National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

**A TEMPORAL REPRESENTATION
FOR
MULTIMEDIA RADIOLOGICAL
REPORTS**

by

CHERYL SCHRAMM, B. ENG

A Thesis submitted to
the Faculty of Graduate Studies and Research
in partial fulfillment of
the requirements for the degree of

Master of Engineering

Department of Systems and Computer Engineering

Carleton University
Ottawa, Ontario
© April 1989

**National Library
of Canada**

**Bibliothèque nationale
du Canada**

Canadian Theses Service Service des thèses canadiennes

**Ottawa, Canada
K1A 0N4**

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-54375-2

ACCEPTANCE SHEET

The undersigned recommend to the Faculty of Graduate Studies
and Research acceptance of the thesis

A Temporal Representation
for
Multimedia Radiological Reports

submitted by Cheryl Schramm, B.Eng.

in partial fulfilment of the requirements for
the degree of Master of Engineering

Thesis Supervisor

Chair, Department of Systems and Computer Engineering

Carleton University
April 1989

ABSTRACT

Multimedia radiological reports tie together radiographic images, image processing operations on these images, and text or voice annotation to produce an animated radiological examination report. A representation for a multimedia radiological report is introduced which describes the temporal behaviour of the animated report. The temporal behaviour specifies the synchronization requirements for the coordinated presentation of the multimedia report components. The temporal behaviour is encoded in a temporal semantic network which is a network of temporally ordered nodes, where each node represents some primitive behaviour. Automatic creation of the multimedia radiological report consists of the capture, interpretation and storage of the temporal semantics of the radiologist's reporting session in terms of a set of defined primitive behaviours. Automatic playback of the multimedia radiological report is the orderly traversal of the temporal semantic network, invoking the behaviour associated with each of the nodes in the temporal order described by the interconnecting arcs.

ACKNOWLEDGEMENTS

I would like to thank both my supervisors for their instruction, guidance, and support and unique perspectives. By always encouraging me to read more, Professor Bernie Pagurek of Carleton University helped me gain more of an education than that strictly within the confines of this thesis. Professor Morris Goldberg of the University of Ottawa offered me his expertise both in the medical communications area and in engineering research. To both, I extend my thanks for the extra effort required to cooperate in a joint supervision. I would also like to thank Gerald Karam of Carleton University for assuming the role of mentor and friend. His enthusiasm and advice were always useful.

I would like to credit the Ottawa-Carleton Research Institute for Electrical Engineering for allowing me, a Carleton University student, to participate in a project at the University of Ottawa. I would also like to acknowledge the financial support afforded to me by the National Science and Engineering Research Council.

Finally, I would like to say thanks to my husband, Steve Janssen. His support encouraged me to pursue my studies, and his help in the production of this thesis was invaluable.

TABLE OF CONTENTS

	PAGE
ACCEPTANCE SHEET	ii
ABSTRACT	iii
ACKNOWLEDGEMENTS	iv
TABLE OF CONTENTS	v
LIST OF TABLES	vii
LIST OF FIGURES	viii
GLOSSARY OF TERMS	x
LIST OF ACRONYMS	xii
 1 INTRODUCTION	 1
2 A LITERATURE SURVEY OF MULTIMEDIA DOCUMENTS	4
2.1 Multimedia Document Models	4
2.2 Multimedia Facilities	15
3 A SELECTION OF KNOWLEDGE REPRESENTATIONS	20
3.1 Semantic Networks	21
3.2 Frames	23
3.3 Production Systems	28
4 TEMPORAL LOGIC	29
4.1 Interval Based Temporal Logic	30
5 THE MUSIC RADIOLOGY MULTIMEDIA COMMUNICATION SYSTEM	42

	PAGE
6 A TEMPORAL REPRESENTATION OF A MULTIMEDIA RADIOLOGICAL REPORT	47
6.1 The Temporal Behaviour of a Multimedia Radiological Report	48
6.1.1 The Temporal Characteristics of Visual Activities	49
6.1.2 The Temporal Characteristics of Audio Activities	58
6.1.3 The Temporal Interactions Between Activities	62
6.1.4 Summary of the Temporal Behaviour Analysis	69
6.2 A MRR Temporal Semantic Network	71
6.2.1 The Node Semantics	72
6.2.2 The Arc Semantics	78
6.2.3 The Network Semantics	85
6.3 Sample Multimedia Radiological Reports	88
6.3.1 The MRR TSN for Voice Playback with a Fixed Display	88
6.3.2 The MRR TSN for Voice Playback with a Changing Display	91
6.3.3 A MRR TSN for Contextual Segmentation and Don't Care Situations	93
6.3.4 A MRR TSN for an Image Sequence	95
6.3.5 A MRR TSN for Multiple Screens	97
7 AN AUTOMATIC MULTIMEDIA RADIOLOGICAL REPORT FACILITY	96
7.1 The Automatic Playback Process	97
7.1.1 A Design for the Playback Process	105
7.1.2 Testing of the Playback Process	116
7.2 The Creation Process	121
7.2.1 A Design for the Creation Process	122
8 CONCLUSIONS AND FUTURE RESEARCH	125
APPENDIX A CODE FOR THE PLAYBACK PROCESS	135
REFERENCES	149

LIST OF TABLES

TABLE	DESCRIPTION	PAGE
Table 4.1	The Primitive Temporal Relations	33
Table 6.1	Nodes for a MRR Temporal Semantic Network	71
Table 6.2	Temporal Predicates for a MRR Temporal Semantic Network	82

LIST OF FIGURES

FIGURE	DESCRIPTION	PAGE
Figure 2-1	The Office Document Architecture Model	7
Figure 2-2	The ODA model for a Simple Letter Document	9
Figure 2-3	The MMCP Representation of a Multimedia Document	12
Figure 2-4	The Tree Structured Representation of a Multimedia Document Used by CCWS.	15
Figure 3-1	Graphics Notation for Semantic Networks	21
Figure 3-2	A Simple Semantic Network	22
Figure 3-3	The Basic Structure for a Frame.	24
Figure 3-4	A Frame Describing a Typical Car	24
Figure 3-5	A Sample Frame Hierarchy - A Horse Taxonomy	24
Figure 4-1	Time as a Continuous Infinite Axis	30
Figure 4-2	A Time Interval	31
Figure 4-3	Timelines for a Sample Scenario	32
Figure 4-4	Timelines for the precedes Defined Relation	34
Figure 4-5	An Algorithm for Maintaining Temporal Information in a Fully Connected Network	36
Figure 4-6	Maintaining Temporal Knowledge Using Constraint Propagation	38
Figure 5-1	The MUSIC Testbed System	42
Figure 6-1	A Sample Screen Display Consisting of an Image Display, A Zoom and a Marker.	49
Figure 6-2	A Timeline For a Screen Update During a Voice Playback.	63
Figure 6-3	A Timeline For an Image Sequence	66
Figure 6-4	Coupling of Visual Actions with DISPLAY Nodes.	73
Figure 6-5	A Node Frame Hierarchy	74
Figure 6-6	The Timeline for All Visual Activities	76
Figure 6-7	The Possible Temporal Relations Between a PLAY and DISPLAY Nodes.	78
Figure 6-8	The Logical Organization of a Temporal Semantic Network	83
Figure 6-9	A MRR TSN for Voice Playback with a Fixed Screen Display.	86
Figure 6-10	The MRR TSN For Voice Playback with a Changing Screen Display.	89
Figure 6-11	A MRR TSN For Contextual Segmentation and Don't Care Situations.	91
Figure 6-12	A MRR TSN For an Image Sequence.	93
Figure 6-13	A MRR TSN For Voice Playback with Multiple Screens	95

FIGURE	DESCRIPTION	PAGE
Figure 7-1	An Input/Output Model of an Automatic MRR Facility	97
Figure 7-2	Two Sample MRR TSNs	99
Figure 7-3	A Simulation of the MUSIC Workstation Using Windows.	105
Figure 7-4	The Software Structure Diagram for the MUSIC Testbed Simulation	106
Figure 7-5	The Playback Modes	107
Figure 7-6	An Algorithm for Sequential Execution	109
Figure 7-7	Sample Rules Defining "Synchronized" Image Screens	112
Figure 7-8	Testing of the Defined Predicate starts_during.	116
Figure 7-9	Command Script Files for a Network Fragment	118
Figure 7-10	The Creation Process as a Two-Step Procedure.	120
Figure 7-11	A Frame Hierarchy for the Successive Refinement Classifications of MRR Situations.	123
Figure 8-1	A Logical Layout for the Multimedia Patient File	133

GLOSSARY OF TERMS

action	An active behaviour which self-terminates within a finite period of time.
activity	A component of the temporal behaviour, made up of either one or two behaviours, which is associated with one function and occurs over some time interval, as seen by the user.
behaviour	A primitive or indivisible component of temporal behaviour.
contextual segmentation	The notion that a single physical voice segment is made up of two or more logical voice segments, each one of which has its own context.
ephemeral activity	An activity which has only an action; it has no persisting effect
defined relations	A temporal relation formed from the logical OR of two or more primitive temporal relations, used to express uncertainty.
don't care situation	A case where the temporal interaction between a voice segment and a screen update is not important because the voice segment is independent of the screen display.
effect	A passive behaviour which persists indefinitely until terminated by some other action.
examination report	A report, made for each x-ray examination, which records the reviewing radiologist's impressions of the examination's radiographs.
external activity	A user-defined activity, invented to describe some complex temporal interaction.
global display	The collective display of several image screens
image sequence	A group of image displays functionally presented as a group, but because of limited available image screens, presented as a sequence.

interval based temporal logic	A logic for the description and reasoning about time, in which the basic unit is an interval of time.
layout structure	The organization of the content of a document according to its physical presentation.
logical structure	The organization of the content of a document according to the functional structure (eg. chapters, paragraphs, figures).
multimedia document	A document composed of inter-related segments of image, voice, text, data, etc.
multimedia radiological report	A multimedia version of the examination report in which the report is tied to the associated radiographic images.
persistent activity	An activity which has both an action and a persistent effect.
presentation descriptors	Recursively defined as the temporal relation between a group of lower descriptors, with the lowest descriptor being mapped to some portion of a document's content.
temporal behaviour	The change in conduct or operation of a device or system over time, as seen by the user.
temporal representation	A description based on changes over time.
temporal semantic network	A collection of temporally ordered nodes, where the nodes represent time intervals to which activities can be associated.
terminating activity	An activity whose action terminates the effect of some previous activity.
zoom image	A full resolution (2024x2024) digital image of a radiograph

LIST OF ACRONYMS

CCITT	International Telegraph and Telephone Consultative Committee
ECMA	European Computer Manufacturers Association
ISO	International Organization for Standardization
MMCP	Multimedia Message Content Protocol
MRR	Multimedia Radiological Report
MUSIC	Multimedia System for Integrated Communications
ODA	Office Document Architecture
TSN	Temporal Semantic Network
WS	Workstation

Chapter 1 INTRODUCTION

Radiography is a diagnostic imaging procedure. One of the main duties of a radiologist is to read the radiographs and to report his findings. The *examination report* is normally in textual form but invariably must indirectly reference anatomical features on the radiographs. Direct references are made in face-to-face consultations with physicians or by drawing on tracing paper placed over the radiograph.

There is now a growing interest in so-called "Digital Radiology Departments" where the radiographs are stored digitally [Huang87]. Digital images offer easier distribution and the potential of image enhancement operations. The computer-based workstations and integrated voice and data communication systems used in these digital radiology departments also make it feasible to directly link the radiographic images with text or voice connections. The result is what we call a multimedia radiological report.

A multimedia radiological report (MRR) is not simply a static collection of multimedia data - the examination report and film images. Instead, the multimedia radiological report is a dynamic animated entity which must tie together the set of

radiographic images, operations performed on these images (displays, enhancements, etc.) and textual, audio or graphical annotations. In other words, the important information in a MRR is the animation requirements - how the composite data are used to prepare a synchronized presentation of the examination report.

The conceptual model that is used to represent the MRR must be able to accomodate a variety of features and performance requirements. The representation must include image processing and manipulation operations as an integral part of the MRR. It must accomodate an automatic MRR facility for the unobtrusive creation and playback of MRRs, although allowing user interaction - such as pausing or rewinding - during the automatic playback. Finally, the representation shall provide for variations in workstation performance due to variable delays in disk retrieval and network transfers in the multimedia communications system.

In this thesis, we introduce a dynamic representation for multimedia radiological reports, and a design for an automatic MRR facility which uses this representation. The MRR representation is based on temporal logic and semantic networks, and describes the *temporal behaviour* of a MRR : the direct ties between the radiographs, operations and annotations are given in terms of their synchronization requirements for the MRR playback.

The thesis is organized as follows. To begin with, a literature survey of existing techniques for the representation of multimedia documents is given in Chapter 2. The survey reveals inadequacies in the existing models and justifies the current work. The following two chapters introduce the tools from which the temporal representation is developed. A selection of knowledge representations are explained briefly in Chapter 3, and Chapter 4 gives a detailed explanation of interval based temporal logic. Chapter 5 introduces a sample digital radiology system, briefly describing the MUSIC multimedia communications system as it pertains to requirements imposed on the MRR representation. The central chapter is Chapter 6 which details the development of the proposed representation and demonstrates its suitability with a few sample multimedia radiological reports. Chapter 7 considers how the proposed MRR representation can be used in an automatic MRR facility. Chapter 8 concludes the thesis by summarizing the major points developed and looking forward to the work that remains to be done.

Chapter 2 A LITERATURE SURVEY OF MULTIMEDIA DOCUMENTS

Multimedia documents are composed of inter-related segments of image, voice, text and data. An overview of the multimedia field is given to introduce current work on multimedia and to clarify the distinguishing posture of the thesis. Two areas of interest are discussed. The first concerns different models for the storage and transmission of multimedia documents; the second considers their use under existing facilities.

2.1 Multimedia Document Models

Multimedia document models have their roots in office automation where the modern electronic office is typified by a distributed (ie. multi-site) multi-vendor architecture and many individual application programs. The need has arisen to move information from one application to another (document sharing) and from one site to another (document communication). Both types of document exchange demand that a document representation standard be established.

Two standards are discussed. The first - the ODA standard - is set out by ISO [ISO88] but is representative of those standards worked on in parallel by CCITT and the ECMA [Horak85]. The ODA is intended as a general or universal multimedia document standard. The second is a specialized standard developed by DARPA Internet. It is noteworthy because of a difference in emphasis in the document model.

The Office Document Architecture Model

The Office Document Architecture (ODA) defines [ISO88] a document as being any amount of structured two-dimensional information¹ that can be exchanged as a unit and is intended for human perception. It contends that any such document is distinguishable by its contents and by its internal organization. The internal organization is said to be a union of two components which "provide alternative but complementary views of the same document" [ISO88] : the logical structure and the layout structure.

The *logical structure* of a document depicts the structural arrangement of the document's content. The content is mapped onto a hierarchy of *logical objects*, representing such logical components as sections, paragraphs, figures, and footnotes.

¹ The ODA according to ISO 8613 handles documents made up of text, graphics, tables and image contents. It is designed to be extendible to include sound content.

The logical structure is established during the editing process. The *layout structure* expresses the document's content in terms of its physical presentation. A hierarchy of *layout objects*, such as pages and blocks, is associated with content portions so as to reflect their presentation order and relative placement. The layout structure is usually established during a formatting process.

Objects - both logical and layout - are grouped into object classes. An *object class* identifies properties and constraints shared by all member objects. The class definition provides a consistent standard for creating and manipulating objects in this class. A set of layout object classes are defined by the ODA but the logical object classes are document-dependent and user-definable. For instance, the logical objects making up a letter (date, addressee and closing signature) differ from the logical objects composing a book (table of contents and chapters).

Like objects, documents with common characteristics can be grouped together into *document classes*, such as memos, order forms and reports. A document class is described by its logical and layout generic document structures. A logical (layout) *generic document structure* is the set of logical (layout) object classes defined for this particular type of document which describes all potential logical (layout) arrangements of this document class.

Figure 2-1 depicts the overall document architecture model. On the left side is the generic document structures for a document class - the set of logical and layout

NOTE: Taken from (Horak85)

Figure 2-1 The Office Document Architecture Model

object classes. The content portions of this left side identify data that is common to all documents in this class. The right side of the drawing describes a specific document that belongs to this document class. The hierarchies here correspond to the specific logical and layout structures of this one document. The content portions are the actual data unique to this document.

Figure 2-2 attempts to clarify the preceding discussion by modelling a letter. Figure 2-2a contains the generic document structures. All letters are composed of four logical objects : a date, an addressee, a body consisting of one or more paragraphs, and a closing signature. "From" and "Sincerely" are common to all letters. The generic layout structure shows the basic layout common to all letters. A particular letter is represented in Figure 2-2b. The specific logical structure associates each logical object with the appropriate portion of this letter. Only one paragraph is present in this case. The specific layout structure arranges the layout of this letter's content.

The ODA is inadequate as a representation for a MRR. The failings lie both in the concept of and the processing of multimedia documents.

Generic Logical Structure

Generic Layout Structure

Figure 2.2(a) Generic Document Structure

Specific Logical Structure

Specific Layout Structure

Figure 2.2(b) Specific Document Structure

Figure 2-2 The ODA model for a Simple Letter Document

The MRR concept of a multimedia document centres on binding voice to visual displays which are composed of images and image processing operations. In particular, these operations are an integral part of the MRR concept. In contrast, the ODA concept is heavily text-biased. Text is the primary medium. Images and graphics simply illustrate the text. The ODA does not support image processing operations because images are thought of in terms of whole static pictures.

The ODA's document processing model does not easily support the unobtrusive external interface required of the automatic MRR facility. For creation, the editing process is more geared to environments where the author is aware of and specifies a structure for his document. For playback, the ODA can describe temporal dependencies in its layout structure by way of implicit connotations on the ordering of layout objects. This approach, however, cannot tightly specify timing constraints nor can it incorporate the possible file retrieval delays.

Essentially, the ODA is a static model of a multimedia document. Its complexity lies in the diversity of documents that must be supported; the same standard must handle everything from the simple to the complex. On the other hand, the MRR is a very specific application. The complexity lies in its requirement to be manipulated unobtrusively yet intelligently. The MRR is a dynamic report and, hence, requires a dynamic document model.

The Multimedia Message Content Protocol

The Multimedia Message Content Protocol (MMCP) [GarciaLuna Aceves84 and Forsdick84] was developed by DARPA for its own messaging system in Internet and is, therefore, based on the transmission of multimedia documents between dissimilar computer systems.

Like the ODA, the MMCP distinguishes between two arrangements of the document body : logical relationships (logical structure) and presentation relationships (layout structure). In contrast to the ODA, the MMCP divides presentation relations into spatial and temporal relations. In the document body representation, however, the MMCP specifies only the temporal relationships because the logical and spatial relations, it says, are obvious. In fact, omitting the spatial relations provides flexibility in layouts required by computers with different screen sizes and windowing capabilities.

The document body representation is a tree structure (See Figure 2-3 for an example). The nodes are called *presentation descriptors*. The terminal nodes are a special type called a presentation element (pe) and they map to some portion of the actual data content of the document. A presentation descriptor (pd) is recursively defined as a temporal relation between a group of other pds. Four types of pds (other than the pe type) exist :

1. Sequential presentation of the group members.
2. Simultaneous presentation of the group members.
3. Independent presentation of the group members : the time ordering is inconsequential as long as no resource conflicts exists.
4. Presentation of a single document : allows documents to be nested within others.

Figure 2-3 is the tree structure for a sample multimedia document. This document consists of two sequential portions, each of which is a simultaneous presentation of two data components. The first portion plays a voice structure in conjunction with a text display. The second portion synchronizes a voice playback with an image display.

In the MMCP, the interpretations of the two temporal relations - sequential and simultaneous - were left up to each application. To illustrate, sequential display of "n" images may be done by displaying the images, one after another, on the same screen; alternatively, the one screen could be divided into "n" windows with one image displayed per window. Like the omission of spatial relations, leaving the interpretation of the temporal relations open achieves a document body representation which is machine independent.

Although the MMCP also neglects image processing operations as part of a multimedia document, its emphasis on the temporal aspects of a presentation is more

Figure 2-3 The MMCP Representation of a Multimedia Document

in tune with our requirements. Its selection of sequential and simultaneous as the temporal descriptors is typical of similar systems which support automatic presentations of multimedia documents. These temporal descriptors are not considered descriptive nor flexible enough for the MRR application. They relate to the content portions of a document, not to the time involved in retrieving these components over a distributed storage system. Because the time delays are not described, they cannot be flexibly accounted for in the document presentation.

2.2 Multimedia Facilities

Whereas multimedia document models are concerned with document standardization, multimedia facilities deal directly with the utility of multimedia documents for information exchange. The usefulness of multimedia documents is seen in two areas : media integration and multimedia databases.

Media integration is the use of multiple media within the same document. Media integration demands two features from the multimedia facility : homogeneous treatment of the multiple media from the user's perspective, and consolidation of different components into a single meaningful document.

Consider the CCWS [Poggio85] multimedia information system as one approach to media integration. The CCWS combines into a single environment several specialized

"component" editors, one for each supported medium. The component editors are used to edit and view the content of the multimedia document. Homogeneity is achieved by a command interface common to all tools.

Consolidation of the individual media components into a single document is done by way of separate, higher-level "document editors" : the Prompter Editor for highly structured documents such as order forms, and the Structure Editor for free-form reports. The document editors are visual, object-oriented editors that manage both the logical and temporal structure of a document. The logical structure of a multimedia document is considered to be a hierarchical arrangement of multimedia objects, down to the individual media - much like the ODA logical structure. The temporal structure describes how the author intends the multimedia objects to be combined over time. The temporal relations supported are "sequential" and "simultaneous", similar to the presentation relations defined by the MMCP. Both structures are stored in a single tree-structured data representation, a sample of which is shown in Figure 2-4. The document editors provide a user with direct access to this tree structure, allowing the user explicit control over the creation and manipulation of the document structure, not the document content. The document editors can be used to view the document structure as a means of exploring a large document : following the links of the tree structure with the document editors, and invoking the component editors to look at the content of those objects of interest. Alternatively, the document editors can operate on the tree-structure, particularly the temporal structure, to provide an automatic presentation of a document, say to an inexperienced user.

Figure 2-4 The Tree Structured Representation of a Multimedia Document Used by CCWS.

The second area, multimedia databases, is epitomized by Hypermedia [Perry87].

According to [Woelk86], the principle of Hypermedia is that :

"the power of a multimedia system does not come from the creation of static documents which look nice. Rather, it will come from creating a body of information which can be shared by many users."

Information in books or speeches is presented linearly in a logical flow of thoughts. Yet, information tends to be associative, with one topic related to another. This branching of thought is evidenced in text by the use of footnotes, sidebars, cross-sectional referrals and references. Hypermedia accomodates human thought processes by allowing non-linear writing. Authors are encouraged to define connections that link together information within and between documents, creating paths through related topics which a user can follow. Documents are, therefore, not considered to stand in isolation, but are parts in a single underlying database. The role of the multimedia facility is to provide an appropriate method of access to this database.

Intermedia [Yankelovich88] is a Hypermedia implementation. Like CCWS, Intermedia is a combination of several component editors, in particular five application tools - a text editor, a graphics editor, a scanned image viewer, a 3-D graphics editor and a timeline editor (to create a chronology of date-event pairs). The user interface for each tool is similar and is styled after the MacIntosh/Microsoft tools with windowing and pull-down menus. Unlike CCWS, there is no separate document editor. Instead, consolidation is done by incorporating the Hypermedia functionality - the assertion and traversal of links - into each of the five applications as just another set

of commands in yet another pull-down menu. In this way, the notions of link creation and traversal are not separated from those of editing and viewing of a document's content.

Links can be established between entire documents, between specific locations - called blocks - or combinations thereof. A block may range from a single point to an entire document. The links and blocks are however not part of the document. Instead, Intermedia records this information in *webs* which are superimposed over documents. Multiple webs can be placed over a single (group of) document(s), leading to the notion of context. Different users can create webs over the same database, each web reflecting their perspective on the interaction of the underlying documents. A reader thus operates within one context at a time, exploring the flow of thought from one viewpoint at a time.

In effect, the concept of a self-contained multimedia document becomes fuzzy in Intermedia. A multimedia document becomes equivalent to a web, to one access pattern to the multimedia database. A multimedia document is simply one perspective on the multimedia database.

In conclusion, the handling of multimedia documents by the multimedia facilities discussed above has two shortcomings with respect to MRRs. First, voice is usually treated as an after-thought, as an appendage to the main document. In the MRR, voice

plays a more central role. In fact, voice controls the flow of the MRR presentation.

Second, the author plays an active role in the definition of the document's structure. He explicitly edits both (or either) the logical and presentation structures. For the MRR, the MRR must be automatically created, leaving the radiologist largely ignorant of its structure.

A number of useful points, however, can be extracted from the above discussion. For an integrated MRR, the different media should be handled consistently, both in the representation of the MRR and in the facility's user interface. Differences should be hidden in the routines that manipulate the MRR. Also, the notion of multimedia databases as found in Intermedia may be applicable to the long-range question of MRR management, to be considered in Chapter 8.

Chapter 3 A SELECTION OF KNOWLEDGE REPRESENTATIONS

A Knowledge Representation manages information about a domain. It not only stores knowledge in data structures much like traditional databases, but also includes methods of extending this explicit knowledge base into a larger virtual knowledge base. There are many types of knowledge representations, examples of which can be found in [Barr81]. Three are introduced in this chapter : semantic networks, frames and production systems. Semantic networks form the foundation of our MRR representation. Frames and production systems are used in the implementation of the automatic MRR facility.

3.1. Semantic Networks

A semantic network consists of nodes and arcs, normally portrayed as circles and arrows, respectively (Figure 3-1). Semantic networks resemble other directed binary graphs but are accessed according to their semantic content, rather than by traditional indexing schemes such as alphabetical or numerical ordering. Semantics are defined in a set of procedures that operate on the semantic network, and they establish what the nodes and arcs mean and how they are to be used. No formal, universal semantics exist. Instead, semantics are chosen based on the goals and domain of the current application. A node may symbolize objects, actions, events, situations, concepts, etc.. Arcs represent any binary relationships between two nodes. A simple example in Figure 3-2 demonstrates a common arrangement.

Figure 3-1 Graphics Notation for Semantic Networks

Figure 3-2 A Simple Semantic Network

Nodes :

- Ralph and Bob are instances, particular parameters of the domain.
- Poodle, dog and postman are classes, a type representing instances sharing one or more characteristics.

Arcs :

- is_a is an instantiation (Data).
- is_a (c) is a class specialization (General knowledge).
- bit is a binary predicate relationship.

The basic notion of a semantic network is largely free-form and unstructured. In large or complex domains, extensions have been developed to add some organization. One extension of interest is partitions [Hendrix75]. Partitions subdivide a semantic network into net spaces. Net spaces group together related nodes and arcs so that part of a node's/arc's meaning is given by its associated net space. In a way, the net space defines a context for the nodes/arcs. Partitions organize and break down the network, allowing a user to be concerned only with the partition which is relevant to his query. Some examples of the uses of partitions are given below.

- When using a semantic network to encode predicate logic, partitions can delimit the scope of quantified variables.
- In a planning problem, partitions can encode alternative plans.
- Partitions can be used to modularize a large semantic network in order to focus attention at particular levels of detail.

3.2 Frames

Frames [Cuadrado86] are a structured data representation for the description of objects or classes of objects. Frames are like semantic networks in that they preserve semantic relationships between objects. Frames are different though because :

1. Frames are well-defined aggregate data structures used to organize knowledge.
2. A formal language, based on the frame data structures, exists which provides procedures for some efficient special-purpose reasoning.

For a formal analysis of frames, refer to [Nebel85].

The basic structure for a frame is shown in Figure 3-3. A frame name is followed by a variable number of *slots* each of which concerns an attribute of the object or object class being described. A slot can take on multiple *facets* where a facet annotates one aspect of the attribute. Facets are commonly used to express :

- a data value for the attribute, or
- properties of the attribute, such as default values of relationships to other objects, or
- procedures to be performed upon insertion or retraction of the slot.

The procedures can enforce constraints on the slot value such as cardinality. Alternatively, the procedures can propagate the effects of the operation, such as using the inserted slot value to calculate the value for some other (separate but related) slot.

The sample frame in Figure 3-4 describes a car. The frame says that a car has four wheels and usually has four doors. If the number of doors is explicitly asserted (by adding a value facet to this slot) then the procedure *add_doors* is to be invoked, say to ensure that the value given is an integer.

Figure 3-3 The Basic Structure for a Frame.

Figure 3-4 A Frame Describing a Typical Car

Figure 3-5 A Sample Frame Hierarchy - A Horse Taxonomy

There are two types of frames : class and instance. Class frames describe a group of objects sharing one or more characteristics. An instance frame represents a specific object. Using well-defined slots, frames are organized into a taxonomy called a generalization hierarchy, such as the one shown in Figure 3-5. The *ako* slot denotes a class-subclass relationship between two class frames. For instance, Arabians are a subclass of horses. The *is_a* slot denotes class membership, and links an instance frame to a class frame. In Figure 3-5, Azkaborro is an Arabian.

Each frame can "inherit" attributes from frames higher up in the hierarchy. For class frames, inheritance allows data descriptions by specialization. A class frame thus contains only those attributes peculiar to the subclass, inheriting the rest from its parent(s). In Figure 3-5, the horse frame contains characteristics shared by all horses (four legs and hooves); the Arabian frame contains those characteristics that make this breed of horses distinctive (small head and large lung capacity). For an instance frame, its parent provides two types of information. First, through the class hierarchy, general knowledge about the class to which this instance frame belongs is given. Azkaborro is known to be a four-legged, hooven creature with a small neck and large lung capacity. Second, parent class frame(s) can serve as prototypes for the creation of new instances, listing attributes that they must have.

The reasoning services provided by the frame language are hidden in the assertion and retrieval operations. The primary service is a deductive retrieval which takes advantage of the inheritance hierarchy to deduce the value of an attribute when necessary. If the attribute is not given explicitly as a slot in a frame, the retrieval procedure moves up the hierarchy, searching for a frame with this slot. The slot found can either provide a value or, failing this, a procedure by which the value can be computed. A second service is automatic knowledge propagation, using the facet procedures described above. The assertion procedure automatically triggers any facet procedures defined for the attribute - either explicitly or through inheritance - to propagate the change of knowledge to related areas of the frames database.

Frames are a natural tool for database modelling due to its concise, structural representation and the automatic procedures for integrity maintenance. For knowledge engineering, the partitioning and indexing of frames into class groupings is a natural expression of context [Fikes85].

3.3 Production Systems

A production system [Nilsson80] represents knowledge in the form of condition-action pairs called production rules :

IF condition THEN action

A production system is composed of three parts :

- 1) The global database : a set of facts about the state of the domain.
- 2) The knowledge base : a set of production rules which operate on the global database to infer new facts.
- 3) The control strategy : A program which decides what rules to apply next to achieve the desired inference. Typical strategies are backchaining (depth-first search) and forward chaining (breadth-first search).

Production systems are often used in applications that describe how to do a specific task because the IF-THEN rule format is a natural expression for what to do in a given situation. In theory, the rules are independent, lending itself to a modular representation of knowledge that allows for continual growth or modification of the knowledge base. On the other hand, configuration control of the rule base is difficult in large applications. One solution has been to use frames to encode the rules, thereby imposing an organization on the rules according to their scope and context.

Chapter 4 TEMPORAL LOGIC

Temporal logic is used to describe and reason about changes over time. Characterizing a MRR by its temporal behaviour suggests the use of temporal logic to describe the synchronization requirements for the MRR's playback. Several types of temporal logic exist, however, and an appropriate choice must be made.

Modal temporal logic applies temporal modal operators to predicate formulae in order to show how their truths vary over time. To be accurate, modal temporal logic demands rigorous and formal domain specifications.

Situation calculus is a state-based representation in which changes over time are modelled by events which cause a transition from one state to another. Situation calculus cannot reason about concurrent events or events with duration.

Event calculus associates knowledge with time periods rather than states. Two main streams exist where the time period is a time point and an interval, respectively. Because even those events which initially appear to be instantaneous may need to be decomposed, interval based temporal logic is considered to be a more flexible and natural expression of activities [Allen83b]. It is described in the following section.

4.1 Interval Based Temporal Logic

Informally, an interval is defined as a continuous, finite length of time. An event or activity is bound to an interval so that the interval comes to represent the occurrence of this activity. The time-ordering of activities is achieved by describing the temporal relationships between their associated intervals. The resulting temporal knowledge representation is domain-independent [Allen81] : it provides only for the description and manipulation of the temporal relations between events; it is separate from the representation used for the remainder of the event specifications.

More formally, time is modelled as a dense, linearly ordered domain, similar to the set of real numbers. Time is made up of an infinite sequence of zero-width timepoints. Like real numbers, time can be visualized as an infinite, continuous, horizontal axis where time passes as one moves left to right (See Figure 4-1).

Figure 4-1 Time as a Continuous Infinite Axis

Arithmetic relational operators perform comparisons between points along the timeline. Later times are interpreted as having a greater "magnitude" than earlier times, so that :

$X > Y$ denotes	"Timepoint X is after Timepoint Y"
$X < Y$ denotes	"Timepoint X is before Timepoint Y"
$X = Y$ denotes	"Timepoint X is at the same time as Timepoint Y"

All rules of equality, symmetry and transitivity also hold. To illustrate, the situation shown in Figure 4-1 is denoted by :

Time A < Time B or by symmetry,
Time B > Time A

An interval is seen as a finite subsection of the timeline (Figure 4-2). The interval's start point is denoted by I-, its end point as I+. The duration is equal to the absolute distance between the two endpoints.

Figure 4-2 A Time Interval

Temporal relations describe the temporal ordering between two intervals. A complete set of thirteen primitive temporal relations (Table 4.1) was established by Allen with which any temporal scenario can be described. The set was found by listing all possible combinations of the relative positions of two intervals' start and end points, constrained by the basic assumption that an interval's end point is always after its start point. Of the thirteen, twelve comprise six pairs of inverse relations. An inverse pair consists of two temporal relations, *relation* and *inverse_relation*, such that *relation* (I,J) is equivalent to *inverse_relation* (J,I). The remaining relation, *equals*, is its own inverse. The temporal relations are mutually disjoint : For any two intervals, only one temporal relation can hold.

Consider an example. Because he is hungry, John phones out for a snack. When the pizza arrives, John immediately begins, eating until he is no longer hungry. Figure 4-3 contains the timelines which, in IBTL, are described by :

phone *during* hungry
 arrive *during* hungry and arrive *after* phone
 arrive *meets* eating
 eating *finishes* hungry

The temporal description is completely relative. No mention is made of absolute times or durations. Only the relative ordering of the intervals is known.

Figure 4-3 Timelines for a Sample Scenario

Relation (I,J)	Definition	Illustration	Inverse (J,I)
before	$I+ < J-$	III JJJJ	after
meets	$I+ = J-$	III JJJJ	met_by
during	$I- > J-$ and $I- < J+$ and $I+ < J+$	III JJJJJ	contains
overlaps	$I- < J-$ and $I+ > J-$ and $I+ < J+$	IIII JJJJJ	overlapped_by
starts	$I- = J-$ and $I+ < J+$	III JJJJJ	started_by
finishes	$I- > J-$ and $I+ = J+$	III JJJJJJ	finished_by
equals	$I- = J-$ and $I+ = J+$	IIIII JJJJJ	equals
after	$I- > J+$	JJJ III	before
met_by	$I- = J+$	III JJJJ	meets
contains	$I- < J-$ and $I+ > J+$	IIIII JJJ	during
overlapped_by	$I- > J-$ and $I- < J+$ and $I+ > J+$	III JJJJ	overlaps
started_by	$I- = J-$ and $I+ > J+$	IIIII JJJJ	starts
finished_by	$I- < J-$ and $I+ = J+$	IIIII JJJJ	finishes

Table 4.1

The Primitive Temporal Relations

In addition to the thirteen primitive relations, the logic permits the addition of *defined relations*. A defined relation is a disjunction of two or more primitive temporal relations. The disjunction expresses uncertainty about an temporal relation, and is useful in situations where the exact connection between two intervals is not known. Consider a situation where it is known simply that Interval J occurs sometime after Interval I (Figure 4-4). Interval J could either begin immediately as Interval I terminates or after some (unspecified) time lapse. Suppose

precedes = before or meets.

Then, the situation in Figure 4-4 is completely described by *precedes* (I,J). The relation holds if, and only if, Interval I is before Interval J or Interval I meets Interval J. Defined relations are simply a more convenient, concise notation for often-used disjunctions and, as stated in [Ladkin87], are correct as long as they are derived from the primitive relations, or are translated back to the primitive relations before being operated on.

Figure 4-4 Timelines for the precedes Defined Relation

The transitivity property is the basic mechanism for reasoning about time. It is defined as :

$$(I \ r \ K \text{ and } K \ s \ J) \Rightarrow I \ t \ J$$

If intervals I and J are related to a common interval K, then I and J are related by the transitive relation *t*, and is called the composition of *r* and *s*. The transitive relation *t* is always a disjunction of one or more primitive temporal relations. To illustrate, reconsider our example in Figure 4-3. Since

$$\begin{array}{ll} \text{phone } \textit{before} \text{ arrive} & (I \ r \ K) \quad \text{and} \\ \text{arrive } \textit{before} \text{ eating} & (K \ s \ J) \end{array}$$

from common sense, we know that :

$$\text{phone } \textit{before} \text{ eating} \quad (I \ t \ J)$$

before is the composition of *before* and *before*.

Allen produced a transitivity table which contains the compositions of all pairs of primitive temporal relations [Allen83b]. With 13 relations, it has 169 entries. The transitivity table is used to infer information not explicit in the temporal description.

Applications

Interval based temporal logic is a useful tool for truth maintenance [Allen83b]. In language understanding, truth maintenance can capture the tense information in sentences. In information systems, truth maintenance can record the time course of database updates.

Temporal information is maintained in a semantic network whose nodes symbolize intervals, and whose arcs identify the possible temporal relations between the two intervals. The network is assumed to completely describe how any two intervals are related; hence, the network is fully inter-connected and reduced.

An algorithm for maintaining the network based on constraint propagation is shown in Figure 4-5. New temporal information is equivalent to adding a new arc to the network (Line 1). Due to the transitivity property, this new arc may constrain the temporal relationships of the two intervals with other intervals in the network. For this reason, the temporal relationships must be re-computed for all nodes related to the original two nodes (Lines 2 and 6). If a temporal relationship is affected, then it must be updated, ie. a new arc is added. Adding this arc may in turn introduce constraints to further nodes; hence, the recursive calls to ADD (Lines 5 and 9). In effect, the original constraint is propagated to all nodes in the network. The algorithm continues until no further constraints are added.

```

ADD New (I,J)  (Procedure for adding an arc)
*****

(1)   Arc (I,J) := New (I,J)  {Store the new arc}

      { Propagate effects of new arc to all
        nodes related to node J }

(2)   FOR each node K related to J by Arc (K,J)
      LOOP
(3)     New (K,J) := Arc (K,J)** AND
          Transitive ( Arc (K,I), New(I,J) )
(4)     IF New (K,J) is a subset of Arc (K,J)
      THEN
(5)       ADD New (K,J)
          { Recursive Call to add this
            arc and propagate its effect }

      END IF
      END FOR

      { Propagate effects of new arc to all
        nodes related to node I }

(6)   FOR each node K related to I by Arc (K,I)
      LOOP
(7)     New (K,I) := Arc (K,I)** AND
          Transitive ( Arc (K,J), New(J,I) )
(8)     IF New (K,I) is a subset of Arc (K,I)
      THEN
(9)       ADD New (K,I)
          { Recursive Call to add this
            arc and propagate its effect }

      END IF
      END FOR

** : A non-existent arc between two nodes is
    equivalent to an "OR" of all 13 primitive
    temporal relations; ie. any temporal
    ordering between the two nodes is possible.

```

Figure 4-5 An Algorithm for Maintaining Temporal Information in a Fully Connected Network

Figure 4-6 gives an example of constraint propagation. The solid lines denote the current state of the network; a dotted line symbolizes a temporal fact to be added. In Figure 4-6(a), a new temporal relation is specified between Nodes A and Y : $New(I,J) = \text{"before (A,Y)"}.$ The algorithm first considers the effect of this new arc on all nodes related to Node Y (Line 2); in this case, to Node X. The current relationship between Nodes X and Y is *"before or meets"*. Through transitivity, the new arc $New(I,J) = \text{"before (A,Y)"}.$ constrains the temporal ordering between Nodes X and Y to *"before"*.

$$(X \text{ finished_by } A \text{ AND } A \text{ before } Y) \Rightarrow X \text{ before } Y$$

The new relation between Nodes X and Y is computed in Line 3 by intersecting the current relationship (*before or meets*) with the transitive relation (*before*). In this case, the new relation is $New(K,J) = \text{"before (X,Y)"}.$ It is more constrained, more fully specified, than originally and therefore must be added to the network (Line 5). If we were to follow through the recursive call to add this new relation, we would see that no new information is added because the network is already fully constrained. Upon returning, the algorithm continues now to propagate the effects of the original new arc ($New(I,J) = \text{"before (A,Y)"}.$) to all nodes related to Node A (Line 6). The only related node is again Node X. Because the network is already fully reduced, when the relation between Nodes X and A is re-computed in Line 7, no additional information is added, and the algorithm halts. Figure 4-6(b) is the final network.

Figure 4-6(a) The Original Network with An Arc Being Added

Figure 4-6 (b) The Final Network After Constraint Propagation

Obviously, constraint propagation becomes costly in time and in space as the number of nodes in a network increase. To reduce the propagation, reference intervals are introduced. Reference intervals "partition" a network into groups of intervals. The reference intervals are themselves time intervals : a higher level "reference interval" network maintains the temporal relations between all reference intervals. For nodes in the same reference intervals (ie. partition), the temporal relations are fully computed. For nodes in different reference intervals, their relationship is derivable by finding a path between the two reference intervals. The same information is available as in the original fully-connected network; reference intervals simply impose a structure on the network.

Interval based temporal logic has also been used in problem solving systems. Planning involves scheduling a series of events to accomplish a task. Temporal logic is useful for the temporal qualification of the actions and for the specification of timing constraints such as resource availability. Interval based temporal logic is particularly useful because it can model actions that have (variable) duration, and it can describe simultaneous or overlapping actions. The transitivity table provides a formalism for reasoning about interacting activities. Examples can be seen in [Allen83a] and [Firby87].

In conclusion, interval based temporal logic is a domain-independent representation of temporal knowledge. It is a relative temporal representation that can model concurrent activities of uncertain duration or ordering. In Chapter 6, we will use interval based temporal logic as the basis for our representation of a MRR.

Chapter 5 THE MUSIC RADIOLOGY

MULTIMEDIA

COMMUNICATION SYSTEM

This chapter introduces the Multimedia System for Integrated Communications (MUSIC) testbed, a multimedia communications system for the transfer of radiological information between departments. The system is being developed at the Ottawa Civic Hospital, and initially only the Emergency and Radiology departments are linked. The discussion is given at a functional level and with regard to the effects on the multimedia radiological report only.

The MUSIC testbed is shown in Figure 5-1. A data network and a voice network interconnect four types of stations : the Radiologist Workstation (WS), the Physician WS, the Image Entry WS and the File Server Station.

Figure 5-1 The MUSIC Testbed System

The Radiologist WS and the Physician WS differ only in who uses them. Each consists of a control monitor, an image monitor, a telephone handset, a mouse (active on both monitors) and some local disk storage. The control monitor provides database access by which the user navigates through the patient files. A single digital film image - selected on the control screen - is displayed on the image monitor. A set of icons located on the bottom of the image monitor provide image processing operations such as

ZOOM : Magnification of a selected area (called a window) of the displayed image.

CONTRAST: Various digital contrast enhancements to be performed on a window of the displayed image.

TOGGLE : Alternation between positive and negative contrast.

MARKER : Pointing or annotation of a point location on the displayed image.

LENGTH : Measurement and annotation of a distance between two points on the displayed image.

ANGLE : Measurement and annotation of an angle on the displayed image.

RESTORE : Removal of any of the previous operations, restoring the display to that of the film image alone.

Presently, only one image monitor is available. Research is being conducted on the adequacy of one image monitor; that is, whether the users can efficiently perform their duties by viewing images sequentially, or if concurrent image displays are required. Future configurations may have multiple image monitors, most likely two. Our work should remain open to this possibility.

As the name suggests, the Image Entry WS is used to digitize the original x-ray films. The Image Entry WS is like the previous two workstations, except that it also has a laser scanner and additional disk storage which serves as a network store.

The File Server Station is simply a PC-compatible together with a voice message system. The PC maintains the patient files, keeping track of the type and location of all data. The Voice Message System is a centralized voice server where all voice reports are stored.

The data network is used for the movement of data between stations and for communication between stations. The data movements are as follows. Films are digitized at the Image Entry WS. Two copies are made : a subsampled image used for normal viewing, and a full resolution *zoom image* used for magnifications. Copies of the subsampled image are distributed to the local storage of both the Radiologist and the Physician WSs, the intent being to provide rapid local display times. Because of its size and less frequent usage, the zoom image is kept centrally in the network store on the Image Entry WS. Consequently, magnifications are subject to network delays while the appropriate zoom file is transferred over the data network to the WS.

The voice network uses a digital PBX to link the File Server's Voice Message System to the WSs' telephone handsets. In order to record or playback a voice report, a session must first be established. Request commands are sent over the data network from the WS to the File Server station. The Voice Message System makes the connection by simply dialing up the WS's handset through the PBX. Like a phone call, the line remains dedicated for the session. The voice is sent directly over the analog telephone lines to guarantee real-time voice service. Playback commands such as rewind and pause are, however, sent over the data network.

Real-time communications is guaranteed by imposing a priority order on the different kinds of network traffic. In descending order, the priority is given to network commands (including playback commands), zoom images and normal images.

The distributed network architecture of the MUSIC testbed introduces certain timing delays in the retrieval and manipulation of the radiological data. In the next chapter, we will see that these timing characteristics play a role in the presentation of a multimedia radiological report, and must be accounted for in the temporal representation of a MRR.

Chapter 6 A TEMPORAL REPRESENTATION OF A MULTIMEDIA RADIOLOGICAL REPORT

A multimedia radiological report (MRR) is an animated version of the examination report in which the contents of the examination report are presented in conjunction with the set of radiographic images and operations performed on these images. A temporal representation for a MRR defines the ties between the report, images and operations in terms of their synchronization requirements; it describes the temporal behaviour of the MRR presentation.

The first section in this chapter discusses the temporal behaviour of MRRs, explaining the temporal information that must be encoded. The second section introduces the chosen representation - a temporal semantic network. The representation is demonstrated on a group of sample MRRs in the final section.

6.1 The Temporal Behaviour of a Multimedia Radiological Report

The temporal behaviour of a MRR can be decomposed into a set of *primitive activities* common to any MRR. The temporal behaviour is thus described by the temporal characteristics of these primitive activities, and the ways in which they interact. As a first approximation, the primitive activities are assumed to consist of the functions available through the MUSIC workstations, as described in Chapter 5. These have been divided into two groups - visual and audio activities - whose temporal characteristics are discussed separately in the first two subsections, respectively. The third subsection discusses how the various activities interact. The major findings are summarized in the final subsection.

To facilitate the ensuing discussions, a temporal notation to describe an activity is introduced.

$$A = (\text{Interval}, \text{Context})$$

identifies an activity that occurs over the time period denoted by Interval, within the given context. The Interval is a finite continuous length of time beginning and ending at Interval- and Interval+, respectively. An activity's Interval serves as its unique identifier, distinguishing it from other activities of the same type. The Context will be defined later.

6.1.1 The Temporal Characteristics of Visual Activities

Visual activities include all activities that affect the display on a single¹ image screen². The visual activities include :

1. LOAD activity : The process of loading a new x-ray film image for display on the image screen.
2. Image activities : Image processing operations corresponding to those available on the radiologist and physician WSs (Chapter 5), and including : ZOOM, CONTRAST, TOGGLE, MARKER, LENGTH, ANGLE and RESTORE.

Five temporal properties of an activity can be identified. The properties are defined below, and are illustrated by examples drawn from the sample screen display in Figure 6-1. This screen display is the product of three visual activities :

1. LOAD (L_Interval, L_Context) : Load Image Z
2. ZOOM (Z_Interval, Z_Context) : Magnify a window at (X1,Y1)
3. MARKER (M_Interval,M_Context) : Put a marker at (X2,Y2)

¹ Multiple screens involve interactions between visual activities. They are discussed later in Section 6.1.3.

² Screen is used as a generic term for any logical display area. The term may refer to an entire image monitor or it may correspond to a (one of many) software window(s) within a larger physical screen.

Figure 6-1 A Sample Screen Display Consisting of an Image Display, A Zoom and a Marker.

Sequential Behaviour

Behaviour is sequential if only one activity may occur at one time. A second activity cannot start until the first is completed; in other words,

Given two activities : A1 (A1_Interval, A1_Ctxt) and
A2 (A2_Interval, A2_Ctxt)

Either A1_Interval+ $\geq$ A2_Interval-
or A2_Interval+ $\geq$ A1_Interval-

To illustrate, while the image in Figure 6-1 is being loaded, the system is inactive; the magnification may be invoked only once the load is completed.

ie. L_Interval+ $\leq$ Z_Interval-

All visual activities are sequential.

Persistent Behaviour

An activity is said to be persistent if it has an effect that persists over time until some *terminating activity* occurs that counteracts or destroys this effect. A persistent activity is better described as a causally-related doublet, consisting of an *action* and an *effect*. Our notation is updated accordingly. An activity is described now as :

Activity (Interval, Context) and if the activity is persistent,
Activity* (Interval*, Context*)

The first denotes the *action* and will always exist. The second denotes the *effect* of the action and will exist only if the activity is persistent. Like the actions, the effect occurs over some time interval and has a context. Unlike the action which will always come to completion, the effect will persist until some other action terminates it. The context for the effect is simply the action with which it is associated. "Activity" is now a general term referring to the pair. The "context of the activity" is the context given in the action's descriptor. The doublet notion of an activity allows other activities to be related to either the action or to the effect.

The visual medium is persistent because all visual activities cause a screen display. In Figure 6-1, once the film image is loaded into image memory, the display of the film image persists through both the magnification and the marker activities. Similarly, ZOOM and MARKER also produce displays. All displays will persist until either a LOAD or RESTORE activity takes place. In our new notation, Figure 6-1 is now described as :

LOAD (L_Interval, L_Context) and
LOAD* (L_Interval*, L_Interval)

ZOOM (Z_Interval, Z_Context) and
ZOOM* (Z_Interval*, Z_Interval)

MARKER (M_Interval, M_Context) and
MARKER* (M_Interval*, M_Interval)

Context Dependent Behaviour

Context dependent behaviour concerns activities that must take place in a particular setting or context. Hence, the context argument in our activity descriptor. From a temporal perspective, the context is some other activity that must precede or occur simultaneously with this activity.

All image activities (ie. ZOOM, CONTRAST, MARKER, etc.) are context dependent because they all operate on an existing display. The existing display is their context. The LOAD activity is context independent because, instead of adding to an existing display, it completely replaces the existing display with a new image. A LOAD activity can occur at any time.

Two examples of context dependent behaviour are given in Figure 6-1. We note, first of all, that it would be meaningless to ask for a magnification unless there is a film image already on display. The context of the ZOOM action is the display of the image, ie. the effect of the LOAD activity. In our notation :

ZOOM (Z_Interval, L_Interval')

where L_Interval' uniquely identifies the LOAD activity associated with the required image display.

A more subtle and complex example in Figure 6-1 is the marker pointing to a feature within the zoom window. The **MARKER** action is only defined within the scope of the zoom window : **MARKER (M_Interval, Z_Interval*)**. The marker is an example of a nested context : The marker operates on the zoom window which, as seen above, operates on a loaded image.

Time Invariant Behaviour

Temporal behaviour is time invariant if it is of the same value or is consistent over some time interval. The temporal behaviour is indivisible over that time interval because, from the user's perspective, the temporal behaviour at one timepoint is indistinguishable from that at any other timepoint in the same interval.

The effects of all visual activities - their displays - are time-invariant. The displays are still pictures. The display at the beginning of the time interval is the same as that at the middle or at the end of the interval. As an example, the display of the **MARKER** remains static over the interval **M_Interval***.

The actions of all visual activities will also be considered as time-invariant. Although the screen display changes as the visual action executes, it is not until the display is complete that it is intelligible to the user. The temporal implication is that the visual actions represent the primitive (smallest) time intervals. These intervals

cannot be decomposed : exact synchronization between points within intervals associated with visual actions is not possible. As an example, when putting up the marker in Figure 6-1, M_Interval is the primitive time interval. We cannot distinguish between the two steps of putting an icon at a given position and then labelling the icon.

Behaviour of Random Durations

The temporal behaviour of an activity is of random duration if the length of the activity's time interval is not fixed.

Visual actions all have random duration because, depending on the action, one or more of the following factors may affect the processing speed :

1. The location of a required file;
2. The loading of the network;
3. The computational complexity of an algorithm.

As an example, in order to perform the magnification in Figure 6-1, the zoom image file associated with Image Z must be retrieved through the network from the central store on the Image Entry WS. The retrieval time depends directly upon the amount of traffic on the network.

The time for which a visual effect persists also is random, depending on the playback process. During the automatic presentation, the playback process may proceed at a set pace, allowing a set amount of time for each display. Alternatively, the playback process may allow the user to control the pace, indicating when he is ready for the next display. Additionally, the user may pause the presentation in which case the display continues indefinitely.

Our notation handles random durations by specifying that activities occur over some interval whose endpoints are specified only through relative comparisons to other intervals' endpoints. Using our example, we know only that the LOAD action occurs from $L_Interval-$ to $L_Interval+$ and, because it precedes the ZOOM action, that $L_Interval+ \leq Z_Interval-$.

6.1.2 The Temporal Characteristics of Audio Activities

The activities associated with the audio medium are :

1. CONNECT activity : Establishes a session on the voice network. (Refer to Chapter 5).
2. PLAY activity : Plays a voice segment from the current voice file. Voice segments are defined formally later. Basically, they are some portion of the overall voice report.
3. DISCONNECT activity: Closes the session.

CONNECT and DISCONNECT are referred to collectively as the session activities. The temporal behaviour of the session activities is identical to that of the visual activities and can be handled by the same representation. Only the PLAY activity differs, posing new challenges for our representation. The following discussion, therefore, focuses primarily on the PLAY activity, with the headings reflecting the temporal properties of the PLAY activity.

The following voice report, played back with the following four activities, illustrates the five temporal properties of audio activities.

CONNECT (C_Interval, C_Context)	To establish the session.
PLAY (P_Interval_1, P_Context_1)	To play the first segment.
PLAY (P_Interval_2, P_Context_2)	To play the second segment.
DISCONNECT (D_Interval, D_Context)	To close the session.

Sequential Behaviour

The audio activities are inherently sequential. Because people can concentrate on listening to only one voice at a time, it is sensible to play only one voice segment at one time. Also, a voice segment cannot be played until the session is established.

In our example, sequential behaviour requires that :

$$C_Interval+ \leq P_Interval_1- \quad \text{and}$$

$$P_Interval_1+ \leq P_Interval_2-$$

Ephemeral Behaviour

Unlike the visual activities, the PLAY activity does not persist. As each voice segment plays, it immediately vanishes. PLAY is an example of an ephemeral activity. An ephemeral activity has an action only; it has no effect. It is completely described by the single activity descriptor, as originally given.

The session activities are both persistent. The CONNECT activity in the example is thus represented by a doublet :

```
CONNECT (C_Interval, C_Context)
CONNECT* (C_Interval*, C_Context*).
```

Context Independent Behaviour

The PLAY activity is context independent because, once open, the voice file can be considered as a random access memory. Voice segments can be played in any order. Obviously, indiscriminate access to voice segments will cause words to be taken out of context but there is no physical requirement on their ordering. Indeed, it is the responsibility of the playback process to arrange the segments into coherent speech. This property is viewed as a means of handling user interactions such as allowing the user to halt the presentation and back up to repeat a portion.

From another perspective, a PLAY activity does indeed have a context; namely, the session established by the initial CONNECT activity. For completeness, the context parameter of PLAY activities is thus used to associate a PLAY activity with a particular voice report. We can use our previous example to illustrate this property as follows :

```
PLAY (P_Interval_1, C_Interval*)  
PLAY (P_Interval_2, C_Interval*)
```

The session activities are context dependent. For instance, a session cannot be disconnected unless it was previously connected. Also, a second session should not be connected if a session already exists.

Time Variant Behaviour

Temporal behaviour is time variant if it changes over the time interval. The temporal behaviour is divisible because it differs from one timepoint to another, within the same time interval.

The PLAY activity is time variant because each timepoint during the playback of a voice segment is unique and distinct. It is meaningful to identify specific points within the time interval for a PLAY activity. The sound at one second into a voice segment is markedly different from that two seconds later, a property basic to the effectiveness of oral communication. Time variance will play a role in the definition of voice segments.

The session activities have time-invariant behaviour : either a session is established or it is disconnected.

Behaviour of Fixed Durations

The PLAY activity is of fixed duration, consisting solely of the length of the associated voice segment. Network delays in delivering the PLAY command to the Voice Message System are considered negligible because the network commands are given top priority. In other words, it is assumed that the voice server provides real-time service.

6.1.3 The Temporal Interactions Between Activities

Interaction between activities are now considered. Interactions between media - between the visual and audio activities - are first discussed. The conclusions are then extended to image sequences and to multiple screens.

Temporal Interactions between Visual and Audio Activities

Interactions between audio and visual activities are basic to the animation of examination reports. Three scenarios exist, each presenting different temporal representation problems : a stable display, an unstable display and a progressive display. In this section, a single image screen is assumed.

Scenario 1 : Voice Playback with a Single Fixed Display

In this simplest scenario, a voice segment is played in conjunction with a completely prepared, stable display. The desired screen display is prepared, and then the voice segment is begun. The display remains stable for the duration of the voice segment. The display preparation may actually require a sequence of visual activities; but, because visual activities must occur sequentially, the important temporal interaction is between the voice segment and the final screen display. To be completely accurate, the voice segment is temporally dependent on the effect of the last visual activity. For instance, to play a voice segment with the display in Figure 6-1, the PLAY activity need not consider or even be aware of the LOAD and ZOOM activities.

Scenario 2 : Voice Playback With Changing Displays

In this scenario, the reporting radiologist starts to dictate while waiting for a screen update to complete. For instance, while waiting for the first image(s), the radiologist may dictate the report header : "Reporting for patient John Smith, examination taken on January 1, 1990".

Unlike the previous scenario, the voice segment is temporally dependent on the visual action. The temporal interaction is unique due to the randomness of the visual actions : The relative starts points are known but the relative end points are unknown. The voice segment could end before, with or after the concurrent visual action.

Scenario 3 : Screen Updates Overlap Voice Playback

This scenario is the converse of the previous one : new visual actions are initiated during dictation. The two scenarios are not interchangeable because of the time variant property of voice. To explain, consider the dictation timeline in Figure 6-2. Two situations are possible, each involving different temporal interactions.

The first situation is a "don't care" situation. It occurs when the voice segment does not refer to the new display. Here, the temporal interaction between

Figure 6-2 A Timeline For a Screen Update During a Voice Playback.

the **PLAY** and the **LOAD** is unimportant and can be varied. For example, if, in general, the **LOAD** action is expected to take longer than the given voice segment, it would be more efficient to begin the **LOAD** action before starting up the voice. The representation of this temporal interaction should allow for this flexibility.

The second situation arises when some portion of the voice segment does refer to the new display. For instance, let the voice segment contain : "Reporting for patient John Smith on January 1, and on this first image, the frontal view, signs of pneumonia are evident in the upper quadrant of the right lung... ". Should the load of the image occur so quickly that the radiologist did not need to pause, as in Figure 6-2, the temporal interaction becomes complex because :

1. Different portions of the same voice segment are associated with different visual activities.
2. The random duration of the visual activities means that the timing during playback may differ significantly from the dictation timeline.

This second situation introduces the idea of *contextual segmentation*. The voice report is artificially subdivided into logical segments such that the temporal interactions are reduced to the (three) simpler scenarios seen above. For our sample voice segment, two logical segments would be created :

1. "Reporting for patient John Smith, examination taken on January 1, 1990"
2. "and on this first image, the frontal view, signs of pneumonia are evident in the upper quadrant of the right lung... ".

The first logical voice segment is a "don't care" situation. The second is an example of voice playback with a fixed display.

Temporal Interactions with Image Sequences

An image sequence is a set of images that are presented as a group.

Following are three situations that use image sequences :

1. **Time Progression** : The progression of a disease is traced by comparing a set of equivalent views taken over time.
2. **Left and Right Symmetry** : The left/right symmetry of the human body is exploited for baseline comparisons.
3. **Tomography** : Tomography uses transaxial slices to view thin sections of the body. The complete picture is given by the composition of a set of slices.

In these situations, a single voice segment refers collectively to a set of displays. Because there are limited number of image screens, the set appears as a sequence of displays. This scenario contrasts with the situation in Scenario 3 which required contextual segmentation : that voice segment referred first to one display and then to another display.

Figure 6-3 shows the time line expected for an image sequence. To use time progression as an illustration, let :

Voice = "Notice how the infection has spread into the lungs over the last three days".

Load 1 = Frontal view of the chest, taken today.

Load 2 = Frontal view of the chest, taken yesterday.

Load 3 = Frontal view of the chest, taken the day before.

A mechanism is required to associate the LOAD activities with the PLAY activity and to sequence through the LOAD activities.

We introduce another group of activities called the *external activities*. External activities will be used to control the flow of a document presentation externally. The activities will serve as synchronization points for the global display, much as do audio activities.

Figure 6-3 A Timeline For an Image Sequence

Temporal Interaction with Multiple Image Screens

To handle a system with multiple image screens, two types of temporal interactions must be accounted for : the interaction between the audio and visual activities, and the interaction between the visual activities on the multiple screens.

Audio activities now occur in relation to a *global display* which is the combination of the visual activities for all image screens. This case is a simple extension of the single screen scenario. The temporal interaction is now given by the voice segment's temporal dependency on the last visual activity on each image screen.

The temporal interaction between the visual activities on the multiple screens is given by a temporal dependency on a common audio activity. Essentially, each voice segment acts as a synchronizing point for the multiple image screens. An exact record of the progress through sequences of intermediate visual activities on the multiple screens is unimportant as long as the final visual activities are all synchronized with the next voice segment.

6.1.4 Summary of the Temporal Behaviour Analysis

To conclude, the major findings of the temporal behaviour analysis are briefly restated.

1. There are two types of activities : persistent and ephemeral. Persistent activities are comprised of an action and an effect. Ephemeral activities consist only of an action.
2. The visual and session activities are persistent activities with the following properties : sequential, context dependent, time-invariant, and random durations.
3. The PLAY activity is an ephemeral activity with the following properties : sequential, context independent, time-variant and fixed duration.
4. A set of external activities must be defined to provide the external flow control required by complex reporting scenarios.
5. There are several complex temporal interactions that must be represented :
 - a) Voice playback with respect to either actions or effects.
 - b) Don't care situations.
 - c) Contextual segmentation.
 - d) Image Sequences.

6.2 A MRR Temporal Semantic Network

The proposed formalism for the temporal representation of a MRR combines semantic networks and interval-based temporal logic. A general *temporal semantic network* (TSN) consists of a set of nodes, interconnected by directed two-pole arcs. The nodes symbolize time intervals. A labelled arc joining a pair of nodes describes the constraints on the temporal interaction between the two time intervals.

The general concept of a TSN is specialized to a domain by associating primitive behaviours with the nodes' time intervals, and by providing a set of valid temporal constraints and a network architecture. The specialization of the TSN to multimedia radiological reports is discussed in three sections. The node semantics define the set of primitive behaviours peculiar to MRRs; the arc semantics, the temporal predicates needed to describe all expected temporal interactions. Finally, the network semantics describe how the MRR TSN as a whole is formed from the parts.

6.2.1 The Node Semantics

The node semantics identify the primitive behaviours common to all MRRs. In the previous section, the temporal behaviour of MRRs was decomposed into primitive activities, which were further divided into actions and effects. These actions and effects thus constitute the primitive behaviours of a MRR. Following from this, a node in a MRR TSN corresponds to an activity descriptor for either an action or effect - Activity (Interval, Context). Table 6.1 lists the set of nodes that may comprise a MRR TSN.

The Visual Nodes

The visual action nodes correspond to the actions of the visual activities identified in the temporal behavioural analysis (Refer to Section 6.1.1). The image processing actions (ZOOM, CONTRAST, TOGGLE, MARKER, LENGTH and ANGLE) are additive : they add some feature to an already existing display. The LOAD and RESTORE actions are terminating actions : the existing display is destroyed and completely replaced. This importance of this difference lies in the temporal relations that will be used to relate visual action nodes.

Node ID	Group	Type	Attributes	Description
Load	visual	action	Image File	Clears the image screen and loads the Image File into image memory for display.
Restore	visual	action	Image File	Removes all image operations on display, leaving only the original image.
Zoom	visual	action	Zoom Factor, Window Position.	Magnifies a portion of the current image display.
Contrast	visual	action	Contrast Algorithm, Window Position	Performs a contrast enhancement on a portion of the current image display.
Toggle	visual	action	None	Toggles the contrast of the image display (from positive to negative, and back).
Length	visual	action	Length, Line Coordinates	Draws a line and labels it with the Length.
Angle	visual	action	Angle, Angle Coordinates	Draws an angle and labels it with the Angle.
Marker	visual	action	Point Coordinate	Draws a marker at the given position.
Display	visual	effect	Action's Node ID	Is the display of the associated Action.

Table 6.1 : Nodes for a MRR Temporal Semantic Network

Node ID	Group	Type	Attributes	Description
Connect	audio	action	Voice File	Establishes a voice session and opens the Voice File.
Connect*	audio	effect	Connect Node ID	Is the voice session established by the given Connect Action.
Dis-connect	audio	action	Voice File	Closes the session of the given Voice File.
Play	audio	action	Position, Duration	Plays back from the currently open voice file the voice segment defined by the position and duration.
Pause	external	action	None	Freezes the global display for a default period of time.
Begin	external	action	None	A dummy node to start the MRR TSN.
End	external	action	None	A dummy node to terminate the MRR TSN.

Table 6.1 : Continued

Figure 6-4 Coupling of Visual Actions with DISPLAY Nodes.

We note that there is only a single visual effect : DISPLAY. DISPLAY represents the effect of the visual action given as its parameter. In the TSN, each visual action node will thus be paired with a DISPLAY node, mimicking the doublet nature of the activity descriptors. See Figure 6-4.

The Audio Nodes

The audio nodes are self-explanatory except for the concept of a voice segment used by the PLAY node. The voice examination report is made up of a series of physical voice segments. Physical voice segments are actual storage partitions caused by pauses during the report dictation. Physical voice segments may be subdivided into logical voice segments to represent contextual segmentation (Refer to Section 6.1.3). Logical voice segments represent points at which the playback process should pause a physical voice segment and artificially synchronize

the audio with some other activity. To the playback process, physical and logical voice segments function identically. They both define a non-decomposable length of voice, non-decomposable in that synchronization with a point within the length is not possible. Physical voice segments are identified automatically. The choice of logical voice segments is an issue to be dealt with by the creation process.

The External Nodes

The PAUSE activity is used to halt the presentation, keeping the global display static, for a specified (default) length of time. The PAUSE activity is considered to be sequential, ephemeral, context dependent, time-invariant and of fixed duration. Thus, the PAUSE node is an action.

The other two external nodes - BEGIN and END - are dummy nodes used simply to delineate the endpoints of the temporal semantic network.

The nodes shall be represented as frames. A frame hierarchy, shown in Figure 6-5, contains node templates which are instantiated to form the nodes in the actual MRR TSNs. The root frame has slots common to all nodes : a label that uniquely identifies a node instance. A "value" slot identifying the behaviour associated with the node - corresponding to one of the given Node IDs - is also common to all node frames. In addition, the node frames have slots corresponding to the node-dependent attributes listed in Table 6.1.

Figure 6-5 A Node Frame Hierarchy

6.2.2 The Arc Semantics

An arc specifies a temporal ordering between the behaviours associated with the two linked nodes, or more precisely, between the time intervals over which these behaviours occur. For a MRR TSN, the arc semantics define a set of arcs - called temporal predicates - that can describe any potential temporal ordering of a MRR's primitive behaviours.

The selection of temporal predicates began with Allen's set of thirteen primitive temporal relations (Table 4.1). The primitive relations describe fixed temporal interactions, those in which the comparative positions of all four endpoints are known. For example, as shown in Figure 6-6, all visual actions culminate in displays which begin as they finish. The ordering is fixed and can be described as :

action *meets* display

Figure 6-6 The Timeline for All Visual Activities

There are scenarios, however, which require a less restrictive description for which defined temporal relations are useful. "Don't care" interactions aim to leave the ordering flexible. Also, all visual behaviour components are of random duration, making it impossible to specify the relative positions of their endpoints.

The selection of the defined predicates is guided by three criteria which are best explained by example, using two sample sessions. Both sessions begin with a voice segment being played during the display, on two screens, of Image X and Image Y. The first session continues by loading an entirely different image on Image X's screen, whereas the second session continues with a magnification operation performed on Image X.

First Criteria : The defined temporal predicates should be expressive enough to handle the variety of interactions possible. For instance, to completely describe the temporal relation of the voice segment to the display of Image X, both endpoints (start and end) of each behaviour must be considered. Figure 6-7 shows the possible four cases in the first session.

- Start : Image X may be the slowest or the only image to be loaded; ie. Image Y is ready and waiting. As soon as Image X is ready, voice playback will be commenced (Figure 6-7a).

Conversely, Image Y may be the slowest or the only image to be loaded; ie. Image X is ready and waiting. Here, the voice playback cannot begin at the same time as Image X's display; it will begin sometime later (Figure 6-7b).

End : The playback process may hold the display until some minimum display period has passed. If the voice segment is very short, it will terminate before the display of Image X is destroyed by the subsequent load (Figure 6-7c).

Conversely, the voice segment may be long enough to let the presentation continue. The display of Image X ends with the voice (Figure 6-7d).

Combining the four possibilities in Figure 6-7 yields a defined temporal predicate :

while = starts or finishes or during or equals.

The description of the first session is then :

PLAY *while* DISPLAY (Image X).

LOAD Image X DISPLAY Image X

PLAY

Figure 6-7(a) Relative Start Points : Case 1

LOAD Image X DISPLAY Image X

PLAY

Figure 6-7(b) Relative Start Points : Case 2

..... DISPLAY Image X LOAD Next

..... PLAY

Figure 6-7(c) Relative End Points : Case 1

..... DISPLAY Image X LOAD Next

..... PLAY

Figure 6-7(d) Relative End Points : Case 2

**Figure 6-7 The Possible Temporal Relations Between a
PLAY and DISPLAY Nodes.**

Second Criteria : The defined temporal predicate should be accurate and unambiguous. Consider now the second sample session where the voice playback is followed by a magnification of Image X. The ZOOM action does not destroy the display of Image X, as did the LOAD action in the first session; instead, the ZOOM action is nested within it. Here

PLAY *while* DISPLAY (Image X)

is not correct. Since the voice segment is played before the zoom, it could never terminate with the image's display. In other words, Figure 6-7d does not hold for the second session. It follows that the voice cannot *finishes* or *equal* the display of Image X.

Third Criteria : The set of defined temporal predicates should be kept to a minimal size. To illustrate, we could solve the problem of the previous paragraph by introducing a new predicate :

ends_while = *starts* or *during*.

While would describe the first session, *ends_while* the second. This approach leads to an unnecessary expansion of the set. Instead, conjunctions of predicates will be used to refine the precision of the temporal descriptions. In terms of the semantic network, conjunctions correspond to a node having more than one arc. To illustrate, a second defined temporal predicate is introduced :

precedes = *before* or *meets*.

The second session is now correctly described as :

play voice *while* display and

play voice *precedes* zoom

Table 6.2 lists the set of temporal predicates chosen. The particular choices were simply a question of suitability to the application, convenience and manageability. With experience, the set may evolve to include others that more conveniently describe commonly occurring behaviours. In Section 6.3, we illustrate more fully the use of the temporal predicates for describing MRRs.

Relation (I,J)	Definition	Illustration	Inverse (J,I)
before	Primitive	III JJJ	after
meets	Primitive	JJJJ IIIII	met_by
contains	Primitive	JJJ IIIIIII	during
precedes	before or meets	JJJJ III or IIIII	follows
begins	starts or equals or started_by	JJJJJJ III or IIIIII or IIIIIIII	begins
starts_during	during or finishes or overlapped_by	JJJJJJ III or IIIII or IIIIIIII	starts_before
while	starts or during or finishes or equals	JJJJJJ III or III or IIIII or IIIIII	encompasses
concurrently	starts_during or starts_before or begins		concurrently
independent	"OR" of all primitives		independent

Table 6.2 Temporal Predicates for a MRR Temporal Semantic Network

6.2.3 The Network Semantics

The network semantics set out how the network as a whole is to be built and interpreted. Issues such as the the network's organization and the degree of connectedness are addressed.

The MRR TSN is organized into a set of $n+1$ subnets, n *screen subnets* and one *annotation subnet* (See Figure 6-8). Each subnet describes sequential temporal behaviour only. Arcs joining nodes from two subnets represent synchronization requirements for concurrent behaviour.

The organization is based on the idea that the MRR presentation resembles a group of cooperating sequential processes. For instance, on a single image screen, updates are performed sequentially. Assuming that multiple image screens are identical, the screens can be modelled as n sequential processes; hence, the n screen subnets. A final annotation process acts as a centralized master for the n screen processes, providing the synchronization requirements needed to coordinate the annotation of the n screen displays. The annotation subnet thus contains all nodes that describe behaviours that require/impose a certain timing between the screen subnets. In our current model, the annotation subnet includes the audio and external nodes.

2

OF/DE

2

Figure 6-8 The Logical Organization of a Temporal Semantic Network

Allen's approach to maintaining temporal relations [Allen83b] used a fully connected and reduced network. Such a complete specification is not necessary for the description of the animation requirements. Instead, the nodes in a MRR TSN are connected according to three guidelines :

1. Because the behaviour is sequential within a subnet, the nodes are simply chained together, with each node related only to its predecessor and its successor.
2. Each node in the annotation subnet represents a synchronization point for the global display : either the playback of a voice segment or a pause period. Each annotation node is connected to at least one node from each screen subnet.
3. The screen subnets are not linked directly. Instead, the temporal interaction between multiple screens is given indirectly through links to a common annotation node. The intent of this "least commitment" strategy is to encourage parallelism in which each screen proceeds independently to the next synchronization point. The result will be a more efficient update procedure.

6.3 Sample Multimedia Radiological Reports

In the next five sections, the use of the temporal semantic net will be demonstrated for each of the scenarios described in Section 6.1.3.

6.3.1 The MRR TSN for Voice Playback with a Fixed Display

The MRR TSN in Figure 6-9 describes the playback of a voice segment during the screen display given in Figure 6-1. Because there is only a single screen, the MRR TSN consists of the annotation subnet (on the left) and one screen subnet (on the right). This architecture will be common to all the single screen scenarios.

The complete form of a MRR TSN is drawn in this sample report. The MRR TSN as a whole is delimited by a **begin** and **end** node which relates to the first and last nodes in each of the subnets, respectively. Also, the annotation subnet contains session activities : connecting and disconnecting the voice session. In the remaining reports, these nodes are omitted; only the portions of the MRR TSN relevant to the scenario at hand are drawn.

Figure 6-9 A MRR TSN for Voice Playback with a Fixed Screen Display.

For this scenario, the relevant portion is comprised of the remaining nodes. The voice segment's association with a screen display is given by "PLAY *while* DISPLAY". *while* means that the voice segment may be played at any time during the display of the MARKER. By backtracking along the screen subnet, we see that the display of the MARKER, in turn, requires that an image first be loaded and then magnified.

It is clear from this sample report that the temporal behaviour is incompletely specified, both within and between the subnets. In the annotation subnet, the DISCONNECT node is not related to the CONNECT node, even though DISCONNECT can only occur if a voice session is connected. In the screen subnet, the nesting of displays is not explicitly noted. Between the subnets, the annotation subnet is linked to the screen subnet only at the critical time point; in this case, to the screen node that describes the completed display. Temporal relations are specified only to the degree needed by the playback process. They are sufficient, however, to infer the other indirect temporal relations with the appropriate application of domain rules and the transitivity property.

6.3.2 The MRR TSN for Voice Playback with a Changing Display

Voice playback with a changing display implies that the audio behaviour is temporally dependent on a visual action. The MRR TSN in Figure 6-10 describes the playback of a voice segment while a new image is being loaded.

Flexibility in the temporal ordering of end points is needed in this scenario. The defined temporal predicate *starts_during* states only that the playback of the voice segment may begin sometime during the image load. Because the duration of the image load is variable, it does not specify when the voice segment will end with respect to the load.

Note that there is an implicit temporal relation between the **PLAY** and the subsequent **DISPLAY** node. As stated in the previous scenario, the playback process could infer this relation using domain rules and the transitivity property of temporal relations.

Figure 6-10 The MRR TSN For Voice Playback with a Changing Screen Display.

6.3.3 A MRR TSN for Contextual Segmentation and Don't Care Situations

Figure 6-11 shows the MRR TSN for two examples of contextual segmentation. Two physical voice segments (1 and 2) are each divided into two logical voice segments.

The first physical voice segment is the example used in Scenario 3. It consists of a report header which can be played at any time plus a subsequent reference to a particular image. The first physical voice segment is thus a combination of a don't care situation and a voice playback with a fixed display.

The second physical voice segment contains two comments, each referring to a separate screen display. It is thus represented as two successive voice playbacks with a fixed display.

Figure 6-11 A MRR TSN For Contextual Segmentation and Don't Care Situations.

6.3.4. A MRR TSN for an Image Sequence

The MRR TSN in Figure 6-12 describes the time progression example used to illustrate image sequences in Section 6.1.3 : where three images show the progression of a disease. This MRR TSN states that the voice segment is to be played sometime during or after the load of the first image. The first image is then displayed for some minimum time. The next annotation node is the first pause node which forces the second image to be loaded. Because of the *while* relation, the first pause node holds the screen display static for some default time, at which time the presentation moves to the next annotation node - the second pause node. In a similar fashion, this second pause causes the third image to be loaded and displayed for (at least, depending on the next annotation node) some default time.

Figure 6-12 A MRR TSN For an Image Sequence.

6.3.5 A MRR TSN for Multiple Screens

Figure 6-13 shows a MRR TSN in which two image screens are used : Screen 1 on the far left and Screen 2 on the far right. Here, the MRR TSN is made up of an annotation subnet (in the centre) and two screen subnets (on either side). Each annotation node is related to nodes in each of the (two) screen subnets.

In this example, two voice segments are played. The first voice segment is played in conjunction with a fixed global display, indicated by the *while* arcs extending to the two DISPLAY nodes. The first voice segment is additionally constrained to occur before the TOGGLE operation. The second voice segment is also played during a fixed global display. The same display on Screen 1 is used but the contrast on Screen 2 must be reversed before playback of the second voice segment commences.

Figure 6-13 A MRR TSN For Voice Playback with Multiple Screens

Chapter 7 AN AUTOMATIC MULTIMEDIA RADIOLOGICAL REPORT FACILITY

A MRR facility is a workstation tool for the creation and playback of multimedia radiological reports. An automatic MRR facility performs these two functions automatically - with a minimum of user input - effectively performing a two-step transformation of the radiologist's reporting session into a multimedia presentation. As shown in Figure 7-1, this transformation can be regarded as two independent processes operating in tandem with the MRR serving as the communications medium between the creation and playback processes.

In this chapter, the design of an automatic MRR facility is discussed. The playback and creation processes are considered separately. For both, their use of the proposed representation for MRRs is examined, leading to a requirements specification for the process. A design for the process is then proposed.

Figure 7-1 An Input/Output Model of an Automatic MRR Facility

7.1 The Automatic Playback Process

According to Figure 7-1, the playback process automatically prepares a multimedia presentation by manipulating the voice system and the n image screens according to the temporal specifications given by the MRR. The playback process uses the MRR in two manners which we have called operating modes : sequential execution mode and interrupt execution mode.

Sequential Execution Mode

During sequential execution, the playback process performs the unprompted and unaided generation of a MRR presentation. Put simply, the presentation is generated by an orderly forward traversal of the MRR TSN. A central traversal down the annotation subnet controls the flow of the presentation. Passage to each successive annotation node - where passage to a node commences the behaviour associated with that node - must be synchronized with parallel traversals down each of the n screen subnets. The synchronization requirements are given by the temporal relations extending from the annotation node to the nodes in the screen subnets.

Traversal of each single subnet - annotation or screen - is simple because it is strictly sequential. The coordination of the joint traversal is complex, especially when there are multiple screens. Difficulties arise in the definition of when the synchronization requirements for an annotation node are met. In particular, concurrent temporal relations must be defined with respect to actions and/or effects. To explain, consider the two sample MRR TSNs shown in Figure 7-2, each of which contain two screen subnets. Let a vector describe locations within the TSNs :

$$\text{Location} = (L1, Ls, L2)$$

where $L1$ = Location in Screen Subnet 1
 $L2$ = Location in Screen Subnet 2
 Ls = Location in Annotation Subnet

Figure 7-2(a) The First Sample MRR

Figure 7-2(b) The Second Sample MRR

Figure 7-2 Two Sample MRR TSNs

The first sample MRR TSN in Figure 7-2(a) describes the playback of two voice segments, each in conjunction with a fixed global display; in other words, the two annotation nodes are temporally related to visual effects. Assume that the first voice segment has just been played, giving an initial position of (DISPLAY 1A, PLAY 1, DISPLAY 2A). The next annotation node is the second voice segment (PLAY 2) which must be played against the display of two zooms on Screen 1 and a marker on Screen 2. Assuming only forward motion during sequential execution, the playback process will move forward along both of the two screen subnets - performing the indicated screen updates - until the required displays are prepared. The movements along the two screen subnets will not proceed at the same rate since the amount of work involved in the updates to each screen will vary, both in amount and in complexity. For example, in this case, two updates must be made to Screen 1 compared to one on Screen 2; as well, the zooms on Screen 1 require network transfers to retrieve the zoom files while the marker on Screen 2 is a simple workstation function. Because the display of the marker on Screen 2 will persist and because the voice segment can be played anytime during the display of the marker ("PLAY 2 *while* DISPLAY 2B"), the update to Screen 2 can be allowed to complete well ahead of those to Screen 1. Concurrency with effects is thus a loose coupling because of the persistence property of effects.

The second MRR TSN in Figure 7-2(b) is identical to the first except that the second voice segment must now be played in conjunction with the action of putting up the marker on Screen 2 : PLAY 2 and MARKER must be done concurrently. Because the act of putting up the marker may take a very short time, no movement along Screen 2's subnet can be made until the updates to Screen 1 are completed. Concurrency with actions thus requires tightly controlled synchronization due to the random durations of actions.

Three issues must be considered in an implementation of the sequential execution mode for the playback process :

1. An efficient algorithm to coordinate the joint traversal of the MRR TSN : The interpretations of the temporal relations, the interactions between multiple temporal relations (especially in the case of multiple screens where an annotation node is related to at least one node from each screen subnet), and the difference between actions and effects must be considered.
2. A software structure to use the parallelism available in the MRR TSN : Concurrent, independent movement down the multiple screen subnets is desirable.

3. Global display time : A means to control the speed of the presentation is needed. A proposed *global display minimum* would set a minimum time for which a global display must remain static before the next annotation node can be approached. The minimum will help in cases where the voice segments are so short that a user could not absorb the display before it was updated. To avoid excessive delays, the minimum could also be used in conjunction with a "Next" button that the user pushes when he is ready to proceed.

Interrupt Execution Mode

The interrupt execution mode is akin to a supervisory state or to an overlying operating system. In this mode, the playback process handles exceptions and user interactions, both of which are situations that deviate from the automatic presentation and that require knowledge about how to proceed. Report management is also done in this mode.

Report management looks after the startup and shutdown of a MRR report. Startup includes retrieving the MRR and establishing the voice session. Shutdown consists of closing the MRR and terminating the voice session.

An exception is any unexpected behaviour that interferes with the MRR presentation; for example, a required file not being available. The playback process must decide how to recover from the exception and continue on with the remainder of the MRR presentation.

User interaction allows the user to halt the automatic presentation; for example, to pause temporarily, to explore something on his own, or to re-position the playback (wind/rewind). The audio playback can be halted immediately with playback commands sent to the Voice Message System. Any visual actions currently in progress, however, will have to be let run to completion. The state of the presentation (given by the current location vector) must be stored, as well as any changes made by the user (ie. any screen updates or (re)winding of the voice report). With this information, the playback process can plan how to resume the automatic presentation of the MRR.

The two operating modes highlight the dual nature of the automatic playback process. Automatic presentation is a deterministic, algorithmic procedure. On the other hand, handling exceptions and user interaction are suited to a flexible knowledge based approach. The requirements of both are met in the design proposal outlined in the following subsection. A second subsection considers the testing requirements for the playback process.

7.1.1 A Design for the Playback Process

This section details a software design for the automatic playback process. This process automatically presents MRRs by controlling the MUSIC workstation in accordance with the behaviour described by the MRRs. It is modelled as a single dispatcher task that sends commands to the workstation's (image screen and voice board) device drivers. A testbed system models these device drivers as server tasks. This testbed is first described, followed by a discussion of the software design for the playback process and the issues encountered.

The design goal of the testbed was to copy the communication paths of the playback process with the MUSIC workstation hardware so that the testbed software could ultimately be replaced by the workstation's actual device drivers. The testbed simulates the MUSIC workstation with a MicroVax workstation using a set of UIS windows, as shown in Figure 7-3. Two image screens are assumed, given by two "image screen" windows. A third window acts as the voice board, with the periodic display of text serving as voice playback.

Figure 7-3 A Simulation of the MUSIC Workstation Using Windows.

Figure 7-3 is the software structure diagram for the testbed software. Two Image Driver tasks control the two "image screens" through a set of routines corresponding to the set of functions available on the real MUSIC workstation : magnification, contrast enhancements, etc.. The Voice Driver task drives the "voice board" by writing the next sentence of a voice report to the voice window at regular time intervals. All Driver tasks - voice and image - loop infinitely : waiting on their own mailboxes to receive a command, performing the command, and returning a "command completion notification" to the playback process via a central system mailbox.

Figure 7-4

The Software Structure Diagram for the
MUSIC Testbed Simulation

The design of the playback process is based on the state machine given in Figure 7-5. The states correspond to the two operating modes introduced earlier, and the transitions outline the execution path of the playback process during the presentation of a MRR. The design details for the two states are given in the following paragraphs.

Figure 7-5 **The Playback Modes**

Sequential Execution Mode

The automatic generation of a MRR presentation has been described as the orderly traversal of the MRR TSN. In this section, an algorithm is introduced which models the network traversal as a form of forward chaining. The network traversal algorithm, presented in Figure 7-6, allows parallel and independent updates of multiple screens, can provide a minimum global display, and handles voice playback in parallel with screen updates. The explanation of the algorithm proceeds in general terms. In addition, for each major point, a reference is given. The references are contained in Appendix A which contains a bare-bones version of the algorithm, coded in MPROLOG. Each reference is a (group of) predicate(s), intended to demonstrate the basic logic structures required.

The network traversal algorithm in Figure 7-6 performs temporal forward chaining. The annotation nodes represent the key steps in the MRR presentation. To generate the next step, the goal of the playback process is to pass to - and enact the behaviour of - the next annotation node. To do so, the temporal conditions for this annotation node, denoted by the arcs extending from the annotation node to the screen subnets, must be satisfied. To satisfy these conditions, the playback process simultaneously and repeatedly moves forward along each of the image subnets, triggering screen updates as it advances to each new screen node, until the temporal conditions are met. At this point, the behaviour of the annotation node can commence. (Reference : *present_report* and *temporal_fwd_chain*).

```

(1)  WHILE the MRR presentation is not finished LOOP
(2)      Find the next annotation node and its temporal relations to all screen
          subnets.
(3)      IF all the screens are not "synchronized" THEN
(4)          FOR each "out-of-sync" screen DO
(5)              Advance to the next screen node and initiate the
                  behaviour associated with this node by sending the
                  appropriate command to the corresponding screen
                  driver.
          END DO
(6)          REPEAT
(7)              WAIT for a completion notification from a screen
                  driver.
(8)              Update the position in this screen's subnet.
(9)              IF this screen is still "out-of-sync"
                  THEN
(10)                 Advance to the next screen node and send the
                        appropriate command to this screen's driver.
                  END IF
(11)          UNTIL all screens are "synchronized".
          END IF
(12)      Initiate the behaviour associated with the annotation node, in
          conjunction with any concurrent visual actions.
(13)      Wait for the completion notifications from (12).
(14)      If necessary, wait until the global display associated with this
          annotation node has been present for some minimum time.

      END WHILE

```

Figure 7-6 An Algorithm for Sequential Execution

Two types of temporal conditions are distinguished to reflect the different synchronization requirements of effects and actions, as discussed earlier : preliminary and concurrent. Preliminary temporal conditions describe screen behaviours that must occur before the annotation node can be traversed. Concurrent temporal conditions describe screen behaviours that occur in parallel with the annotation node's behaviour.

Lines 3-11 of Figure 7-6 fulfill the *preliminary temporal conditions*. These are met by forcing the sequential traversal of all "out-of-sync" screen subnets until the image screens are all "synchronized". A node in a screen subnet is traversed by sending a command to the corresponding screen driver to perform the screen update associated with this node, and then waiting for the completion notification. To traverse multiple screen subnets concurrently, the playback process first initiates all possible screen updates, and then waits on a single system mailbox. When a completion notification is received - with its source identified as part of the message - the playback process initiates further updates to that screen, if necessary, before returning to wait for the remaining outstanding completion notifications. The procedure is repeated until all completion notifications are received and no further update on any screen can be made. (Reference : *begin_annotation*).

The important question here is : when is a screen "synchronized", or equivalently, what is a preliminary condition ? Rules, such as those in Figure 7-7, will define the criteria for "synchronized" screens (Reference : *synchronized*). In defining the situations under which a given temporal condition holds, the rules use certain assumptions about :

- 1) The behaviour of actions and effects, especially the persistence property of effects.
- 2) The use of temporal relations only in certain situations. For example, *while* is used only to relate an annotation node to a DISPLAY node. *while* cannot be used to relate an annotation node to a visual action node because its definition includes the specification of end points (which is not possible due to the random durations of visual actions).

The implication is that the creation process must build the MRR TSNs in accordance with these rules. Certainly, this raises an ideological question of whether the definition of the temporal relations should be independent of the application. Doing so, however, proved overly complex and was abandoned for this trial system.


```

IF    the visual node related to the synchronizer
      node is an action      AND
      the relation describes some kind of parallel
      behaviour (ie. starts_during, begins,
      concurrently)
THEN
      this image screen is complete when the
      subnet position is at the DISPLAY node
      just preceding this visual node.

IF    the visual node related to the synchronizer
      node is a DISPLAY AND
      the relation is while
THEN
      this image screen is complete when the
      subnet position is at this DISPLAY node.

```

Figure 7-7 **Sample Rules Defining "Synchronized" Image Screens**

The *concurrent temporal conditions* are satisfied by Lines 12 and 13. The proper ordering of the annotation behaviour with concurrent visual actions is achieved by ordering the temporal relations into a precedence priority. For instance, suppose that the annotation node X had two concurrent conditions :

X starts_during Y and X begins Z

where Y and Z are nodes in two different screen subnets. *Starts_during* is given precedence priority over: *begins*.

With (*X starts_during Y*), X must start before Y.

With (*X begins Z*), X and Z start at the same time.

Y must be started first, thereby satisfying the *starts_during* condition. Then, the *begins* condition can be met, by starting X and Z together. The playback process then waits for all three completion notifications. Because of the small number of concurrent temporal relations used, using precedence priorities is possible even when dealing with multiple screens. (Reference : *schedule_annotation*, *high_priority_concurrencies* and *low_priority_concurrencies*).

The final line (Line 14) in the traversal algorithm ensures that the global display remains static for at least some minimum time period before the next annotation node is considered. The receipt of the last completion notification from a screen driver task signals the commencement of the global display. (Reference : *minimum_global_display*).

Other than completion of the MRR presentation, this traversal algorithm is exited when either an exception or a user interaction occurs (Refer to Figure 7-5). Both occur as an interrupt to the playback process, causing it to store the current MRR TSN position and change to interrupt execution mode.

Interrupt Execution Mode

A rule-based planner is proposed for the implementation of the interrupt execution mode. In this mode, the chief duty of the playback process is to plan for the resumption of sequential execution, beginning at some starting location in the MRR TSN. This starting location¹ assumes that a certain global display exists. The goal of the playback process is, therefore, to ensure that the current global display is *compatible* with the assumed global display. Compatibility is defined by rules, and does not necessarily require equality. For instance, if the user simply adjusted the contrast on the image monitor, a rule will dictate whether the original contrast must be restored or if it can be left as is. These rules will be used to guide the comparison of the current and assumed global displays. Because a global display can be defined by the sequence of visual actions used to build it, the comparison will be done by retracing the history of the assumed global display as given in the MRR TSN, beginning at the starting location. The comparisons will dictate whether the assumed display must be built from scratch or by upgrading the current display.

To conclude the software design, MPROLOG is suggested as the implementation language of the playback process. This logic programming language is suitable both for the forward chainer of the sequential execution mode, and for the backward chaining planner of the interrupt execution mode.

¹ The starting location will either be the position stored by the playback process when sequential execution was interrupted, or some new position if the user has (re)wound the voice report.

7.1.2 Testing of the Playback Process

The ultimate goal for testing the playback process is to establish that the presentation automatically generated by the playback process is indeed that described by the MRR TSN. The areas to be tested are first outlined. Suggestions regarding the test procedures and tools are then given.

Testing must demonstrate that the playback process makes proper use of the temporal semantic network. With regard to the network traversal algorithm, this involves the following issues :

- 1) **Completeness :** Handling of all possible scenarios, ie. those introduced in Chapter 6.
- 2) **Correctness :** Correct interpretation of the scheduling requirements for each of the temporal predicates, including interactions between multiple temporal conditions and multiple image screens.

3) Performance under variable timing :

In a MRR TSN, defined temporal predicates express uncertain synchronization requirements due to the random nature of actions (such as loading images). The defined temporal predicates actually represent two or more possible timings, each one of which must be tested. For instance, consider the case given by *PLAY starts_during LOAD*.

As shown, *starts_during* covers three possibilities : the playback of the voice segment ends before, at the same time, or after the loading of the image. To demonstrate the proper interpretation of defined predicates by the playback process, its operation under all three cases must be tested.

Figure 7-8 Testing of the Defined Predicate *starts_during*.

Thorough and successful testing requires organization, repeatability and documentation. To this end, two testing tools are suggested. A trace facility will record the progress of the testing. Script command files will drive the testing.

A trace facility for the logging of events during the presentation will provide a documented time history of each presentation. A general trace facility, usable by other software packages such as the creation process, is suggested. To this end, hooks should be inserted in the device drivers such that, as a driver receives each command, it logs the time of activation and completion. For instance, if a load command is issued to a screen driver the time at which the load is begun and when it is finished will be recorded. By putting the hooks in the device drivers, it does not matter whether the commands are issued by a user at the workstation by clicking on the mouse, or by the playback process, or even by the creation process. A mapping can then be made to interpret the time history in terms of the MRR TSN. For instance, the completion of a load terminates the time interval associated with a LOAD node, but it also commences the time interval associated by the paired DISPLAY node.

Basically, script command files are input files, used to provide values which vary from one test to another. They not only offer repeatability but they are also a natural format for documenting the test procedures. For the playback process, a script command file will be used to simulate variable performance in the

workstation. A file will contain delays for each action node in the MRR TSN being tested. The delays will be used to request timeouts, to simulate a delay in performing the action. Only when this timeout has expired is the action considered to be completed. In this way, the testbed system operates as if network delays are incurred.

For a particular MRR TSN, a set of script commands must be generated, representing all the test scenarios for that TSN. For instance, to test the network fragment shown in Figure 7-9, three script files are needed, corresponding to the three possible timings accounted for by *starts_during*. The first script forces the LOAD to end after the voice segment, the second at the same time, and the third before the voice segment.

Figure 7-9 Command Script Files for a Network Fragment

To test the automatic playback of MRRs, a number of sample MRR TSNs must be manually created. Within the screen subnets, various combinations of actions should be tried; in particular, the nesting of image processing activities, and the interaction of LOADs/RETURNS with the image processing activities. Between subnets, the set should, at the minimum, include the five scenarios described in Chapter 6. For each MRR TSN, a set of script command files must be generated. The scripts should provide input values to test the variable timings of each defined temporal predicate in the MRR TSN. When used in conjunction with the trace facility, the script command files will help to partially automate the test procedures, and provide results which are completely documented and available for manual examination and verification.

7.2 The Creation Process

The creation process automatically produces a multimedia radiological report based on the reporting session. It is visualized as a two-stage procedure (Figure 7-10). A capture subprocess captures in real-time the sequence of the radiologist's reporting activities, recording the sequence as a time history. A generate subprocess is a post-processor that extracts the MRR from the time history.

The capture subprocess is a simple time-stamp generator. The difficult issue here is the selection of events to be captured. The time history must provide the cues to be used by the generate subprocess. Some cues will influence the user interface. For instance, in order to associate voice with images, it may be found necessary to impose the requirement on the radiologist that he explicitly refer to an image by clicking on it with the mouse during dictation. The selection of cues must, therefore, be made with care so as to not conflict with our primary goal, that of an unobtrusive and automatic report creation facility.

Figure 7-10 The Creation Process as a Two-Step Procedure.

To extract the MRR from the time history, the generate subprocess must be able to :

1. Identify irrelevant exploratory activities to be excluded from the MRR.
2. Identify and classify the temporal semantics of the relevant portions of the time history in terms of the known playback scenarios.

7.2.1 A Design for the Creation Process

The automatic creation process has not yet been fully investigated; however, this section discusses some thoughts regarding the software designs for the capture and generate subprocesses.

In its simplest form, the capture process will require that the device drivers log the time at which they perform each command when an examination is being reported. Whether all commands will be logged will depend on the selection of events to be used as cues by the generate subprocess. Actually, such a trace facility would also be useful in system testing and monitoring, and as a documentation tool for the performance testing of the automatic playback process.

For the generate subprocess, it appears that a production system would be an effective vehicle for its implementation. Its function is comparable to some medical and fault diagnostic systems implemented as production systems : to classify a situation in terms of a set of known situations. The knowledge bases in such systems contain IF-THEN rules which set out the criteria for classifying each type of situation. In the case of MRR generation, the situations are defined by temporal interactions between the cues provided by the capture subprocess. The set of known situations are those discussed in Chapter 6 (voice playback with fixed or changing screens, image sequences, don't care situations).

[Fikes85] suggests a control strategy for production systems that deal with the classification of situations. The *successive refinement strategy* is based on the notion that the classification of situations can be thought of as a class taxonomy, thereby suggesting the use of frames to organize the production rules and the reasoning processes.

First, a frame hierarchy is required to encode the class taxonomy. Each frame is a prototype for a class of situations. The slots of the frame specify the necessary conditions for membership in this class. In other words, each frame contains only the conditions needed to specialize its situation from the more general situation represented by its parent frame. A frame hierarchy that could be applied to the MRR generate subprocess is shown in Figure 7-11.

Figure 7-11 A Frame Hierarchy for the Successive Refinement Classifications of MRR Situations.

Second, the production rules must specify the sufficient conditions for class membership and will always be of the form :

IF situation is a member of Super_Class AND
 a set of local conditions are met

THEN

 situation is a member of this Class

The rules are thus organized by their association with a class frame. They indicate whether members of a superclass (the class represented by the class frame's parent) can be members of this class.

The successive refinement strategy uses the frame hierarchy as a discrimination net. Beginning at the root frame (ie. the most general case), the reasoning process moves down the frame hierarchy to lower and more specific classifications, using the groups of rules to guide its movements.

The use of frames in conjunction with rules provides a means of organizing and indexing a production system's rule base. Even if frames are not used per se in the generate subprocess, at the very least, the frame hierarchy acts as a design and documentation tool for its production rules.

The creation process is considered to be the most difficult part of the automatic MRR facility. In the next chapter, we will see how work in this area will proceed.

Chapter 8 CONCLUSIONS AND FUTURE RESEARCH

Multimedia radiological reports tie together radiographic images, image processing operations on these images, and voice annotation to produce an animated examination report. A temporal representation for MRRs was developed which supports the use of MRR over a distributed multimedia communications system. The use of this temporal representation for the manipulation of MRRs was then considered in a design proposal for an automatic MRR facility.

The representation introduced - a MRR temporal semantic network - captures the temporal behaviour of a MRR. A MRR temporal semantic network is a network of temporally ordered nodes, where each node is associated with some primitive component of the MRR presentation. A MRR TSN thus describes the synchronization requirements for the coordinated, animated presentation of the examination report.

The primitive behaviours from which the MRR TSNs are built were taken from a temporal analysis of radiological reporting. Two types of primitive behaviours are recognized : actions which are active and self-terminating; and effects which are passive and persistent, somewhat like a state. The effects are represented explicitly, rather than as a hidden state variable associated with node transitions, for the following three reasons :

1. The temporal behaviour being described by the MRR TSN is more transparent.
2. The level of inference is reduced by one because the state is given explicitly.
3. Effects also have temporal properties. With explicit representation, the expression of these temporal properties is consistent with those done for the actions. The MRR TSN completely and uniformly describes all aspects of temporal behaviour.

The design proposal for an automatic MRR facility covered the automatic creation and playback of MRRs. Automatic creation consists of the capture, interpretation and storage of the radiologist's reporting session. A simple time-stamp

generator *captures* the radiologist's reporting behaviour. A rule-based production system then operates on the time-stamp history, classifying the temporal semantics of the reporting session in term of a set of defined primitive behaviours, thereby *generating* the MRR. Automatic playback involves the orderly and algorithmic traversal of the MRR TSN, invoking the behaviour associated with each of the nodes in the temporal order described by the interconnecting arcs. Rule-based planning handles exceptions and user intervention.

A skeleton version of the sequential mode of the playback process was coded in MPROLOG, and is included in Appendix A. The program contains all the logic structures for the network traversal algorithm, but handles only the most common temporal predicates. The testbed described in Chapter 7 is ready for development purposes. The testbed was implemented on the MicroVax workstations at the University of Ottawa. All the functionality of a MUSIC workstation (eg. magnifications and contrast enhancements) was simulated with a set of windows, using the Digital UIS Windowing Utility to perform various operations on these windows. Access to the testbed by the automatic MRR facility is through mailboxes, managed through the VAX VMS system services. A MPROLOG language interface to both the FORTRAN UIS routines and the system services was coded.

Continuing work and future research

The automatic MRR facility remains to be built, starting with the playback process. Completion of the playback process will serve several purposes. First of all, the suitability of the TSN as a representation for a MRR will be demonstrated. The appropriateness of the chosen temporal relations can be tested. Secondly, the playback process will allow the evaluation of the MRRs through hospital field trials. The reactions of reviewing physicians to a multimedia radiological report as a useful alternative to the current practise can be gathered. In particular, the multimedia features which add value to the basic examination report can be identified. The emphasis here is on the opinions of the reviewing physicians. Finally, the playback process will facilitate the development of the capture process. It can serve as a knowledge acquisition tool for hospital trials geared towards the reporting radiologists. These trials will attempt to find the cues needed to capture a reporting session and the rules which will generate the MRR from the captured time history. The playback process can be used to present reports generated by these rules, thereby evaluating the rules and suggesting further refinements.

Future research will consider extensions to the MRR. Extensions will involve the addition of text, the inclusion of contextual dependencies, and the incorporation of the MRR into a multimedia patient file.

The Addition of Text

An unwritten assumption has been made that the examination report will be stored as a voice report. In keeping with present day practises, it may also be necessary to offer a text version. This is easily done without any modification to the MRR TSN and only adjustments to the playback process.

Because the text version is a simple transcription of the voice report, its content structure reflects that of the voice report : a group of text paragraphs corresponding to the group of voice segments. To effect the same multimedia presentation, the text paragraphs must share the temporal behaviour of the voice segments : sequential, fixed duration, time-variant, and ephemeral. Although text is persistent - several paragraphs may appear concurrently on a large text screen - it can be made to act ephemerally. A window can highlight the active paragraph - the paragraph corresponding to the voice segment for which the global display is defined. As playback continues, the window travels to the next paragraph. In this way, the same TSN can be used for either a voice or text examination report. The difference lies in the playback process, in the hardware devices which it manipulates.

The Inclusion of Contextual Dependencies

In the temporal behavioural analysis of a MRR in Section 6.1, the visual activities were found to be context dependent. In the MRR TSN, these contextual dependencies are minimally represented as a temporal dependency on some preceding behaviour; for example, all image processing operations act on a previously loaded image. Context, however, can act as a higher level description of a MRR that allows intelligent and automatic manipulation of the MRR.

For instance, another research area in the MUSIC project is investigating automatic contrast enhancements. The goal is to automatically optimize the contrast of a display. The contrast enhancement algorithm used, however, depends on the body area being displayed : whether it is a dark or light area; whether the edges or the area are of interest.

A second example of context is the use of symmetry in reading radiographs. There are different types of symmetry, however, depending on the body area under review. On a frontal chest view, if a zoom is performed on the left lung then left-right symmetry may dictate a similar zoom on the right lung on the same view as a baseline comparison. If the zoom was done on the heart then projection symmetry would use the lateral view of the heart to form a "pseudo" 3-D projection of the heart.

The definition of context varies among the visual activities. For the LOAD activity, context identifies the film image being loaded by a composite of the body area (arm, chest, brain, etc.) and the view (eg. frontal, lateral, occipital, etc.). "Chest frontal" is a sample context for a LOAD. For the image processing activities, the context is a body location (eg. fifth vertebrae or right lung), instead of a pixel location within the digital image.

Context can easily be included explicitly in the MRR representation as simply another slot - another piece of information - in the current node descriptors. Automatic identification of context by the creation process is a problem which must be addressed. The use of context by the playback process, however, is still an open area, with the two examples given above serving only as starting points.

A Multimedia Patient File

Finally, the inclusion of the MRR into the overall system must be considered with regard to its storage and access : The dynamic MRR representation must be combined with a static database representation. A second, higher level multimedia document - the multimedia patient file - will organize all radiological information pertaining to a single patient. It will form a multimedia filing system.

Ideas from Intermedia about multimedia databases may be applicable to multimedia patient files; in particular, the notion of multiple perspectives. Multiple perspectives provide different views of the same data. This would allow the use of data in more than one examination. For example, a film image from one examination may be used as a baseline to check the progress of a disease in a later examination. The different needs of various users could also be accommodated. Although both use chest films, a lung specialist looks at different areas - with different contrast characteristics - than a heart specialist. Similarly, an expert may want only a brief summary while a resident requires a detailed survey. Multiple perspectives enable the facility to be more responsive to the user.

By extending this line of thought, the multimedia radiological report becomes simply another perspective into a multimedia patient file : The MRR provides a temporal view of the radiological data. Our MRR representation must, therefore, be integrated into the model for the overall multimedia patient file.

As a first approximation, the multimedia patient file can be modelled by the Office Document Architecture model. The ODA model can provide the flexibility in content and structure needed by multimedia patient files. The basic logical structure for a multimedia patient file will probably resemble that pictured in Figure 8-1. This arrangement reflects the filing system currently in use in the Ottawa Civic Hospital, and is quite simple : all data relevant to an examination is stored in a single examination folder which, in turn, is stored in the appropriate patient folder.

Figure 8-1 A Logical Layout for the Multimedia Patient File

In conclusion, the development of a representation for MRRs marks only the beginning. The implementation of the MRR facility as proposed will allow an assessment of the given temporal representation, and provide the means to develop the research possibilities cited above.

APPENDIX A CODE FOR THE PLAYBACK PROCESS

This appendix contains the MPROLOG code listings for the sequential mode of the playback process. The program is a bare-bones version; in particular, error conditions have not been accounted for, and not all temporal predicates or interactions have been included. The logic of the program is complete, although not tested. Its inclusion is meant to demonstrate how the code for the network traversal algorithm (Figure 7-6) would likely be structured.

The organization of the program is mapped out below. Each heading represents a logical grouping of predicates. In general, the headings map to the major points brought out in the discussion of the network traversal algorithm in Section 7.1.1.

Module Interface

Contains references to routines used to send and receive commands to the device drivers.

Local Data Declarations

Local data structures, including the network location vector to record the position of the traversal within each subnet.

Main Logic of the Network Traversal Algorithm

The basic Interpreter for the Temporal Forward Chainer
(Lines 1-14)

Predicates for the Preliminary Temporal Conditions

Performs repeated and simultaneous traversals of the screen subnets to update out-of-sync screens.
(Lines 3-11)

Predicates for Defining Synchronized Screens**Predicates for the Concurrent Temporal Conditions**

Triggers the annotation behaviour in conjunction with any concurrent visual actions (ie. screen updates).
(Line 12-13)

Minimum Global Display

(Line 14)

Predicates for Movements within a Subnet

Sequential forward movement within a single annotation or screen subnet.

Predicates for Triggering Node Behaviours

Sends commands to the device drivers, receives completion notifications, and updates the subnet positions appropriately.

For each major predicate, a header is given. The header gives the predicate's name and describes how it is used : the list of arguments and the syntactic type of each argument. The following conventions are used :

`predicate_name ( Arguments )`

Argument*	The argument must contain no unbound variables.
Argument+	The argument must be partially bound, but may include some unbound variables.
Argument	The argument may be an unbound variable, or it may partially or fully bound.

module playback.

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%           Module Interface
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%
%       The Temporal Semantic Network
%
%       The temporal semantic networks for MRRs will be
%       stored in separate modules. To access a particular
%       MRR TSN, the module must be linked with this
%       program
%

```

```

import (node/3).      % node (ID, Subnet, Type)
import (arc/3).       % arc (Node1, Relation, Node2).

```

```

%
%       Communication Predicates with the Device Drivers
%       (Implementation Dependent)
%
import (send/2).      % send (Destination*, Command*)
import (receive/1).   % receive (Source)

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       The Body of the Module
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

body.

```

%
% There are two screens in this system.
%
screen (screen1).
screen (screen2).
%
% Position of the network traversal within each subnet
%
value (screen1).
value (screen2).
value (annotation).

```

%%%%%%%%%%		%
%	The Main Logic of the	%
%	Network Traversal Algorithm	%
%		%
%%%%%%%%%%		%

```
present_report :-
    set_value (annotation, start),
    set_value (screen1, start),
    set_value (screen2, start),
    temporal fwd chain.
```

```
%
% temporal_fwd_chain
%
% The predicate controls the flow of the presentation.
% Each step in the presentation is generated by performing
% a forward central traversal of the annotation subnet of
% the MRR TSN, using what we call "temporal forward chaining".
%
```

```

temporal_fwd_chain :-
    next_annotation (end). % Termination Condition
temporal_fwd_chain :-
    repeat,
        next_annotation (Annotation_Node),
            % Goal = Next Annotation
            % Node
        begin_annotation (Annotation_Node),
            % Satisfy both the preliminary
            % and the concurrent temporal
            % conditions, thereby triggering
            % the annotation behaviour.
        annotation_completed, % Wait until this step
            % in the presentation is
            % complete
        minimum_global_display
            % Pause, if necessary,
            % before continuing.
        next_annotation (end).
            % Repeat until the end of
            % the TSN is reached.

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Predicates for Preliminary
%      Temporal Conditions
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%
% begin_annotation (Annotation_Node*)
%
%      Effects the movement to the next annotation node
%      by invoking the behaviour or activity associated
%      with this annotation node.
%
begin_annotation (Annotation_Node) :-
    % IF all preliminary conditions met
    % (ie. all screens are synchronized)
    move_annotation (Annotation_Node),
    !,
    % THEN trigger the annotation behaviour
    % in conjunction with all concurrent
    % temporal relations.
    schedule_annotation (Annotation_Node),
    !.

begin_annotation (Annotation_Node) :-
    % OTHERWISE, first meet the
    % preliminary conditions
    % (ie. synchronize the screens)
    % before activating the
    % annotation's behaviour.
    update_screens (Annotation_Node),
    % Begin updates on ALL
    % out-of-sync screens.
    screen_update_completed (Screen),
    % Wait for a completion
    % notification
    begin_annotation (Annotation_Node, Activity),
    % In the recursive call,
    % EITHER Clause 1 will be used if
    % all screens are now
    % synchronized
    % OR Clause 2 will be used if
    % other updates are still in
    % progress or if there is
    % another update to be
    % triggered on Screen.
    !.

```

```

%
% move_annotation (Annotation_Node*)
%
%     The predicate succeeds only if all the preliminary
%     temporal conditions for the annotation node
%     (represented by arcs extending to the screen subnets)
%     hold.
%
%     ie. if all the screens are "synchronized".
%
move_annotation (Annotation_Node) :-
    bag_of (Relation-Screen_Node,
            arc(Annotation_Node,Relation,Screen_Node),
            Temporal_Conditions),
    temporal_conditions_met
        (Annotation_Node, Temporal_Conditions),
    !.

%
% temporal_conditions_met (Annotation_Node*,
%                          Temporal_Conditions*)
%
%
temporal_conditions_met (Annotation_Node, []) :-
    !.
temporal_conditions_met (Annotation_Node,
                        [Relation-Screen_Node|Conditions]) :-
    synchronized
        (Annotation_Node, Relation, Screen_Node),
    temporal_conditions_met
        (Annotation_Node, Conditions).

%
% update_screens (Annotation_Node*)
%
%     Advances forward along each of the "out-of-sync"
%     screen subnets, triggering updates on their displays.
%
%
update_screens (Annotation_Node) :-
    screen (Screen),
    update_screen (Screen, Annotation_Node),
    fail.
update_screens (Annotation_Node).

```

```

%
% update_screen (Screen*, Annotation_Node*)
%
%      If the screen is out-of-sync with the Annotation
%      Node and no screen update is already in progress,
%      the predicate triggers the next pending update
%      (as indicated by the next action node in the
%      Screen's subnet).
%
update_screen (Screen, Annotation_Node) :-
    out_of_sync (Screen, Annotation_Node),
    !,
    next_visual (Screen, Visual_Action),
    ( node (Visual_Action, Screen, display,
        trigger_visual (Screen, Visual_Action),
        ! ;
        succeed
        % Screen is out-of-sync but an update
        % is in progress.
    ).
update_screen (Screen, Annotation_Node).

%
% out_of_sync (Screen*, Annotation_Node*)
%
%      A Screen is not synchronized with the Annotation
%      Node if the temporal relation between this
%      Annotation Node and Screen (as indicated by
%      an arc between the two respective subnets) does
%      not hold.

out_of_sync (Screen, Annotation_Node) :-
    not screen_is_sync (Screen, Annotation_Node).

screen_is_sync (Screen, Annotation_Node) :-
    value (Screen, Screen_Node),
    arc (Annotation_Node, Relation, Screen_Node),
    synchronized
        (Annotation_Node, Relation, Screen_Node),
    !.

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%   Predicates for Defining
%   Synchronized Screens
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%
%   synchronized (Annotation_Node*, Relation*, Visual_Node*)
%

```

```

%   Defines when a screen is synchronized,
%   ie. when a preliminary temporal condition holds.
%

```

```

%   Assumption : Temporal Predicates are used in
%                 strictly defined cases. A clause is
%                 defined for each temporal predicate,
%                 with each APPLICABLE combination of
%                 action and effect nodes.
%

```

```

%   NOTE : Not all clauses have been defined in this
%           preliminary system.
%

```

```

synchronized (Annotation_Node, while, Visual_Node) :-
    % Annotation "while" DISPLAY
    !,
    node (Visual_Node, Screen, display),
    value (Screen, Visual_Node).

```

```

synchronized (Annotation_Node, Relation, Visual_Node) :-
    % Annotation "starts_during"
    %           or "begins"      Action
    ( Relation == starts_during ;
      Relation == begins ),
    !,
    node (Visual_Node, Screen, Action),
    value (Screen, Display_Node),
    next_visual (Display_Node, Screen, Visual_Node).

```

```

synchronized (Annotation_Node, independent, Visual_Node) :-
    !,
    node (Visual_Node, Screen, _),
    value (Screen, Display_Node),
    ( next_visual (Visual_Node, Screen, Display_Node) ;
      next_visual (Display_Node, Screen, Visual_Node)
    ).

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       Predicates for Concurrent
%       Temporal Conditions
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%
% schedule_annotation (Annotation_Node*)
%
%       Activates the behaviour associated with the
%       annotation node. Includes the definition of
%       concurrent temporal conditions, making use of
%       temporal precedence priorities to schedule
%       concurrent visual actions.
%

```

```

    schedule_annotation (Annotation_Node) :-
        high_priority_concurrencies (Annotation_Node),
        trigger_annotation (Annotation_Node),
        low_priority_concurrencies (Annotation_Node),
        !.

```

```

%
% high_priority_concurrencies (Annotation_Node*)
%
%       Activates the behaviour associated with ALL visual
%       nodes which occur concurrently with the annotation
%       node, but which must begin before it.
%

```

```

    high_priority_concurrencies (Annotation_Node) :-
        high_concurrent_condition (Relation),
        arc (Annotation_Node, Relation, Visual_Node),
        trigger_visual (Visual_Node),
        fail.

```

```

    high_priority_concurrencies (Annotation_Node).

```



```

%
% low_priority_concurrencies (Annotation_Node*)
%
%     Activates the behaviour associated with visual
%     nodes which occur concurrently with the annotation
%     node, but which must after before it.
%
low_priority_concurrencies (Annotation_Node) :-
    low_concurrent_condition (Relation),
    arc (Annotation_Node, Relation, Visual_Node),
    trigger_visual (Visual_Node),
    fail.
low_priority_concurrencies (Annotation_Node).

% Remember the assumption that the temporal predicates
% will be used in strictly defined arrangements. For
% this reason, while is not a concurrent condition.

%
% high_concurrent_condition (Relation)
%
%     Assuming an arc of the form
%     "Annotation_Node Relation Visual_Node",
%     the predicate lists, in descending priority, all
%     temporal relations which require that the visual
%     behaviour be started before the annotation
%     behaviour.
%
high_concurrent_condition (starts_during).

%
% low_concurrent_condition (Relation)
%
%     Assuming an arc of the form
%     "Annotation_Node Relation Visual_Node",
%     the predicate lists, in descending priority, all
%     temporal relations which require that the
%     annotation behaviour be started before the
%     visual behaviour.
%
low_concurrent_condition (begins).

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Minimum Global Display
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%
% Minimum_global_display
%
%      Ensures that the global display is static for
%      some minimum time, so that it can be seen by the
%      user. For now, it is enough for all outstanding
%      screen updates to complete For later, it will be
%      necessary to log when the last screen update
%      occurs, to record the passage of some minumum
%      time period.
%
minimum_global_display :-
    repeat,
        minimum_global_display (1).
minimum_global_display (N) :-
    matching_statement ((screen(Screen), N, M)),
    !,
    value (Screen, Node),
    node (Node, Screen, display),
    matching_global_display (M).
minimum_global_display (N).

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%      Predicates for Movements
%      within a Subnet
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%      NOTE : The procedures for finding the next node
%              within the sequential subnets make assumptions
%              about the subnet architecture; in particular,
%              a standard use of certain temporal predicates.

```

```

%
% next_annotation (Annotation_Node)
%
%      Finds the next node in the annotation subnet.
%
% next_annotation (Annotation_Node) :-
%
%      Annotation nodes are chained together into a
%      "PRECEDES" chain
%
%      value (annotation, Current_Node),
%      arc (Current_Node, precedes, Annotation_Node),
%      node (Annotation_Node, annotation, _).

```

```

%
% next_visual (Screen*, Visual_Node)
%
%      Finds the next node in the Screen's subnet.
%
% next_visual (Screen, Visual_Node) :-
%      % DISPLAY either "meets" or "contains"
%      % the next screen update
%      value (Screen, Current_Node),
%      node (Current_Node, Screen, display),
%      !,
%      ( arc (Current_Node, meets, Visual_Node) ;
%        arc (Current_Node, contains, Visual_Node)
%      ),
%      node (Visual_Node, Screen, _).
% next_visual (Screen, Visual_Node) :-
%      % Visual Actions always "meet" their
%      % DISPLAY
%      value (Screen, Current_Node),
%      arc (Current_Node, meets, Visual_Node),
%      node (Visual_Node, Screen, display).

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%       Predicates for Communicating
% with the Workstation Device Drivers
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

%
% trigger_annotation (Annotation_Node*)
%
%       Begins the behaviour associated with the
%       annotation node updating the position within
%       the annotation subnet.
%

```

```

    trigger_annotation (Annotation_Node) :-
        set_value (annotation, Annotation_Node),
        ( node (Annotation_Node, annotation, play),
          send (voice_driver, play) ;
          node (Annotation_Node, annotation, pause),
          send (timer, pause)
        ).

```

```

%
% trigger_visual (Visual_Node*)
%
%       Begins the behaviour associated with the
%       annotation node by sending the appropriate
%       command to the respective screen driver,
%       and updates the position within this
%       screen's subnet.
%

```

```

    trigger_visual (Visual_Node) :-
        node (Visual_Node, Screen, Action),
        set_value (Screen, Visual_Node),
        send (Screen, Action),
        !.

```

```
%
% screen_update_completed (Screen)
%
%     Invoked when only screen updates are in
%     progress, it waits for a completion notification
%     (from any of the screen drivers), and then updates
%     the position with the associated screen's subnet.
%
```

```
screen_update_completed (Screen) :-
    receive (Screen),
    next_visual (Screen),
    set_value (Screen, New_Display),
    !.
```

```
%
% annotation_completed
%
%     Waits until the annotation activity completes,
%     processing any completion notifications from
%     concurrent screen updates that arrive in the
%     meantime.
%
```

```
annotation_completed :-
    receive (Source),
    process_completion_notification (Source).
```

```
%
% process_completion_notification (Source*)
%
%     When both audio and visual operations are in
%     progress, the completion notification may be
%     returned in random order and thus require
%     different processing depending on the Source
%
```

```
process_completion_notification (Source) :-
    ( Source == timer ;
      Source == voice ),
    !.
```

```
process_completion_notification (Source) :-
    next_visual (Screen),
    set_value (Screen, New_Display),
    annotation_completed,
    !.
```

```
endmod.
```

Chapter 9 REFERENCES

Allen, J.F. "An Interval-Based Representation of Temporal Knowledge", *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, B.C., August 1981

Allen, J.F., Koomen, J.A., "Planning Using a Temporal World Model", *Proceedings of the National Conference on Artificial Intelligence*, Volume 2, 1983.

Allen, J.F., "Maintaining Knowledge About Temporal Intervals", *Communications of the ACM* 26(11), p.832-843, November 1983.

Barr, A., Feigenbaum, E.A., *The Handbook of Artificial Intelligence*, Volume 1, William Kaufmann, Inc. California, 1981.

Cohen, Paul R., Feigenbaum, Edward A., *The Handbook of Artificial Intelligence*, Volume 3, William Kaufmann, Inc. California, 1981.

Conklin, Jeff, "Hypertext : An Introduction and Survey", *IEEE Computer*, p.17-41, September 1987.

Cuadrado, John L., Cuadrado, Clara Y., "AI in Computer Vision", *BYTE*, p.237-258, January 1986.

Fikes, Richard, Kehler, Tom, "The Role of Frame-Based Representation in Reasoning", *Communications of the ACM* 28(9), p. 904-920, September 1985.

Firby, R.James, McDermott, Drew, "Representing and Solving Temporal Planning Problems", (Nick Cercone and Gordon McCalla) *The Knowledge Frontier : Essays in the Representation of Knowledge*, Springer-Verlag, New York, 1987.

Forsdick, H., Thomas, R., Robertson, G., Travers, V., "Initial Experience with Multimedia Documents in Diamond", *Computer-Based Message Services*, p. 99-113, 1984.

Garcia Luna Aceves, J.J., Poggio, A., "Multimedia Message Content Protocols for Computer Mail", *Computer-Based Message Services*, p.87-98, 1984.

Hendrix, Gary G., "Expanding the Utility of Semantic Networks Through Partitioning", *International Joint Conference on Artificial Intelligence*, 1975.

Horak, Wolfgang., "Office Document Architecture and Office Document Interchange Formats : Current Status of International Standardization", *IEEE Computer*, p.50-60, October 1985.

Huang, H.K., *Elements of Digital Radiology*, Prentice-Hall, 1987.

ISO DP/8613/1:1988, "Part 1 - Introduction and General Principles", *Information Processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format*, ISO/IEC JTC1/SC 18/WG3 N, March 1988.

Ladkin, Peter, "Logical Time Pieces", *AI Expert*, p.58-67, August 1987.

Nebel, Bernard, "How well does a vanilla loop fit into a frame?", *Data and Knowledge Engineering 1*, p. 181-194, 1985.

Nilsson, Nils J. *Principles of Artificial Intelligence*, Morgan Kaufmann Publishers, Inc., 1980.

Perry, T.S, "Hypermedia : finally here", *IEEE Spectrum*, Volume 24, Number 11, p. 38-39, November 1987.

Poggio, A., Garcia Luna Aceves, J.J., et al., "CCWS : A Computer-Based Multimedia Information System", *IEEE Computer*, p.92-103. October 1985.

Woelk, D., Kim, W., Luther, W., "An Object-Oriented Approach to Multimedia Databases", *Communications of the ACM*, 1986.

Yankelovich, N., Hann, B.J., Meyrowitz, N.K., Drucker, S.M., "Intermedia : The Concept and the Construction of a Seamless Information Environment", *IEEE Computer*, Volume 21, Number 1, p.81-96, January 1988.

END

13|09|90

FIN