a group to make will be those that do not affect other groups or those that are valuable to all groups. Even these changes may require a significant overhead to come to agreement. This scheme also requires that all changes be propagated to all groups.

Applicability. This scheme would be useful when there are not already well-established groups and when the groups do not differ greatly in their needs. This may occur, for instance, when the domain is very restricted or artificial, or because there is a well-accepted theory about the domain, or simply because the groups share the same goals and environments insofar as they need to communicate. For example, there may be a small set of generic message types such as Request and Commitment that are useful in almost all office environments (Winograd & Flores, 1986).

3.3 External Common Language

In this scheme, each group uses its own separate language for communication within the group, but there is a single common language for communication between groups. A group communicates with another by first translating its language into the common language, and then having the receiving group translate the common language into its own group language. In this way, each group only needs to know its own language and how to translate it into and out of the common language. An example would be the adoption of an international language such as Esperanto. Another example can be found in the idea of abstract data type that allows data structures to be implemented in any locally convenient ways as long as they maintain consistent interface to the outsiders (Guttag, 1977).

If Lens were to use this scheme, there would be a common type hierarchy shared by all groups in the sense that each group would know how to translate its own types into the types in this hierarchy and vice versa. For example, suppose the common hierarchy consists of the root node, Message, and three specialized message types Action Request, Notice, and Commitment, as in Fig. 3.1. Then when group A sends a message of type Information Request with "Lens" in the topic field, it might first be translated into the type Action Request of the common hierarchy. Then group B might interpret the

message as a Lens-related Request because group B has a translation rule that says that any message of common type Action Request with the value Lens in its topic field should be treated as a message of type Lens-related Request.

The translation can be done in the same way as described above for the No Common Language scheme--that is, by means of a dictionary containing translation rules. The differences between this scheme and No Common Language scheme is that in this scheme all the groups need to know only how to translate back and forth between their own languages and the common language. So each group only needs two dictionaries, one for translating to and the other for translating from the common language; overall only $2n$ dictionaries are needed for n groups.

Evaluation. This scheme can fully preserve meaning provided that the condition discussed in Section 3.1 is met for translations both into and out of the common language-- that is, when there are one-to-many mappings between types, there must be some other characteristics of the documents that enable the unambiguous selection of a target type. However, often this condition is not satisfied. Hence, in general, the more translations a scheme requires, the less it is able to preserve meaning. Since an External Common Language scheme requires more extra translation steps than the others we have considered, we expect that it is less likely to preserve meaning than the others.

Since each group can design its own language without being constrained by others, the group languages can presumably have as much expressive adequacy as desired. The need for consensus applies only to deciding what the common language should be and how it should be modified. If the common language changes, the translations for all the groups have to change, so some propagation of updates is required. However, changes in group languages need not be propagated to other groups.

Applicability. This scheme would be useful when groups that already use different languages want to communicate and when the cost of setting up translation rules or dictionaries for all the groups is too high. Presumably it is for these reasons that people propose an international language or use English as one. Also, this scheme might be useful when there is a language that is expressive enough, but difficult to use for that reason. For example, we can define a language to be the union of all group languages for the cases where such a union is meaningful. Such a language could express all the distinctions that are of interest to any group, but it might be difficult to use if the types are too fine-grained or place too much cognitive load on the user. Such a language, although not suitable as a group language, might serve as the common language.

3.4 Internal Common Language

For schemes in this class, every group has a part of its language that is shared with all the other groups, and the groups use this shared language to communicate with one another. For example, different academic specialists (say, physicists and anthropologists) may each have their own technical vocabularies, but they can usually communicate with specialists from different fields using ordinary English.

A particularly interesting example of this class arises when the languages are type hierarchies. If all the groups share the root node and other types in the top part of the type hierarchy (i.e., they share the most general types), then all the other types that exist in the separate group languages are specializations of types in the common language. This makes it especially easy to translate group types into the common language, as we illustrate below.

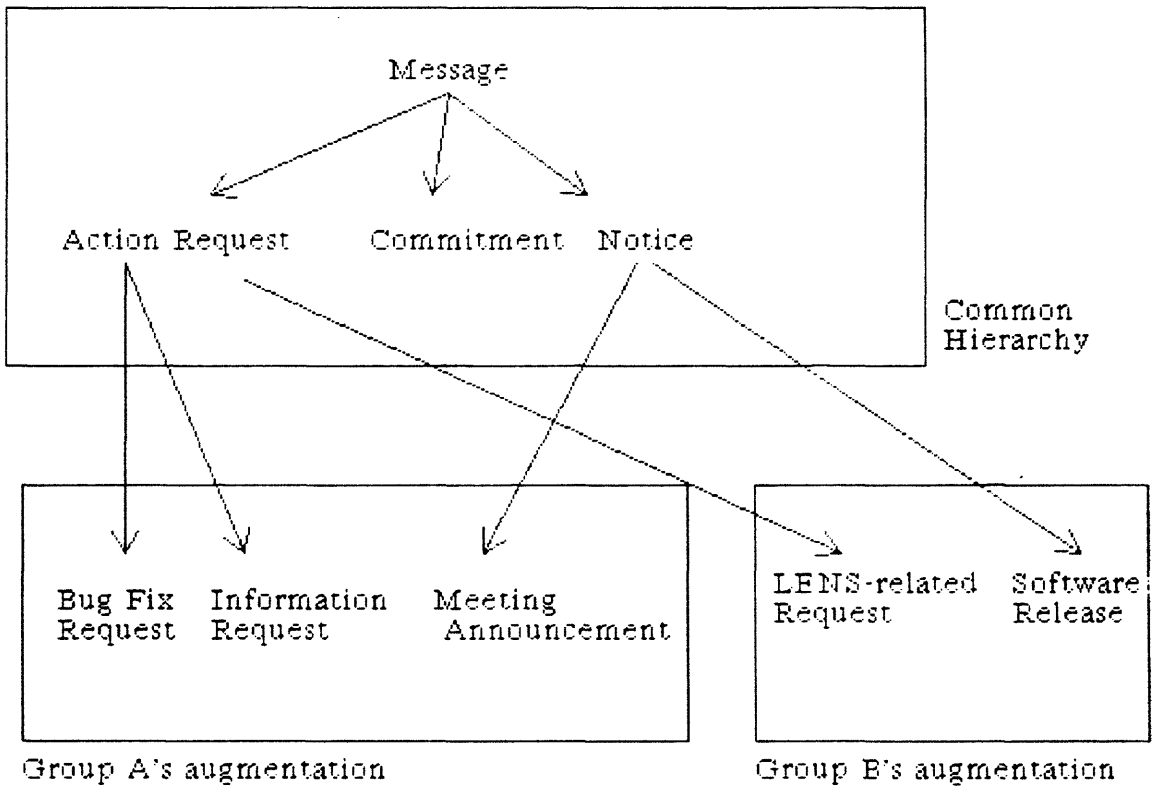


FIGURE 3.2 A common hierarchy and its local augmentations.

We have explored a scheme of this type for Lens called the Local Augmentation Scheme. For instance, all groups might share the message types that are useful almost everywhere, such as Message, Action Request, Commitment, and Notice. Then each group is allowed to locally augment the hierarchy by creating subtypes of these shared types (See Fig. 3.2). When group A sends to group B a message of type, say, Meeting Announcement, the type is automatically translated into the nearest ancestor in the common hierarchy, in this case Notice. Those fields of the type Meeting Announcement (such as Topic) that are also in the nearest common ancestor, Notice, are preserved as they are; those that are not in the ancestor (such as Meeting Time and Meeting Place) are treated as a part of the main text and put in the beginning of the Text field. Of course, if group A sends a message of a type that is already in the common hierarchy, then no translation is necessary.

One way of implementing this scheme is to have each message include its type in the message type hierarchy used by the sending group. If this type is not in the common hierarchy shared by all groups, the message should also contain: (a) the name of the sending group and (b) the nearest "ancestor" of the message type in the common hierarchy. In this way, receivers in the sending group can process the message as being of exactly the type intended by the sender, and receivers in all other groups can process it as being of the nearest "ancestor" type in the common language.

Evaluation. This scheme preserves meaning to the extent that the type structure is shared. If the message is of a type in the common hierarchy, then its meaning is fully preserved. If not, then its meaning will be abstracted away to the degree that its ancestor in the common hierarchy preserves its properties. This scheme allows a great latitude in expressive adequacy since each group can add as many types as desired to express distinctions that are important to them. The only constraint on expressive adequacy is that groups cannot completely ignore distinctions made in the common hierarchy. For example, suppose, as before, that the common hierarchy defines speech act categories such as Action Request or Commitment as the immediate children of the root type, Message. If a group does not want to use this speech-act based categorization, it can add other children of Message and use them for internal communication, but the group members may still receive messages from other groups based on the speech act categorization.

This scheme only requires consensus among groups on what the common hierarchy should be, and it only requires the propagation of changes in this common hierarchy. The changes made locally within a group need not be propagated to other groups. However, a change in the common hierarchy can have far-reaching implications. All the groups that have rules or subtypes of the modified type would have to make necessary adjustments.

Applicability. This scheme would be useful when (1) there do not already exist numerous well-established and incompatible language groups, (2) there are important commonalities across groups, and (3) there are also significant local variations among groups. For instance, it seems quite useful in the Lens environment to have a set of common message types used by all groups (such as Action Request, Commitment, and Meeting Announcement), and to also let each group develop their own specialized message types for their own unique situations. The X.400 standards for electronic messaging are compatible with this approach since they include a field for specifying a message type, and leave open the possibility of different groups developing their own conventions about how this field is used.

This approach also seems quite applicable in many other document interchange situations where the definition of unchanging universal standards is difficult. For example, much recent attention has been focused on developing Electronic Data Interchange standards for a variety of business forms (such as invoices, purchase orders, etc.). The general approach suggested here would be to develop generic versions of these forms that contained the fields and values used in almost all industries as well as one or more additional "miscellaneous" fields (like the Text field in Lens). Then all the companies are ensured a certain level of interoperability, i.e. to the extent that they share the generic version. At the same time, some of the companies can locally augment this generic version to accomodate their special needs by establishing conventions about how to use the miscellaneous fields. Also, in this scheme, it would be possible to develop packaged software for EDI applications that included "hooks" for industry-specific and firm-specific routines.

A similar approach might be used for interchanging formatted documents between different word processing systems. Current approaches to this problem usually involve writing many pairwise translation programs to convert between different formats, or trying to agree on a single standard that is expressive enough to represent all other formats. The approach described here would suggest agreeing on a generic standard for the most basic kinds of documents and formatting, and then having additional miscellaneous "fields" whose interpretation could be standardized by different subgroups. Then, when group B's system reads a document from group A, it can display the standard information in its usual way and merely use indications such as "The original document contained a figure here" when it encounters non-standard formats.

3.5 Other translation schemes

Other set theoretic possibilities

There are two other set theoretic possibilities listed above for the relationship between the common language and the group languages: (1) the common language is a superset of all the group languages, and (2) the common language partly overlaps with all the group languages. (See Fig. 3.1 e & f). The first case amounts to assuming that each group is able to translate any other group's language into its own. For example, a group can always send a message in its own language and have all the other groups translate it into their own. Thus, this case is similar to the No Common Language scheme. The difference is that in this case, a group needs to know how to translate not only the languages with which it communicates but all the group languages because they are all part of the common language. In practice, there is probably no reason for such extra efforts. So we dismiss this case as a theoretical possiblity but a practically implausible one. The second case, where the common language partly overlaps with all of the group languages, amounts to assuming that the common language consists of a part of every group language as well as some language external to all the group languages. Thus, it can be analyzed as a combination of the Internal Common Language and the External Common Language cases.

It is also possible to have hybrid cases, where some groups have one relationship with the common language and other groups have another relationship with the common language. For example, some groups might use only the common language within their own groups (as in the Identical Group Languages case), while other groups augment this language with their own local language (as in the Internal Common Language case), and still other groups might use their own local languages within the groups but have translation tools into and out of the common language (as in the External Common Language case). We believe that all these hybrid cases can be analyzed as combinations of the "pure" cases discussed above.

Sequential composition of translation schemes

The schemes discussed above can be used sequentially in various ways to create even more complex and powerful schemes. For example, Fig. 3.3 shows an example of a group that uses language G1 communicating with another group that uses G4. Its message is first translated to G2 using an Internal Common Language scheme, then into G3 using an External Common Language, and then finally into G4 using No Common Language scheme . Such complicated sequential translations may, of course, "lose something in the translation."

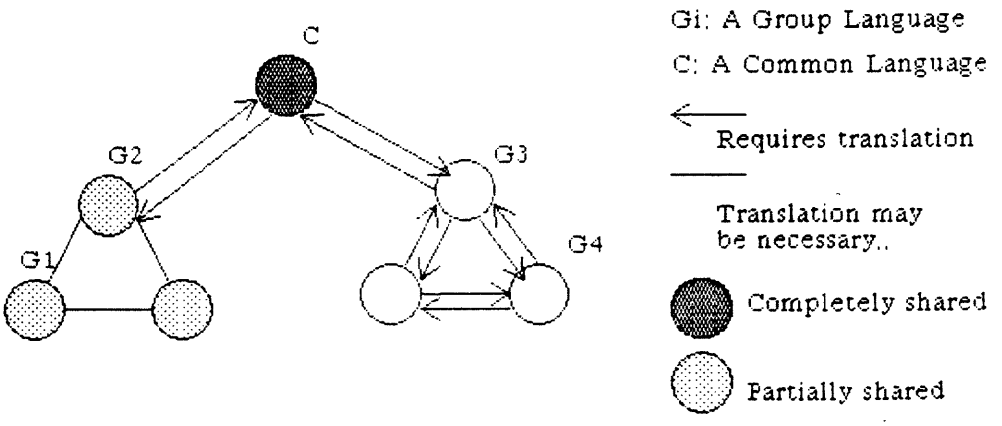


FIGURE 3.3 A sequential composition of translation schemes

Schemes exploiting an implicit type hierarchy

So far, we have categorized translation schemes based on the relationship between the common language and the group languages. This dimension does not make any assumption about the kind of language used. However, when the languages are restricted in some ways, we can have schemes that exploit the special properties of the languages. For instance, we discussed above how the Local Augmentation Scheme exploits the inheritance property of type hierarchies. Here, we describe another scheme that exploits the subsumption property of some type hierarchies (Brachman & Schmolze, 1985).

We define a type hierarchy to be *explicit* if the user creates a type by declaring it explicitly and specifying what its parents and children are. On the other hand, we say that a type hierarchy is *implicit* if a type is defined strictly on the basis of certain attributes of its instances, that is if there can be an automatic classifier that can determine the type of any object given its attributes (Schmolze & Lipkis, 1983). A scheme that exploits an implicit type hierarchy would work as follows in Lens:

Suppose the type of a message type is determined by the fields the message has. For instance, any message that has only the following fields, To, From, CC, Subject, In-Reply-To, Topic, Text is of type Message. Any message that has additional fields is an instance of a subtype of Message. For example, any message that has the additional field, Action Deadline, is classified as being of type Action Request, which is a subtype of Message. Suppose that all the groups agree on the set of fields that define message types. Then any group can send a message to another group without having to

explcitly agree on the types of messages used, because the types can be automatically determined by the fields the messages have. Expressive adequacy is uncompromised because a group, or in fact any individual, can create a new message type whenever needed by composing a message with a new combination of fields. Consensus is not required on what message types to use, but it is required on what fields to use and how to change them when needed.

Although this scheme is attractive in many respects, it would work only if certain assumptions are satisfied in addition to the implicit type hierarchy assumption we made. For example, the type hierarchy should allow multiple parents, because the classification algorithm based on fields would not determine a unique parent. Also, this scheme does not allow two different types to have the same set of fields. Yet there are cases where this assumption is too restrictive. We will not pursue these questions any further here. The point of mentioning this scheme is that often specialized schemes can be designed that exploit particular features of the language used. Nevertheless, the general set-theoretic framework we provided above still holds for these cases as well. For example, depending on whether the set of fields shared are identical, internal, or external, we can have different versions of this scheme with the different trandeoffs in the objectives as we have discussed above.

3.6 Summary of translation schemes

Table 3.1 summarizes the different classes of translation schemes we have considered. The detailed discussions above are summarized in the table by rough qualitative rankings of the schemes on each objective. In each column, the schemes that in general satisfy the objective better receive more stars. For example, the schemes that receive *more* stars in the column, Need for Consensus, have *less* need for consensus.

As we have seen above, there are complex tradeoffs among the objectives we are trying to achieve. For example, as highlighted in the table, the pairwise translation scheme used when there is no common language allows full expressive adequacy within groups and little need for consensus, but it leads to more difficulty in preserving meaning across groups. Using Identical Group Languages has the opposite virtue: It preserves meaning quite well across groups, but it severely restricts the ability of individual groups to adequately express their own distinctions without consensus from other groups. The Internal Common Language scheme is intermediate on all the dimensions, suggesting that it may be a reasonable compromise in situations where no single objective is most important.

TABLE 3.1 Evaluation of the different classes of translation schemes.

	Meaning Preservation	Expressive Adequacy	Need for Consensus	Need for Propagation of Change
No Common Language	**	****	****	*
Identical Group Languages	****	*	*	**
External Common Language	*	***	***	***
Internal Common Language	***	**	**	***

We have shown how the classes of translation schemes we have discussed exhaustively partition the space of possible solutions. However, the examples of these schemes that we considered are by no means the only possible ones. For instance, there may be other schemes that exploit special properties of particular languages just as the Local Augmentation Scheme exploits the inheritance property of type hierarchies.

4. A composite scheme: Partially Shared Views

In this section, we present a composite scheme that combines some of the best features of the basic schemes discussed above. This scheme exploits the notion of *views* to modularize related type defintions and make relations among such modules explicit. The notion of views used here is similar to viewpoints in OMEGA or layers in PIE (Barber, 1982; Goldstein & Bobrow, 1981). For our purpose, a *view* is a set of message types and their relations. A view V_2 is a child of V_1 if some of the message types in V_2 are specializations of ("children" of) message types in V_1. We first illustrate this scheme in the context of Lens and then present its algorithm. We then discuss how well it meets the objectives we listed above.

4.1 Illustration

Consider the following scenario. Suppose the initial view V_0 consists of only the single message type, Message. All the groups can be assumed to share this view because any message can be treated as an instance of the generic type Message. Suppose Group A creates a new view, V_1, containing three specialized types of Message: Action Request, Notice, and Commitment. In other words, group A declares a new view V_1, makes V_0 its parent view, and within the new view creates the three subclasses of the Message type. Now Group A sends a message of type Action Request to group B. Since V_0 is the only view that B has adopted so far, the type of the message is unknown to B.

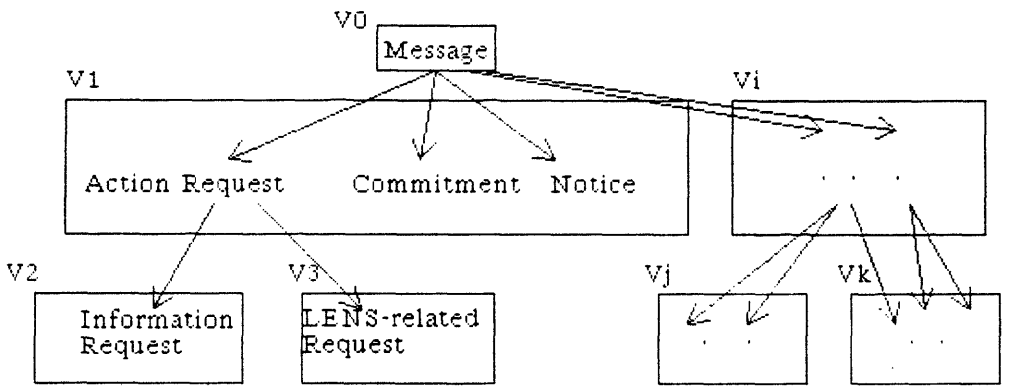


FIGURE 4.1 Example of Views

When a message of unknown type is received, Lens treats it as an instance of the most specific ancestor type that both the sender and the receiver know about. This information can be obtained from the message itself as we will see below. In this case, V_0 is the most specific view that both groups share, and the most specific ancestor type is still Message. Hence, the only operations that Lens is able to apply to the message are those that apply to the type Message. If a group receives messages from a view it does not share, the group may wish to (a) adopt the whole view, or (b) create translation rules from the type in the unshared view to a type within its own views.

Suppose that B adopts V_1 in this case. Then Group A and B share the same view, V_1. So the type T is known to both groups now and no translation is necessary. The scheme becomes an instance of the Identical Group Languages scheme as far as groups A and B are concerned. Now suppose B wants to create a type called Information Request as a child of Action Request. So B creates a view of its own, V_2, that has V_1 as its parent.

Suppose B sends a message of this newly created type, Information Request, to Group C. Group C likes V_1 and adopts it. But C does not like V_2 because it classifies Action Request messages by the kind of object that the request is about (eg. Information Request). Instead, C wants to classify Action Request messages by the subject domain of the requests. So C creates a view V3, with V1 as its parent, in which there is a subclass of Action Request, Lens-related Request. C also sets up the following translation rule:

If the message is of type Information Request in view V2,
and its topic field has as its value Lens,
then change its type to Lens-related Request in view V3.

If the appropriate value is in the topic field of the incoming message, this rule will fire and the incoming message will be translated into a Lens-related Request.

4.2 Algorithm

First, we make a few assumptions:

1. All groups share the default view, V_0, which consists of the most general type, Message.
2. When a group adopts a view, it also adopts all the views which are its ancestors.
3. Each message includes the following information:
    a. the type of the message, t.
    b. a globally unique name of the view within which this type is defined, v.
    c. $\{(t_1, v_2), .., (t_n, v_n)\}$, where v_i is the name of an ancestor view of v ordered from most specific to more general, and t_i is the most specific ancestor type of t within v_i.
    This information is actually a bit more complex than presented here because multiple parents are possible. In these cases, the information has to be presented as a nested list and one to make sure that we deal with multiple parents appropriately. But for simplicity, we assume below that there is only a single parent. For example, in the view structure shown in Fig 4.1, the message type information for type Information Request would consist of the following sequence of (type, view) pairs: {(Information Request, V2), (Action Request, V_1), (Message, V_0)}.

With these assumptions, the algorithm is the following.

Given a message M from Group G,
let t be the type of M;
let v be the view to which t belongs;

if Adopted?(v), **then** process M as is, i.e. as of type t in view v.
;Adopted?(v) returns TRUE if v is one of the views that has been adopted, FALSE if not.
;If the view v has been adopted, then there is no need to translate because we know what it is.
else

if TranslationRule?((t,v)), **then** apply the translation rules and process the message as of the type to which it has been translated.
;TranslateRule?((t,v)) returns TRUE if there is a translation rule for the type t in view v,
;FALSE if not.
;If the view v has not been adopted, then first check if a translation rule has been set up for
;this type.
;If there are more than one translation rules, then make multiple copies of the message, M,
;and apply each translation rule to a copy.
else

From the most specific (t_i, v_i) in the inheritance path, **if** Adopted?(v_i), **then** stop searching and process M as of type t_i **else** check the next general (t_i,v_i).
;So the view has not been adopted and there is no translation rule for the type t.
;So in this case, process the message as its nearest ancestor in the most specific view which is an ancestor of v as well as which has been adopted.
;This algorithm is guaranteed to terminate because of the assumption 1.

;After the message has been processed and presented to the user, an option is availabe for the user to examine the view of the message, adopt the view, or set up translation rules for the type of the message.

4.3 Evaluation

This view-based scheme is a composite of several of the basic schemes discussed in the previous section. As noted above, between groups that share the same view, the scheme is an instance of Identical Group Language scheme. To the extent that the groups have translation rules between views, it serves as an instance of No Common Language scheme. When the groups share only some views and do not yet have translation rules set up between the views not shared, the scheme acts as an instance of Internal Common Language scheme. The scheme thereby allows finer-grained adoption of translation methods among different groups while still providing a unifying framework.

As the scheme is a composite one, its evaluation with respect to the objectives listed in Section 2 depends much on which of the basic methods is adopted between two groups. Semantic preservation will be best if the type belongs to the shared view. If it does not, then its semantics will be fully preserved if the translations can be set up between the types in the two views. Otherwise, some of the semantics will be lost between two views without translation rules. But the semantics will be

preserved to the extent that the two views have the same ancestor view. Expressive adequacy might have to be compromised if a group wants to share the same view with another. But a benefit of this scheme is that it allows a group to weigh the importance of the different objectives for different cases and select a translation method appropriate for each case. For example, if expressive adequacy and semantic preservation are both important enough to justify the efforts, then it can create its own view and set up translation rules between the two views. If, on the other hand, semantic preservation is not important enough to justify the efforts, the group can create its own view but may not set up translation rules, in which case the default translation of the type into the nearest ancestor in the shared view will be used.

No consensus between groups is needed in this scheme because no group is required to adopt any other group's views. If a group chooses not to adopt an existing view V, the group can simply create a new view V' as another child of V's parent. The inheritance structure of the viewbase (i.e., the collection of views) is useful because it allows groups to share agreements about types at one level and disagree about types at another level.

Another useful property of this approach is that the viewbase is monotonic, that is, existing views are never modified but only new views are added. This means that other groups can continue to use old views even when a group adopts a new view, and therefore communication can continue without propagating any updates. However, in order to achieve greater preservation of meaning, other groups may wish to know about new views so that they can adopt them or create translation rules for them. In practice, the information about new views could be distributed in a variety of ways such as: (a) posted in a global database that is accessible to all the groups, (b) broadcast to all the other groups, or (c) distributed in physical documentation such as newsletters.

4.4 Applicability

Since the Partially Shared Views scheme combines many of the best features of the other schemes it is potentially relevant in very many situations. In a sense, each of the other schemes is a special case of this one. This scheme, therefore, may require somewhat greater implementation complexity, but it allows a great flexibility in using a combination of different translation schemes for different situations.

For instance, this approach seems clearly desirable for the Lens system and we are currently implementing a version of it there. For similar reasons, the approach also seems desirable for many other office systems. We described above in section 3.4 how the Local Augmentation scheme could let

different EDI or document preparation systems achieve a certain degree of interoperability by agreeing on generic global standards that left room for augmentation by different subgroups. The Partially Shared Views scheme seems even more appropriate in these situations because it allows many overlapping subgroups to use their own standards for communication with members of the same subgroups, and also to set up translation rules for communication with other subgroups, even when there are no significant global standards.

5. Conclusion

We began this paper with a description of the problem of translating between different document types (such as messages and forms) in various kinds of office systems. At that level, our analysis suggested several immediately useful extensions to the Information Lens system and several intriguing possibilities for other office systems such as forms processing systems, document preparation systems, and Electronic Data Interchange systems.

Most of our analysis of this problem, however, was conducted at an abstract level that is much more general than just documents and office systems. In most cases, our analysis seems to apply to languages in general and to communication among any kind of agents, including both human and computational agents. It is intriguing to speculate about the implications this line of research might have for some of the issues that arise in more general contexts. For example, schemes for translating between different versions of objects in an object oriented database (e.g., Skarra & Zdonik, 1986) can be seen as translation schemes of the sort we have analyzed here. The space of solutions we have outlined may suggest new type management schemes for object-oriented databases, or existing type management schemes may suggest new translation schemes for office systems. Other instances of this general translation problem occur in communication between computational agents in a "blackboard architecture" (Nii, 1986) and in communication between heterogeneous databases in a networked environment (Heimbigner & McLeod, 1985). We suspect that the approach we have outlined here may eventually help illuminate and show connections among these other applications, in addition to the office system applications we have explored here.

References

Barber, G. 1982. Office Semantics.. Ph.D. thesis. Massachusetts Institute of Technology, 1982.

Barwise, J. & Perry, J. 1983. *Situations and Attitudes* MIT Press. Cambridge, MA.

Brachman, R.J. & Schmolze, J.G. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2). April-June. 171-216

Fikes, R. & Kehler, T. 1985. The role of frame-based representation in reasoning. In *Comm. ACM* v.28(9) September. pp.904-920

Goldstein, I.P. & Bobrow, D.G. 1981. Layered networks as a tool for software development. *Proc. 7th Int'l Conf. on Artificial Intelligence*, 1981.

Guttag, J.V. 1977. Abstract data types and the development of data structures. *Comm. ACM*, v.20(6). June 1977, pp.396-404

Jackendoff, R. 1983. *Semantics and Cognition* MIT Press. Cambridge,MA.

Heimbigner, D. & McLeod, D. 1985. A federated architecture for information management. In *ACM Trans. Office Information Systems* 3(3) July, 1985.

Malone, T.W., Grant, K.R., Turbak, F.A., Brobst, S.A., Cohen, M. 1987a. Intelligent information-sharing systems In *Comm. ACM* v.30(5) May. pp.390-402

Malone, T.W., Grant, K.R., Lai, K.Y., Rao, R., & Rosenblitt, D. 1987b. Semi-structured messages are surprisingly useful for computer supported coordination. In *Trans. on Office Information System* v.5(2) pp.115-131

Nii, P. 1986 The blackboard model of problem solving. *The AI Magazine.* Spring, 1986 pp.38-53

Putnam, H. 1975 The meaning of 'meaning'. In *Language, Mind, and Knowledge*, ed. by K. Gunderson. Minneapolis: Univ. of Minnesota Press.

Schmolze, J.G. & Lipkis, T. 1983. Classification in the KL-ONE knowledge representation system. *Proc. 6th Int'l Joint Conf. on Artificial Intelligence*, 1983.

Skarra, A.H. & Zdonick, S.B. 1986. The management of changing types in an object-oriented database. In *Proc. OOPSLA*, September 1986. pp. 483-495.

Stefik, M., Bobrow, D.G., Mittal, S. & Conway, L. 1983. Knowledge programming in LOOPS: Report on an experimental course. *The AI Magazine*, pp.3-13, Fall 1983

Tsichritzis, D. 1982. Form management. In *Comm. ACM* v.25(7) July pp.453-478

Winograd, T. & Flores, F. 1986. *Understanding Computers and Cognition: A New Foundation for Design.* Ablex, Norwood, N.J.