ELSEVIER

Computer Standards & Interfaces 18 (1997) 477–496

COMPUTER STANDARDS & INTERFACES

A standard reference model for intelligent multimedia presentation systems

M. Bordegoni [a,*], G. Faconti [b,1], S. Feiner [c,2], M.T. Maybury [d,3], T. Rist [e,4], S. Ruggieri [f,5], P. Trahanias [g,6], M. Wilson [h,7]

[a] *CNR ITIA, Milano, Italy*
[b] *CNR CNUCE, Pisa, Italy*
[c] *Columbia University, New York, NY, USA*
[d] *MITRE, Bedford, MA, USA*
[e] *DFKI, Saarbrücken, Germany*
[f] *Università di Pisa, Pisa, Italy*
[g] *FORTH-ICS, Heraklion, Greece*
[h] *CRLC-RAL, Chilton, Didcot, UK*

Abstract

This article summarizes the main results of a joint endeavor towards a standard reference model (SRM) for intelligent multimedia presentation systems (IMMPSs). After a brief motivation, we give basic definitions for *media terms* and *presentation systems*. The core of this contribution is a generic reference architecture that reflects an implementation-independent view of the processes required for the generation of multimedia presentations. The reference architecture is described in terms of *layers*, *components*, and *knowledge servers*. Our SRM focuses on the *functions* assigned to the layers and components, rather than on the methods or communication protocols that may be employed to realize this functionality. Finally, we point to some possible extensions of the reference model. © 1997 Elsevier Science B.V.

Keywords: Intelligent multimedia presentation system; Automated generation of multimedia presentation; Standard reference model

1. Introduction

1.1. The need for intelligent multimedia presentation systems (IMMPSs)

The acceptance and utility of a broad range of application systems is substantially affected by their limited ability to present information in an effective

* Corresponding author. E-mail: mb@itia.mi.cnr.it
[1] E-mail: faconti@cnuce.cnr.it
[2] E-mail: feiner@cs.columbia.edu
[3] E-mail: maybury@mitre.org
[4] E-mail: rist@acm.org
[5] E-mail: ruggieri@di.unipi.it
[6] E-mail: trahania@ics.forth.gr
[7] E-mail: mdw@inf.rl.ac.uk

PII S0920-5489(97)00013-5

and appealing way to human users. Rapid progress in the development of multimedia technology promises more efficient forms of machine/human communication. However, multimedia presentation design is not just a matter of merging output fragments, but requires fine-grained coordination of communication media and modalities. This may even become a harder and more complex task than solving the application problem. Furthermore, in the vast majority of nontrivial applications the information needs will vary from user to user and from situation to situation. Consequently, a presentation system should be able to flexibly generate various presentations for one and the same information content in order to meet the individual requirements of users and situations, the resource limitations of the computing system, and so forth.

As the need for presentation flexibility grows, 'manual' authoring and preparation of presentations becomes less and less feasible. Starting from this observation, the development of mechanisms for the automated generation of multimedia presentations has become a shared goal across many disciplines. To ensure that the generated presentations are understandable and effective, these mechanisms need to be *intelligent* in the sense that they are able to make appropriate design decisions based on *presentation knowledge* and *contextual knowledge*, and to manage the various interdependencies between choices.

1.2. The need for a reference model for IMMPSs

During the last decade, some research and development effort has been directed toward automated multimedia presentation generation (e.g. [1]). There are already examples of successful technology transfer from research lab prototype systems to industrial applications including, for example, the generation of assembly sheets for electrical circuits [2], and interfaces for traffic management systems [3]. However, no generic model for IMMPSs has emerged. Projects typically begin from scratch, relying only on the past experience of the developers and the reports of other researchers. Consequently, there is often replication of results, limited reuse of previous solutions, and limited synergy among parallel ongoing efforts. Furthermore, there is no agreement on the terminology to be used, on the functional definition of an IMMPS, or on a generic architecture that reflects an implementation-independent view of the processes required for the generation of multimedia presentations.

By proposing a standard reference model, this article tries to fill a major methodological gap by providing a sound basis for ongoing and future development of IMMPSs. Throughout the rest of the article we adopt 'SRM' as an abbreviation for 'standard reference architecture/model for IMMPSs.'

There are several advantages to agreeing on a reference model. The development and the analysis of systems would benefit from a uniform approach to the problem. The modular development of large-size programs would be feasible, as each module has a well defined role, with well defined interfaces and communication protocols with the other modules. From a theoretical point of view, the analysis and comparison of systems could be made on the basis of a single general architecture and by means of a common terminology. These considerations are universally valid for any reference model. In the case of IMMPSs, however, they assume a deeper significance, as the literature is especially confusing with respect to basic terminology.

We also argue that existing general reference architectures for user interfaces and user interface management systems (UIMSs) are not sufficiently detailed to capture the structure of IMMPSs. For example, the Seeheim model [4] has gained considerable popularity among user interface designers. As shown in Fig. 1, the Seeheim model consists of three components: an application interface component, a dialogue control component, and a presentation component. The *application interface component* represents those data structures and functions in the application that are relevant to the user interface. The

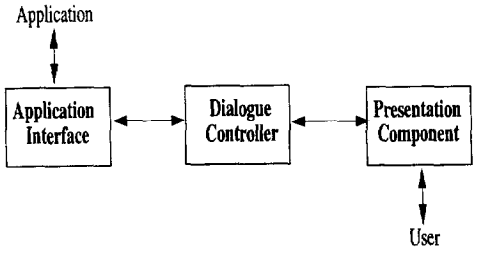


Fig. 1. The Seeheim reference model for UIMSs.

dialogue control component oversees the user-application dialogue, mapping information between the application interface and the presentation component; for example, by encoding the range of possible dialogue as recursive transition networks, BNF grammars, or events. The *presentation component* controls the display and interaction devices, determining how and where information is presented to and obtained from the user. It is the only part of the model that has any dependencies on the devices, media, and modalities used.

Other more recent architectures have refined the Seeheim model. For example, the ARCH model [5] explicitly separates the presentation component from its general purpose interaction toolkit, and partitions the application interface component into a domain specific component and an interface with the dialogue component. Because of their generality, however, the Seeheim model and its descendants do not provide any guidance for describing an IMMPS, except insofar as the IMMPS's tasks would be divided among the high level components.

An early attempt to extend and elaborate the Seeheim model specifically for IMMPSs was developed and presented at the 1988 Workshop on Architectures for Intelligent Interfaces [6]. Its *application interface component* formulates a set of goals to be accomplished by the rest of the interface. These goals are presented to a knowledge based *dialogue design component*, which replaces the Seeheim model's dialogue control component. The dialogue design component plans how to accomplish the goals by determining what information to present, working at a level above that of generating material in any specific modality. Finally, the Seeheim model's monolithic presentation component is replaced with: a *media coordinator*, which decides how to map information to modalities and oversees generation; a set of *media experts*, which design for each supported modality; a *media layout component*, which determines the size and position of the material designed by the media experts; *rendering, typesetting*, and *interaction* components, which use conventional graphics, text, and interaction handling software to realize the generated presentation; and device-dependent *display and interaction hardware* components, which interface with the actual devices. A set of knowledge sources accessible to the components includes: models of the user, the current situation, and the display and interaction hardware; a history of the current and previous sessions; and application-specific knowledge. While some of this structure is reflected in the reference model developed in this article, the earlier model was proposed as a strawman, representing only a single researcher's experience, and doesn't attempt to formalize the internals of its individual components.

1.3. Scope of the SRM

Our reference model is targeted towards the class of systems (or components of superordinate systems) whose task is to flexibly present information to the user in an *effective* way. We use the word 'effective' to mean that the specific information needs of the individual user are met, given a set of presentation constraints, such as resource limitations and the user's knowledge and stylistic preferences. There are some applications that are sufficiently constrained that automated presentation design would not provide any significant added value. Instead, we focus on applications with sufficiently complex presentation tasks, such as medical, educational, or industrial systems, which present large amounts of complex information to users with a wide range of knowledge, skills, and tasks, in a variety of changing environments and situations.

We assume throughout that a human user is the addressee of a generated presentation. While, in principle, the addressee could be *any* external entity that is able to interpret the presentation, we choose to ignore the question of whether some other form of communication might be more effective and efficient if the entity were another computer system.

We have explicitly decided not to address a more general class of systems: *intelligent multimedia dialogue systems*, which analyze multimedia input from the user, in addition to generating multimedia output. While we consider this generalization desirable, it is beyond the scope of this effort. However, excluding full-fledged intelligent multimedia dialogue systems from the discussion does not necessarily mean excluding interactivity completely. Many interactive systems provide interaction facilities that do not require deep analysis of user input. Examples include hypermedia systems in which pointing gestures are

associated with commands, and systems that obtain user input through menus or a command language. If a multimedia system supports simple input of this sort, but performs intelligent presentation processing, then we consider it an IMMPS, rather than an intelligent multimedia dialogue system.

Finally, we stress that the reference model is intended to fit *real* systems. However, this does not necessarily mean that the proposed architecture has to be followed in an implementation. The reference model simply provides a logical view of the generic tasks that occur in multimedia presentation generation.

1.4. Underlying design rationale

Having identified the class of systems to be captured by the model, we turn our attention to the general guidelines that underly our design decisions: adequate modularization, appropriate degree of abstraction, identification and classification of knowledge sources, modeling of shared sources using a client–server paradigm, and openness to other standards.

Adequate modularization. To facilitate the development and comparison of practical large scale systems, the reference model must represent a modularization of a generic process for multimedia presentation generation. Ideally, this modularization should break down the generation process into logically distinct and computationally feasible subtasks.

Appropriate degree of abstraction. The reference model should reflect the unique characteristics of multimedia generation, while remaining general enough to capture the whole class of IMMPSs. It should be sufficiently abstract that it does not specify the exact mechanism needed to accomplish an individual generation subtask, or the exact format needed to represent a particular kind of knowledge.

Identification and classification of knowledge sources. Since intelligent presentation design is a knowledge-intensive task, the reference model should include the basic set of different *knowledge sources* that are required for multimedia presentation generation. The reference model should also make clear how processes and knowledge sources are related to each other. In particular, *private* knowledge sources (i.e., sources for which a single owner component can be identified), should be distinguished from *shared* sources that are used by several components.

Modeling of shared sources using a client–server paradigm. To facilitate knowledge sharing (in the generic architecture, as well as in concrete system implementations) shared knowledge sources should be modeled using a client–server paradigm. Such sources will be referred to as *expert modules*, and are designed to serve requests from client modules, possibly belonging to other systems.

Openness to other standards. Multimedia generation comprises subtasks that have been treated in other disciplines. Therefore, the model should be open so that it can be used in conjunction with existing or potential standards in these disciplines. For example, the *Computer Graphics Reference Model* [7] may be used to instantiate the subcomponent for graphics display in the generic architecture of our model. Results from the PREMO [8] standardization initiative may be used for both the description of media objects and an object-oriented conceptualization of the proposed reference architecture. A standardized language could be used to exchange knowledge among components, such as KQML (*Knowledge Query and Manipulation Language* [9]). Conversely, the reference model can itself become a component of a superordinate model that includes a user interface.

2. Basic notions

We begin by defining basic terminology [8] for multimedia presentation design and corresponding computational implementations.

2.1. Terms related to media and modalities

There is much confusion in the literature over what is meant by the term *medium* and its plural

[8] Caution: The authors are aware that the definitions given here may not be compatible with the meanings that other authors have assigned to the same concepts. However, most of our definitions are borrowed from, or at least in the spirit of, the cited source documents. Also, we do not make any claims for completeness, but have instead restricted ourselves to introducing only those terms that are needed to serve as a basis for defining the reference architecture.

media. One reason for this misunderstanding is that these terms are used with different meanings in different contexts, such as semiotics, psychology, telecommunications, and computer science. The closely related term *modality* is a similar source of confusion. Some authors use medium and modality as synonyms, while others tend to reserve modality only for input (to be processed by a machine interpreter), and medium only for output (to be produced by a machine generator). It is not our intention to provide the ultimate solution to the medium/modality debate across all disciplines. Rather, our use of terms results from a pragmatic 'merge' of different approaches. Our view of the *medium / modality* distinction is illustrated in Fig. 2.

The leftmost column represents a human user sensing an environment that includes output delivered by dedicated display devices. Taking this view, we give different meanings to the term *medium*, depending on the particular focus or perspective.

Medium as physical space to realize perceptible entities. With the human sensory apparatus as 'target system,' we can consider a medium as being a certain physical space in which perceptible entities are realized. As there are different types of perceptible entities (e.g., visual, auditory, haptic, and olfactory), one may use these terms for differentiating among media as well. Further refinements can also be made (e.g., to differentiate between mono and stereo visual or auditory media). Thus, we can characterize a medium by the physical dimensions needed to realize a particular perceptible entity.

Medium as a type of information and / or representation format. In this view, medium refers to a certain type of information and/or the representation format in which information is stored. Examples include pixmap graphics, 3D scene descriptions, sequences of video frames, and audio data. It is assumed that for each medium there is a dedicated physical device that is able to produce perceptible entities. That is, devices serve to display media objects to the user's senses. In addition to conventional display devices, such as screens, printers, and speakers, more futuristic examples are possible, such as spatial light modulators for electronic holography, or devices that can produce smells. A framework for a characterization of media types is proposed as the PREMO primitive hierarchy in the PREMO standard (Part 4 of [10,11]).

While the term *multimedia* clearly refers to the use of multiple media, one has to take into account which meaning of media is meant. In the first case, one may think of a common physical space in which different perceptible entities can be realized (e.g., the simultaneous production of visible and audible material). In the second case, multimedia stands for a composition of basic media types. For example, one may compose a medium *video-with-audio* from the two component media *video* and *audio*. A characterization of composed media may include additional

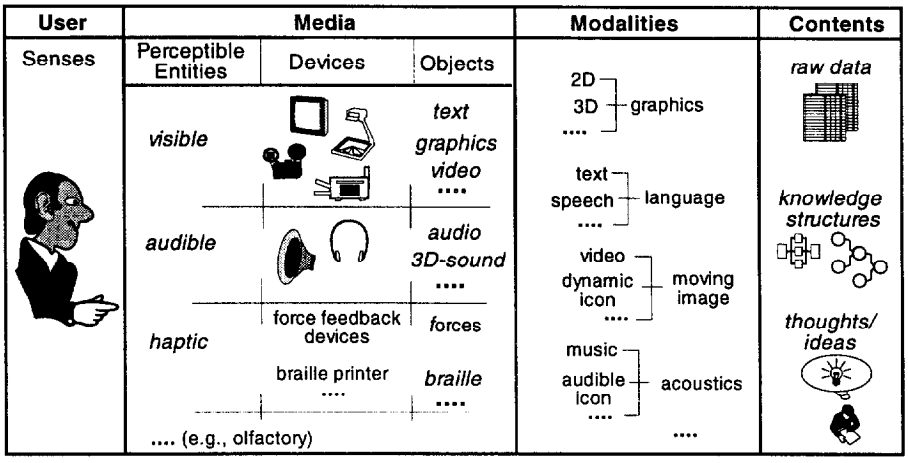


Fig. 2. The medium/modality distinction.

properties such as temporal relationships among the involved media.

A further distinction exists between *types* and *instances*. Again, what we mean by medium is decisive. When considering medium as a physical space, instantiations are particularly perceptible entities. For example, a beep is an instantiation of an audible entity. In the other reading of media, an instance corresponds to a particular set of data in a certain format; for example, this document's ASCII source file is a media object.

Multimedia systems research has concentrated on the representation, creation, storage, conversion, distribution, and display of media objects. When dealing with IMMPSs, however, the information content of a media object and the way it is encoded must also be addressed. The rightmost column of Fig. 2 illustrates what is meant by information content. If the author is human, information content may be an intellectual entity existing only in the author's mind; if the author is a computer system, there must be an explicit representation of the information content. As these representations are usually not in a format that is directly presentable to human users, we may speak of them as *raw data*, or as *knowledge structures* in the case of highly structured representation formats.

To act as bridge between information content and media objects, we introduce the terms modality, and multimodality.

Modality refers to a particular way or mechanism of encoding information for presentation to humans or machines in a physically realized form. Examples include 2D and 3D graphics, written and spoken natural language, music, and audio icons (For a more fine-grained classification of presentation modalities, see [12].).

Multimodality refers to the use of multiple modalities to encode information. For example, this article is a multimodal presentation since it utilizes written text and 2D diagrams to describe the reference model. Note that a single medium may be used as a common physical realization of several modalities. For example, the audio medium is often to convey multimodal information in the form of speech accompanied by background music.

Choosing a modality places restrictions on the medium to be used. Moreover, in many concrete system implementations, there may even be a one-to-one mapping from certain modalities to certain media objects. Therefore, it is hardly surprising that modality and medium have often been used interchangeably.

In the context of this article, the ultimate goal of using media and modalities is to communicate to the user. However, the *use* of media/modalities may be further distinguished with regard to different aspects (e.g., *sequential* vs. *parallel* use with regard to time, *redundant* vs. *complementary* use with regard to *information encoding*, or *informative* vs. *decorative* use with respect to *communicative function*).

We now introduce some additional terms: presentation, multimedia/multimodal referring expression, and cross-media/cross-modal reference.

Presentation refers to any composition of media objects that have been created for the purpose of communicating information to a user. Depending on context, we will use the term 'presentation' either to emphasize the material collection of media objects (i.e., presentation = document), or to emphasize the activity of communicating information to a user.

Multimedia / multimodal referring expression means a composition of at least two different media/modalities included in a presentation with the intention of referring to an object in the application domain.

Cross-media / cross-modal reference means part of a presentation intended to refer to another part of a presentation, in which the referring part and the referenced part are realized in different media/modalities (e.g., text in the caption of a graphic that refers to a part of the graphic).

2.2. Terms related to presentation systems

In this section we introduce some basic terms for characterizing presentation systems.

Presentation system refers to the class of computer systems that are designed for presenting information to users. Presentation systems perform information transformation processes. Depending on the nature and complexity of the transformation, we may further classify systems using the intuitively motivated distinction between those that merely *display* presentations, and those that *generate* presentations. Our focus is on generative presentation systems, whose behavior cannot be described by a simple

one-to-one mapping of input units to media objects that will be displayed to the user.

Presentation goals have to be achieved by a presentation system, and are possibly accompanied by a set of *presentation commands* or *presentation constraints* (e.g., 'the presentation must not exceed one page') affecting the presentation process. These constitute the primary input to a generative presentation system. Both goals and commands are assumed to be formulated outside the presentation system (e.g., by the user, by an external system, or by a superordinate component if the presentation system is a part of a larger system). Goals and commands include high-level references to collections of data together with the purposes (or intentions) for communicating information.

Application data / knowledge provides the semantic grounding for each presentation that may be generated by a system. As with presentation goals and commands, it is assumed that application data/knowledge is part of the input to a presentation system (see also the rightmost column of Fig. 2). In other words, there must be an external source (e.g., an application system or a database) that makes available to the presentation system the application data necessary for achieving posted goals.

Multimedia presentation system is a presentation system that employs multiple media to present information to the user. Again, the focus of the reference model is on generative multimedia presentation systems (i.e., systems that transform presentation goals into multimedia presentations).

Intelligent multimedia presentation systems are essentially knowledge-based systems, i.e., systems that maintain a declarative knowledge representation (usually called a *knowledge base*) and application targeted computation processes that exploit the knowledge base. In an IMMPS, the knowledge base is used to make presentation design decisions.

The internal processes that a presentation system may perform can be characterized using the terms: content selection, media/modality allocation, and media/modality-specific encoding.

Content selection refers to the process of selecting specific content (i.e., application data/knowledge) for presentation to achieve certain presentation goals. Content selection can be regarded as a filtering process.

Media / modality allocation refers to the process of deciding which medium/modality or combination of media/modalities to choose for presenting selected content to achieve its associated presentation goals.

Media / modality-specific encoding refers to the process of generating media objects conveying a specific message in a certain combination of media/modalities. For example, a presentation system may comprise dedicated generation modules for encoding relational data sets either by formatted tables, charts, or pure text.

3. The reference architecture: Overview

The core of the reference model is a generic architecture for IMMPSs. Following standard architectural models, the building blocks of the architecture are layers, components, and connectors. *A layer* serves as an abstract location for tasks, processes, or system components. A *component* is an objectification of a task, function or computing process in a real or abstract system. Components are essentially characterized by their particular input/output behavior. The internal structure of a component may be given by means of a set of subcomponents together with input/output relations between these subcomponents. A *connector* between two entities enables (directed) interchange of information between them.

Fig. 3 outlines the IMMPS reference architecture. Its conceptual design reflects a modularization of the generation process into five layers, which represent particular subtasks: Control Layer, Content Layer, Design Layer, Realization Layer and Presentation Display Layer. In addition to their private knowledge, the layers can exploit explicitly encoded knowledge provided by a separate Knowledge Server, which is composed of four shared knowledge sources: Application Expert, Context Expert, User Expert and Design Expert.

The following conventions are used in Fig. 3 and all subsequent figures: *Rectangular boxes* denote either layers or components that form the *internal parts* of an IMMPS. *Ellipses* indicate *external* entities with which an IMMPS may interact. The interchange of information between components is reflected by arrows. Double-headed arrows indicate

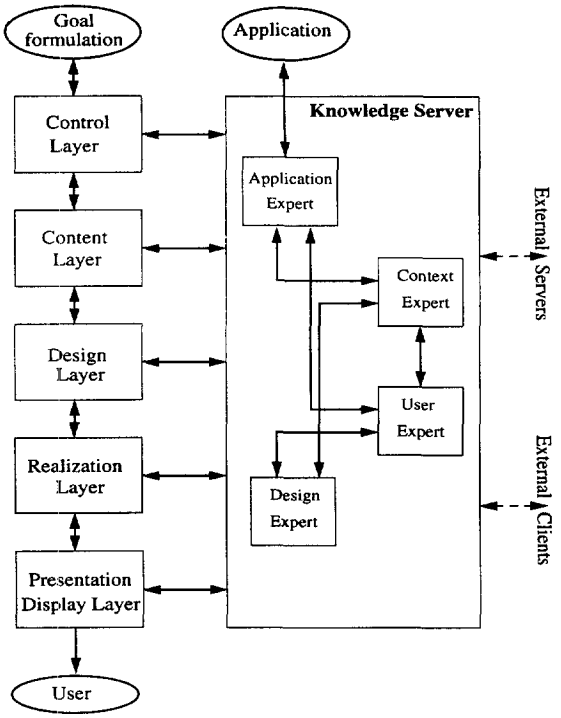


Fig. 3. The standard reference architecture for IMMPSs.

bidirectional information flow. Dashed arrows indicate *optional* interaction channels to external parts. To indicate exchange of information between *some* internal components of a layer and another entity outside the layer, we use arrows that point to the interior of the layer but not specifically to one of the subcomponents. Throughout the rest of the article we treat references to the SRM's layers and components as proper names starting with capital letters.

3.1. Interfaces to external entities

It is quite common to describe systems with respect to their input/output functionalities. As a first step, we have to clarify the boundaries between the components of the SRM and external entities.

Presentation goals and commands form the triggering input to an IMMPS. As goals and commands are processed through the layers together with application data/knowledge, they are eventually transformed into presentations composed of media objects. This view is reflected in Fig. 3 through the ellipses labeled *Goal Formulation*, *Application*, and *User* (the presentation consumer).

Although in many concrete system implementations Application and Goal Formulation and are combined as a single component, we have separated them to distinguish between the *availability* of application knowledge/data, and the *use* of that knowledge/data to satisfy presentation goals. While we assume the presentation consumer is a human user, goal formulation may be done either by a human (e.g., using a keyboard to type in goals, or selecting predefined goals from a menu) or a computer. We do not further distinguish whether or not the *presentation consumer* and the *goal formulator* are identical, although this should be reflected in the user model of a concrete system.

As presentation systems are typically not standalone applications, but parts of user interfaces to complex systems, they often have interfaces to external components. For example, an IMMPS may acquire knowledge from *external knowledge bases*, or directly from the user. Although not explicitly handled here, interactions with external unspecified systems/entities might be modeled by means of *data capture metafiles*, as in the Computer Graphics Reference Model [7].

4. Layers

The partitioning of the transformation process into five layers reflects a rough grouping of the subtasks that are performed by an IMMPS.

4.1. Control Layer

The first layer of the SRM embodies components, shown in Fig. 4, that handle incoming presentation goals, presentation commands, and possibly further commands (e.g., *start*, *stop / interrupt*, or *refine goal*) that allow the user (or a superordinate system) to control the generation process. Control is carried out by an external entity via the Goal Formulation Interface, a component whose task is to convert incoming messages into internal formats understandable by the IMMPS. For example, some systems have a menu-based Goal Formulation Interface that allows the user to choose from a set of preformulated

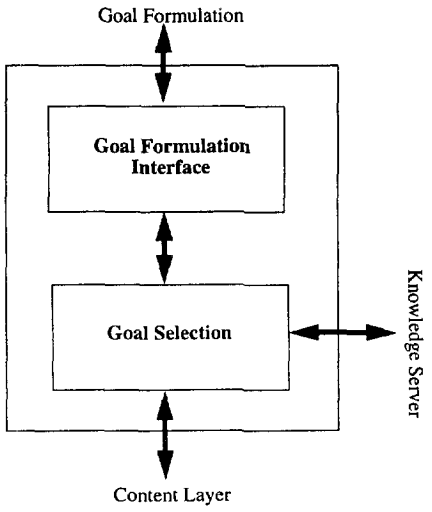


Fig. 4. The control layer.

presentation goals, whereas other systems are always triggered by the 'standard goal' (*Present* ⟨*data*⟩), where ⟨data⟩ is a reference to a specific portion of application data.

External control must not be confused with *internal* control functions that are required to coordinate various generation sub-processes in an implementation of an IMMPS. While early IMMPSs mostly have hierarchical or centralized control structures, there are already implementations that are moving towards distributed system designs where control is hidden in communication and negotiation patterns of the distributed components. Since the SRM is intended to capture the structure of a generic generation process, we leave open the architectural organization of system components that one may choose for a concrete implementation.

Regardless of how an IMMPS is actually implemented, it must select from the Goal Formulation Interface the next goal to be achieved or the next presentation command to be executed. This is the task of the Goal Selection component, and it occurs in multimedia generation for two different reasons: First, it might be the case that the (external) goal formulator poses a set of goals to the presentation system, either all at once or piecemeal. In the latter case, an ongoing generation process may need to be interrupted to achieve a new incoming goal immediately. Second, the presentation goals input to the system may be complex, and may need to be split into sets of less complex goals. While goal decomposition will be done by the Goal Refinement component of the Content Layer, the Control Layer still has to decide the order in which the subgoals will actually be processed. Deciding the next goal to be achieved might be a simple task, requiring only that a goal be popped from the discourse model (maintained by the Context Expert, described in Section 5.2). On the other hand, the Goal Selection component may contain some private knowledge if its decisions involve more complex reasoning.

4.2. Content Layer

Four high-level authoring tasks are grouped together in the SRM's Content Layer, shown in Fig. 5: goal refinement, content selection, media allocation, and ordering.

The term *goal refinement* captures both the decomposition of a goal into a set of subgoals and the specialization of a goal. The Goal Refinement component is needed because presentation goals are often formulated at a high level of abstraction. During goal refinement, the content of the final presentation will be determined. The Content Selection component assists in carrying out this task. In a concrete

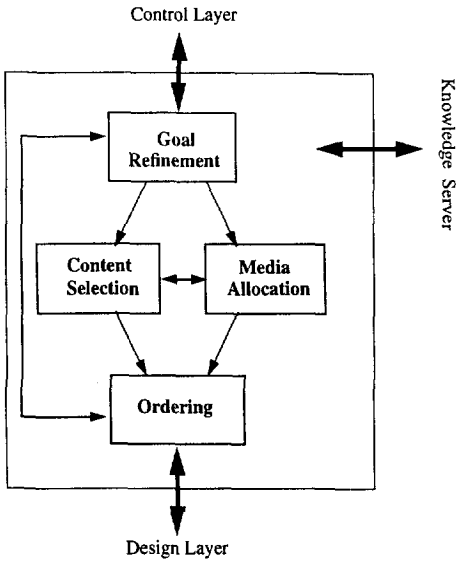


Fig. 5. Content layer.

system, the Content Selection component may appear as a retrieval and filter module that communicates with the Application Expert. The output of Goal Refinement and Content Selection is a set of *communicative acts* and a structural description of the relations that may hold between these acts. Communicative acts are an extension of speech acts [13] to multimedia communications (also see [14,15]).

As soon as the communicative acts have been determined, the Media Allocation component decides which modalities and media should be used to convey them. Note that there are different approaches for media allocation. One approach is to map specific types of data or information onto specific types of media (e.g., database entries onto tables or maps). A more compositional approach to media allocation involves matching properties of information (e.g., quantitative vs. qualitative, fixed vs. varying) with properties of media (e.g., persistent vs. transient, static vs. dynamic, 2D vs. 3D). As the Media Allocation component assigns particular media/modalities to communicative acts, its output is a set of what are now called *media communicative acts*.

The Ordering component determines the order in which the selected content should be communicated to the user. This ordering will be constrained by the semantic and pragmatic relationships among the media communicative acts. As the semantic and pragmatic relationships eventually must be made recognizable to the presentation consumer, the ordering of the media communicative acts will constrain the preceding generation steps. While the term *ordering* is related to the term *linearization* used in the field of text generation, it is important to note that multimedia presentations do not necessarily follow the linear structures that written text and speech do.

Since there are many dependencies among choices, the components of the Content Layer are all connected to each other to make negotiation possible. However, actual system implementations often do not handle some dependencies, and thus do not allow for negotiations among all tasks in the Content Layer.

4.3. Design Layer

An important task of an IMMPS is the transformation of media communicative acts into specifications of media objects together with a specification of an overall presentation layout. While it might seem straightforward to have two layers in the SRM, one for the production of media objects and a subsequent one for their layout, we have adopted another structure that is based on the following observations: (a) both media-specific production and presentation layout are complex tasks that can each be broken down into (at least) a *design task* and a *realization task*; (b) there is no justification for assuming media-specific production must necessarily precede presentation layout.

The SRM's Design Layer and Realization Layer reflect these observations. By *design* we refer to the task of planning how to convey a certain communicative act by using a particular medium/modality and/or layout structure. The design task itself is further broken down into tasks for a Media Design component and a Layout Design component, as shown in Fig. 6.

The Media Design Component, shown in more detail in Fig. 7, contains a set of Media/Modality-Specific Design components, which are dedicated modules for designing 2D and 3D graphics, natural language utterances, animation, video, music, etc. A Dispatching Component is included to represent the task of dispatching media communicative acts to the relevant Media/Modality-Specific Design components, and to collect and pass on the results of these components. While we assume that an IMMPS is

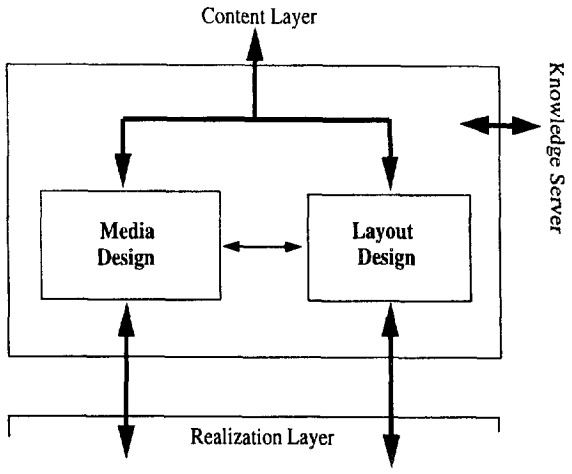


Fig. 6. The design layer.

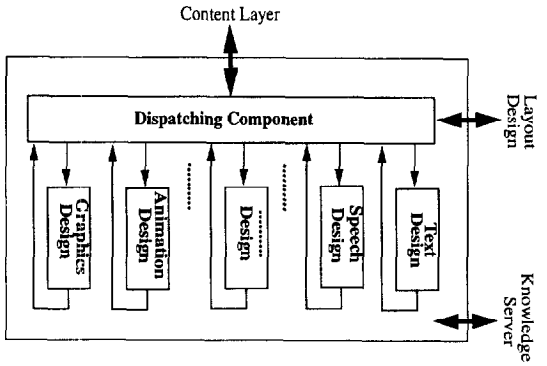


Fig. 7. The media design component.

capable of producing output in at least two different media, we do not impose an upper bound on the total number of Media/Modality-Specific Design components. Moreover, an IMMPS might even include several components for one medium but different modalities (e.g., different graphics design components for different subtypes of graphics).

For the sake of generality, the SRM does not provide fine-grained descriptions of the Media/Modality-Specific Design components. However, we note that media communicative acts constrain the mechanisms employed for designing media objects. For any single medium or modality, different IMMPSs often include components that rely on very different design mechanisms (e.g., grammar-based, case-based, plan-based, or genetic programming-based approaches).

The Layout Design component determines the spatiotemporal arrangement of media objects in the presentation. The SRM does not further formalize the details of the layout task. We refer to the article by Graf [16] in this issue, which outlines a reference model for layout systems. As indicated in Fig. 6, the Design Layer does not prescribe a particular ordering in which media design and layout design must be carried out. Thus any of the following are possible [17]:

- layout *after* production: the resulting media objects constrain layout decisions.
- layout *before* production: layout decisions constrain the production of media objects.
- layout *interleaved with* production: production and layout can constrain each other.

The results of the Design Layer are *realization plans*, which are ordered sets of *realization commands* to be executed by dedicated realization components. We assume that the Media Design and Layout Design components have counterparts in the Realization Layer's media realization and layout realization components. Since these realization components handle only media realization commands and layout realization commands, respectively, the exchange of information between the Design Layer and Realization Layer is indicated by two separate arrows in Figs. 6 and 8.

4.4. Realization Layer

By *realization*, we refer to the task of creating from a design plan the specifications for concrete media objects and their layouts. As mentioned above, the Realization Layer is structured like the Design Layer in that it consists of a Media Realization component, and a Layout Realization component, as shown in Fig. 8. In analogy to the media design component, the media realization component contains a set of Media/Modality-Specific Realization components, which are dedicated modules for realizing material in specific media and modalities according to design plans (Fig. 9).

A Dispatching Component is included in the Media Realization Component to represent the task of dispatching media realization commands to the relevant Media/Modality-Specific Realization components, and to collect and pass on the results of these components. It is important to note that there is not necessarily a one-to-one relationship between the Media/Modality-Specific Design components and the Media/Modality-Specific Realization compo-

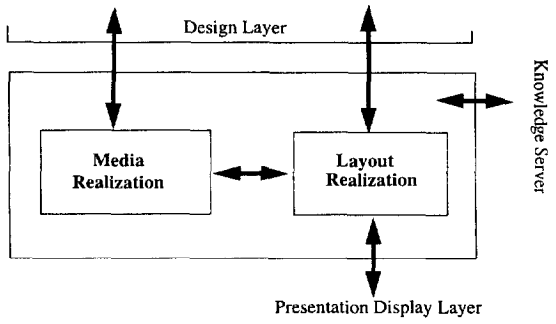


Fig. 8. The realization layer.

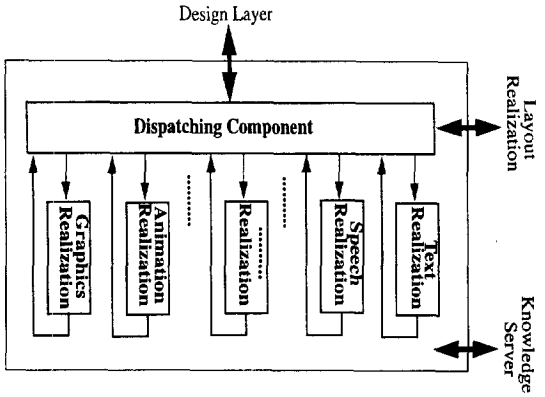


Fig. 9. The media realization component.

nents in an IMMPS. For example, a Media/Modality-Specific Design component may rely on several Media/Modality-Specific Realization components to convey design plans, using them in alternation or even in parallel.

It is beyond the scope of the current SRM to describe the Realization Layer's components in detail. However, we believe that it is important for the realization subcomponents to distinguish whether the transformation of realization commands to media objects is handled as a *retrieval task* or as a *generation task*. If it is a retrieval task, the commands serve as descriptors that must be matched against descriptors of available media objects. (See the literature on multimedia retrieval and multimedia databases for a more detailed treatment of the retrieval task [18].) In contrast, in a generative approach, media realization commands trigger and constrain the mechanisms employed for generating media objects.

The result of the Realization Layer is a specification of displayable media objects together with layout information. It entails all the information required to 'run' the presentation properly. To represent this information, one could use a specification language as proposed in [19], or rely on an object oriented approach for describing multimedia objects such as MHEG [20], or the more general PREMO [8] framework. Note that the output of the Realization Layer cannot be said to be the *complete* internal representation of a presentation. Rather, it is only one piece of the representation, since the Realization Layer has only a restricted view of the presentation process and may be asked to achieve goals that are less complex than the overall one. On the other hand, there is a clear distinction between the media objects as such and their presentation (or display). This distinction is reflected in the SRM by separating the Realization Layer from the Presentation Display Layer that follows it. Moreover, it might be the case that presentation generation (as described by the first four layers of the SRM) is completely separate from presentation display because these tasks might occur at different times and places, and on different unconnected machines; for example, consider the automated generation of a multimedia presentation that will be distributed to users on a CD-ROM.

4.5. Presentation Display Layer

As the output of the Realization Layer is an unrendered presentation, at least one additional module is needed that takes this representation as input and converts it into a presentation that is perceivable by the user. This part of a presentation system is often called the 'presentation display component' or 'presentation runtime environment.' The SRM's Presentation Display Layer, shown in Fig. 10, embodies this functionality. An Output Coordination Component dispatches each part of the unrendered presentation to the interface of a suitable Media-Specific Display component. The Media-Specific Display components serve as abstract references to any of the devices that can be used for the display of a media object of a certain type.

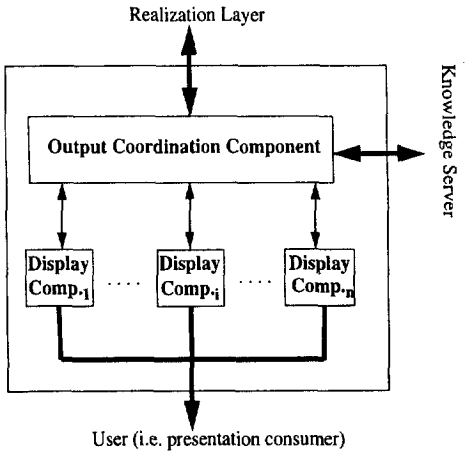


Fig. 10. The presentation display layer.

In Fig. 10, the arrows emanating from the Media-Specific Display components are merged to illustrate the fact that the result presented to the user is the spatiotemporal *coordination* of these outputs. Note that solving this task may require intelligent decision making, because resource limitations must be taken into account. Therefore, the Presentation Display Layer also has access to the SRM's Knowledge Server.

As reference models for multimedia runtime environments are currently under development, we do not repeat this work here. Instead, our strategy is to refine the Presentation Display Layer with appropriate concepts from emerging standards such as PREMO.

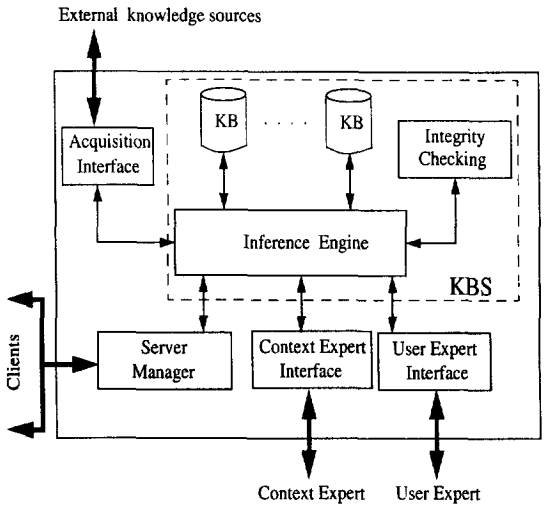


Fig. 11. General structure of an expert module.

5. Knowledge Server

The *Knowledge Server* provides the layers of the SRM with several types of knowledge. It consists of four *expert* modules, designed along the lines of knowledge bases. Each of them represents knowledge on a particular aspect of the presentation process: *application*, *user*, *context*, and *design*. They all share the same general structure.

5.1. Generic structure of expert modules

The overall structure of an expert module is not specific to the class of IMMPSs. Rather, the notion of *expert module* is generic and suitable for other applications as well. Each expert is accessed by other components using a client–server approach. Requests will be granted taking into account the user model and the context. Experts may require/provide services from/to external knowledge sources, such as the application and the user.

Fig. 11 shows the overall logical structure of a generic expert module. The core of the expert module is the knowledge it represents. This knowledge can consist of a number of logically distinct Knowledge Bases (depicted by cylindrical icons in Fig. 11), each for a logically different aspect of the expert's knowledge, or only a single Knowledge Base, when knowledge partitioning is not needed. The Inference Engine provides a uniform and general view of the stored knowledge and of that which can be inferred from it. Maintenance involves incorporating new knowledge, which could be inconsistent with the current state. If this is the case, inconsistency must be resolved by some triggered action performed by an Integrity Checking component.

An expert provides services to other components of the system through its *Server Manager*. Its clients are the layers of the IMMPS, other experts, or external entities (i.e., the External Clients or the application). The Server Manager transforms the interface operations into (a collection of) messages to the Inference Engine, collects the answers, and responds to the client. Two interfaces allow the Inference Engine to acquire knowledge from the User Expert and the Context Expert. Finally, the Acquisition Interface supports access to other undefined servers. This is necessary when the knowledge of the expert depends on other factors in addition to the user and context. For example, the Application Expert has to interact with the application to request application data.

It is often useful and intuitive to have a logically structured view of a knowledge base, as indicated in Fig. 11. Real systems are not required to satisfy this logical distinction in an implementation. For instance, the implementation could combine all the knowledge in a single knowledge base, or structure it under other points of view. We also stress that this SRM is not concerned with the proposal of specific

formats, languages, or protocols for the representation of knowledge or the exchange of knowledge between a knowledge base and other components. Such issues are treated, for example, in the ongoing standardization effort for KQML [9].

5.2. Expert modules of an IMMPS

The knowledge present in an IMMPS may be located in a number of conceptually distinguishable knowledge bases that are difficult to explicitly characterize. However, based on the design of previous IMMPSs, some knowledge bases seem to be indispensable and therefore are represented here as *shared* resources rather than as *local* or *private* resources. These resources are four expert modules in the Knowledge Server that represent knowledge of the application domain, the user, the context, and the design tasks.

The knowledge present in an IMMPS often includes explicitly encoded knowledge sources other than those listed here. For example, when designing a presentation system as a self-reflective system, one may add a knowledge source that comprises metalevel knowledge. However, for the purpose of this SRM, we will classify a multimedia presentation system as an *intelligent* system, if it exploits knowledge sources, as those mentioned above, to achieve presentation goals.

5.2.1. User expert

The User Expert maintains the knowledge that forms the IMMPS's user model. As suggested by [21,22], a user model may include representations of a user's:

- goals and plans;
- physical or mental abilities;
- attitudes and preferences;
- knowledge and beliefs.

With the current SRM, however, we only intend to provide a conceptual location for such knowledge. Issues concerning completeness of user models, representation forms and reasoning processes are not addressed. However, it is worth mentioning that there are already multipurpose tools for building and maintaining user models (e.g., the user modeling shell system BGP-MS [23]), and a standardization effort for user modeling is underway within the research community.

The User Expert's clients are the IMMPS's layers and all the other experts, since there are many decision points in presentation generation process at which the user model must be taken into account. For example, presentation content should be selected according to the user's information needs, and the determination of which media to use should be made based on the user's perceptual abilities and preferences.

5.2.2. Application Expert

The Application Expert provides the IMMPS with application-specific knowledge. The following tasks are assumed to be carried out by this component:

- interfacing with the application system(s);
- information/data format conversion;
- provision of information/data;
- information/data characterization.

The first two tasks always occur when information has to be transferred from one system to another, say from an application to an IMMPS. By 'information/data provision,' we mean the process of making accessible the pool of information from which content can be selected. This task may involve reasoning processes and thus goes beyond pure information/data storage. Finally, we assume that the Application Expert is able to characterize the incoming information/data.

5.2.3. Context Expert

The Context Expert serves as a container for knowledge concerning the context. It may be structured into generation context and presentation context.

Generation context, i.e., a representation of what has been generated so far. For example, the context expert may comprise a lookup table that contains the encoding relations holding between the media objects and their underlying semantic concepts.

Presentation context, i.e., a representation of what has been presented to the user so far, and, if applicable, in which way the user has interacted with (interactive) parts of the presentation.

The task of the Context Expert is similar to the role a discourse model plays in a natural language generation system or a dialogue system. In particular, this component is responsible for the resolution of context-dependent references.

5.2.4. Design Expert

The Design Expert complements the other experts in the sense that it provides a container for all other knowledge which: (a) is relevant for decision making in an IMMPS, (b) does not fit into one of the above expert modules, and (c) should be modeled as a shared source as it will be accessed for solving different tasks. It may include: Media/Modality Model; Design Constraints; Device Model—a (partial) model of the computational environment; for example, comprising knowledge about the characteristics and availability of the output(/input) devices.

5.3. Acquisition of knowledge

The proposed SRM does not make particular assumptions about how knowledge is fed into an IMMPS. With regard to the expert components of the knowledge server, we note that there are several different ways to fill them with knowledge. As a portion of the knowledge stored in the expert modules is static in nature, it may be directly coded by the system builder, or read in only once from an external source (via the Acquisition Interface). However, it may also become necessary to acquire some knowledge during runtime (e.g., from the user).

In an IMMPS such interactions can be handled in at least two different ways. One approach is to equip an expert's Acquisition Interface with its own user interface, so that the module can itself initiate a clarification dialogue. For example, there are systems that pop up forms for the user to fill in when some knowledge is missing. While it has the advantage of being simple, this method has the drawback that the clarification dialogue is not well integrated into the presentation, and thus can be rather disturbing. A more ambitious approach to the problem is to handle the acquisition of missing knowledge as an additional presentation task. Basically the idea is to design an interactive presentation in such a way that the user will provide the missing information when interacting with the presentation. For example, if the user can walk around a 3D graphics scene that is part of a multimedia presentation, the system may learn about the user's information needs and viewing preferences without explicitly asking her/him.

6. Flow of information in the SRM

In the previous sections, not much has been said about the flow of information in the SRM, including control, knowledge, and intermediate representations employed during generation. These details have been omitted because the current SRM is not intended to promote a particular processing model. While the builder of an actual IMMPS may have reasonable arguments for choosing a particular architecture, we remind them that the dependencies that exist between the SRM's layers and components must be taken into account by whatever processing model is adopted.

6.1. The pipeline view

Structuring the SRM into a pipeline of five layers implies that the flow of information in an IMMPS follows the pipeline. At a first glance, this is the case. Presentation goals and commands form the triggering input to an IMMPS. As they are processed through the layers, goals/commands are transformed into presentations composed of media objects to be displayed to the user. Thus the flow of information during this transformation is primarily 'top-down.' On the other hand, many interactions can occur between the layers. As indicated by the notification arrows in Fig. 12, the SRM also supports bidirectional interactions between layers. Each layer can

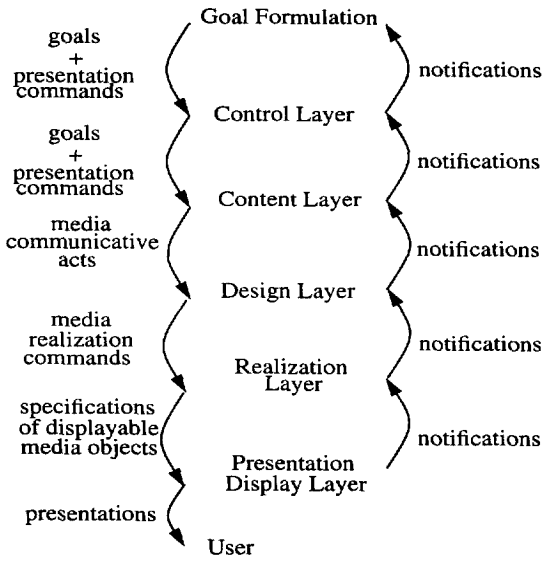


Fig. 12. Information flow through the layers of the SRM.

notify the layer above it about the result of its processing, possibly together with additional information, such as causes of failure, recovery strategies, or explanations. For example, the task division between Goal Selection and Goal Refinement requires bidirectional information flow between the Control Layer and the Content Layer (see Sections 4.1 and 4.2. Alternatively, consider the generation of cross-media references. To make a textual reference to the location of a picture or to part of a picture, the Content Layer needs to know the spatial attributes of the particular location. However, these attributes may become available only after the layout has been realized. Thus the Content Layer has to wait for notification from the Layout Realization component before it can accomplish the reference. Generally speaking, bidirectional communication between layers becomes necessary when decision making in a layer also depends on decisions to be made in one of the layers shown in Fig. 12.

6.2. *Information flow in real IMMPSs*

Like the SRM, the pipeline view abstracts from actual system implementations, too. That is, in a real system, the flow of information not only depends on the logical task structure but also on the particular mechanisms employed for generation. As generation entails many combinatoric issues, most existing IMMPSs rely on a *generate-and-test* approach, at least for solving some of the generation subtasks. The basic idea is to have a mechanism that allows the withdrawal of earlier decisions (e.g., a choice among alternative candidates) in case the proceeding computation does not lead to satisfying results, or any result at all. For example, a system may have generated a graphic that turns out not to fit properly into the overall presentation layout. To fix the problem, the system may withdraw the layout design, the graphics design, or may even decide to use text for presenting the information instead of graphics.

To precisely characterize the flow of information in a concrete system, one has to look at: (a) decisions that are subject to backtracking; (b) the architectural organization of the components performing the decision making; (c) the generation, testing, and backtracking mechanisms themselves, as this can be done in a more intelligent way than exhaustively enumerating combinations [24,25].

7. Discussion

This article has outlined a reference model for IMMPSs that is meant as a step towards broader agreement among members of the scientific and industrial community. The core of the resulting SRM is a generic reference architecture that may be summarized by the equation:

IMMPS = Layers + Experts.

The design of this architecture reflects a decomposition of multimedia generation into logically distinct subtasks (represented as Layers) and a separation of these tasks from the knowledge sources (called Experts) that are exploited to accomplish them.

7.1. *Using the SRM*

The current model is intended to provide a framework for analyzing and comparing existing IMMPSs, and guidance for the development of new ones. For illustration, we derive from the SRM a list of questions that one may answer to set up a profile for some existing concrete system S.

Input:
- What are the presentation goals and commands to be handled by S?
- What kinds of application data/knowledge are assumed?
- Are there any further input parameters for S?

Output:
- What kinds of presentations are delivered?
  - Which types of media objects, and which combinations?
  - Which modalities?
- Is the realization of media objects uncoupled from displaying them?

Generation:
- Which of the subtasks specified by the layers (and their components) are handled by S?
  - Which mechanisms are used to handle them?
  - Does S assume a fixed order in which subtasks are carried out, or is the ordering dynamically determined based on some knowledge?

- How well are these tasks performed?
- Which knowledge sources are exploited?
- Which sources are private, which are shared?

...

Performance:

- To what extent are variations in the input to S reflected as variations in the output?
- How much interaction between system components is required when generating cross-media references?

...

When describing a complex IMMPS with the SRM one may realize that there is not always a one-to-one mapping from system modules to the layers of the SRM and their components. For example, some concrete implementations of IMMPSs incorporate media-specific generation components that can also be operated in a stand-alone fashion. Such generators actually do some goal decomposition and content selection by themselves, and thus perform tasks located outside the SRM's Design and Realization Layers. For example, COMET's graphics generator IBIS [26] selects objects for inclusion in the generated pictures (e.g., ones that were needed to disambiguate objects or avoid false visual implicatures). To describe such IMMPSs strictly in terms of the SRM, one would have to identify these content selection tasks and 'move' them to the SRM's Content Layer. However, in order to achieve a description that is closer to the actual modularization of a system and the distribution of tasks among its modules, one can also treat the generation component itself as a special case of an IMMPS in which media allocation always returns the same single medium. Following this approach, the SRM can be used twice, firstly to describe the overall IMMPS, and secondly to describe embedded generation components and reflect all tasks that are performed locally in these components.

7.2. Possible refinements of the SRM

There are a number of possible refinements and extensions to the current SRM. Two of them are sketched below. Further ones are discussed in the contributions by Hardman et al. [27] and Graf [16].

7.2.1. Moving towards a formal specification

A further phase in the development of a complete formal reference model is to express in a semiformal notation the functionality of each black box component of the system. As an example, consider the Goal Refinement component in the Content Layer. Below we sketch a semi-formal specification of parts of its interactions (in the style of CSP [28]).

The specification expresses the following: The Goal Refinement component nondeterministically receives goals from the Control Layer's Goal Selection component (line 1). Received goals are decomposed (2) and stored in the context knowledge in the Context Expert (3).

If the Content Selection component is ready to proceed with a new goal (4), and the Media Allocation component is also ready (5) and there is such a new goal reported from the Context Expert (6), then the goal will be actually forwarded to the Content Selection (7) and Media Allocation (8) components.

Goal Refinement also receives *notifications* (9), from the Ordering component. Notifications are analyzed: if they report success (10) then they are elaborated (11) and eventually propagated upwards to the Control Layer (13). If any problem occurred then a repair strategy should be determined (14) which is pushed into the Context Expert (16). Finally, if no repair strategy is applicable, then an elaborated version of the notification (17) will be forwarded to the Control Layer (18).

□ Control Layer?*Goal*(*goal*)	(1)
→	
goals: = *Decompose*(*goal*);	(2)
ContextExpert!*Push*(*goals*)	(3)
□ ContentSelection?*Ready*(),	(4)
MediaAllocation?*Ready*(),	(5)
ContextExpert?*NextGoal*(*goal*)	(6)
→	
ContentSelection!*Goal*(*goal*);	(7)
MediaAllocation!*Goal*(*goal*)	(8)
□ Ordering?*Notification*(*not*)	(9)
→	
if OK(*not*) *then*	(10)
not': = *Elaborate*(*not*);	(11)
if Notifiable(*not'*) *then*	(12)
ControlLayer!*Notification*(*not'*)	(13)

endif
else
if Repairable(not) then (14)
rep: = *Repair*(*not*); (15)
ContextExpert!*Push*(*rep*) (16)
else
not': = *Construct Notification*(*not*); (17)
ControlLayer!*Notification*(*not'*) (18)
endif
endif
□
.
.
]

7.2.2. Moving towards multimedia dialogue systems

As already emphasized in Section 1, the current SRM is rather limited in its abilities to handle user interactions. To be more precise, the SRM covers only interactions of the following types: direct triggering of presentation goals, and presentation unrolling.

Direct triggering of presentation goals, i.e., there is a direct mapping from a user input to a presentation goal. For example, the system may provide menus from which the user can select presentation goals for the system to accomplish.

Presentation unrolling, i.e., the system designs a branching presentation structure and lets the user determine how this structure is traversed. Examples of such presentations include hypermedia documents with statically linked parts, as well as presentations in which the target of a link is generated when the link is selected.

A full-fledged multimedia dialogue system would allow the user to enter information through the full range of supported modalities and media. Such interactions are obviously beyond the scope of the current SRM. However, if one considers multimedia analysis as the inverse of multimedia generation (see also Ref. [29]), one can obtain a rough task decomposition just by turning the SRM upside down, and replacing the generation subtasks by analysis subtasks [6]. For example, at the counterpart to the Presentation Display Layer one would find media input devices and a component that tracks the spatiotemporal relations of user input in different modalities. Medium/modality-specific analyzers will be found at the layers that correspond to Realization and Design. Instead of Media Allocation, one would have a component for fusing media-specific communicative acts in order to bring about the underlying semantics and eventually the user's communication goals. Future work will show to what extent such a *reversibility* approach will lead to useful results.

Acknowledgements

The work described in this article has been carried out under the auspices of the ERCIM Computer Graphics Network, with financial support from the EU Human Capital and Mobility Programme (contract CHRX-CT93-0085).

References

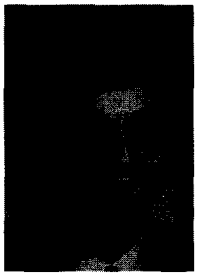
[1] M. Maybury (Ed.), Intelligent Multimedia Interfaces, AAAI/The MIT Press, 1993.
[2] F. McCaffery, M. McTear, A multimedia interface for circuit testing in a shop floor environment, Standford Univ., CA, March 21–23, 1994, pp. 49–53.
[3] G. Herzog, Fluids annual review report (April, 1997), deliverable issue no. 102/04/97, IV Framework EU Project FLUIDS (TE 2006), 1997.
[4] G.E. Pfaff (Ed.), User Interface Management Systems: Proceedings of the Seeheim Workshop, Springer Verlag, Berlin, 1985.
[5] ARCH, A metamodel for the runtime architecture of an interactive system, The UIMS Developers Workshop, SIGCHI Bull. 24 (1) (1992).
[6] S. Feiner, An architecture for knowledge-based graphical interfaces, Intelligent User Interfaces, ACM Press/Addison-Wesley, Reading, MA, 1991, pp. 259–279.
[7] Computer Graphics Reference Model, International Standard Organization, ISO/IEC IS 11072, 1992.
[8] I. Herman, G.J. Reynolds, J. Van Loo, PREMO: An emerging standard for multimedia presentation, IEEE MultiMedia 3 (3) (1996) 83–89.
[9] Y. Labrou, T. Finin, A Proposal for a new KQML specification, TR CS-97-03, Computer Science and Electrical Engineering Department, Univ. of Maryland Baltimore County, Baltimore, MD 21250, February 1997 (See also: http://retriever.cs.umbc.edu:80/kqml).
[10] PREMO, Presentation Environment for Multimedia Objects (PREMO), International Standard Organization, ISO/IEC 14478, (see also: http://www.cwi.nl/Premo), 1997.

[11] D.J. Duke, I. Herman, T. Rist, M. Wilson, Relating the primitive hierarchy of the PREMO standard to the standard reference model for intelligent multimedia presentation systems, Comput. Stand. Interfaces 18 (6,7) (1997) 527–537.
[12] N.O. Bernsen, Defining a taxonomy of output modalities from an HCI perspective, Comput. Stand. Interfaces 18 (6,7) (1997) 539–551.
[13] J.R. Searle, What is a speech act?, in: M. Black (Ed.), Philosophy in America, Allen & Unwin, 1965, pp. 221–239.
[14] E. André, T. Rist, Towards a plan-based synthesis of illustrated documents, Proceedings of the 9th ECAI, Stockholm, Sweden, 1990, pp. 25–30.
[15] M.T. Maybury, Planning multimedia explanations using communicative acts, in: M. Maybury (Ed.), Intelligent Multimedia Interfaces, AAAI/The MIT Press, 1993, pp. 60–74.
[16] W. Graf, Intelligent multimedia layout: A reference architecture for the constraint-based layout of multimedia presentations, Computer Standards Interfaces 18 (6,7) (1997) 517–526.
[17] M. Friedell, Automatic graphics environment synthesis, PhD thesis, Case Western Reserve Univ., Cleveland, OH, Computer of America Technical Report CCA-83-03, 1983.
[18] M.T. Maybury (Ed.), Intelligent Multimedia Information Retrieval, AAAI/The MIT Press.
[19] L. Hardman, D. Bulterman, G. van Rossum, The Amsterdam hypermedia model: Adding time and context to the Dexter model, Commun. ACM 37 (2) (1994) .
[20] MHEG, Coding of Multimedia and Hypermedia Information, International Standard Organization, ISO/IEC IS 13522, 1995.
[21] R. Kass, Building a user model implicitly from a cooperative advisory dialogue, User Model. User-Adapted Interact. 1 (3) (1991) 203–258.
[22] T.W. Finin, GUMS—a general user modelling shell, in: A. Kobsa, W. Wahlster (Eds.), User Models in Dialog Systems, Springer Verlag, Berlin, 1989, pp. 411–430.
[23] A. Kobsa, W. Pohl, The user modeling shell system BGP-MS, User Model. User-Adapted Interact. 4 (2) (1995) 59–106.
[24] Kenneth D. Forbus, Johan De Kleer, Building Problem Solvers, MIT Press, Cambridge, MA, 1993.
[25] Richard M. Stallman, Gerald J. Sussman, Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis, Artificial Intelligence 9 (2) (1977) 135–196.
[26] D.D. Seligmann, S. Feiner, Automated generation of intent-based 3D illustrations, Proceedings of ACM SIGGRAPH '91, Las Vegas, NV, 1991, pp. 123–132.
[27] L. Hardman, M. Worring, D. Bulterman, Integrating the Amsterdam hypermedia model with the standard reference model for intelligent multimedia presentation systems, Computer Standards Interfaces, this issue, 1998.
[28] C.A.R. Hoare, Communicating Sequential Processes, Prentice-Hall, Englewood Cliffs, NJ, 1985.
[29] M.T. Maybury, Research in multimedia parsing and generation, in: P. McKevitt (Ed.), Artificial Intelligence Rev. 9 (2–3), 1995.

Monica Bordegoni holds a Laurea degree in Computer Science from the University of Milano. She is currently employed as researcher at the Italian National Research Council, and is lecturer for the course of Computer Graphics at the University of Parma and University of Como. Her research interests include human–computer interaction, user interface development tools, multimodal interfaces, novel input devices, haptic interaction, visual languages, multimedia and hypermedia, virtual reality and CAD.

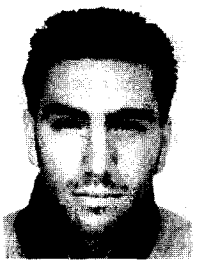
Dr. Ing. Giorgio Faconti received a Laurea in Mechanical Engineering and a Post-Laurea degree in Computer Science from the University of Pisa. In 1974 he joined CNR-CNUCE, an Institute of the National Research Council of Italy, where he holds the position of Vice-Director and leads the Department of Design and Analysis of Computing Systems. He is also a member of the Scientific Advisory Committee of the Institute, and Deputy Member of Executive Board of European Research Consortium in Informatics and Mathematics. He has been involved in several research projects, holding both management and research positions. Specific project experience includes Computational Geometry, Distributed Graphics, Industrial Applications of Mathematics, Technology Innovation in SME, Human–Computer Interface Design. Currently is the coordinator of Task 2: Computer Graphics and Knowledge Engineering of the ERCIM Computer Graphics Network, that has developed the Reference Model for Intelligent Multimedia Presentation Systems. His current research interests include the application of formal methods to the specification of multimedia systems, interactive system design, and reasoning about human–computer interaction.

Dr. Steven Feiner is an Associate Professor of Computer Science at Columbia University, where he directs the Computer Graphics and User Interfaces Laboratory. He received a PhD in Computer Science from Brown University. His research interests include knowledge-based design of graphics and multimedia, user interfaces, virtual worlds and augmented reality, visualization, animation, visual languages, image synthesis, and hypermedia. Prof. Feiner is coauthor of *Computer Graphics: Principles* and Practice (Addison-Wesley, 1990) and of *Introduction to Computer Graphics* (Addison-Wesley, 1993). He is an associate editor of *ACM Transactions on Graphics*, and is a member of the editorial board of *IEEE Transactions on Visualization and Computer Graphics*, and of the executive board of the IEEE Technical Committee on Computer Graphics. In 1991 he received an Office of Naval Research Young Investigator Award.

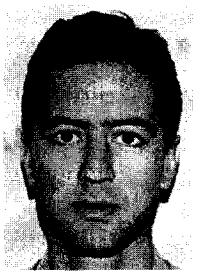


Mark Maybury received his BA in Mathematics from the College of the Holy Cross in 1986 where he was valedictorian. As a Rotary Scholar at Cambridge University, England he received his M.Phil. in Computer Speech and Language Processing in 1987 and his PhD in Artificial Intelligence in 1991 for his dissertation, 'Generating Multisentential Text using Communicative Acts.' Mark was awarded an MBA from RPI in 1989. Mark has published over fifty technical and tutorial articles in the area of language generation, multimedia presentation and intelligent multimedia information retrieval. He has given tutorials on intelligent multimedia interfaces at multiple international conferences, including CHI, AAAI, IJCAI, COLING, and ACM Multimedia. He chaired the AAAI-91 Workshop on Intelligent Multimedia Interfaces, the IJCAI-95 Workshop on Intelligent Multimedia Information Retrieval and co-chaired the EACL/ACL '97 Workshop on Scalable Text Summarization. Mark is editor of *Intelligent Multimedia Interfaces* (AAAI/MIT Press, 1993), *Intelligent Multimedia Information Retrieval* (AAAI/MIT Press, 1997) and co-editor of the forthcoming *Readings on Intelligent User Interfaces* (Morgan Kaufmann Press). Mark is MITRE's Corporate Thrust Leader for New Information Services and leads the Defense Information Infrastructure Common Operating Environment Multimedia and Collaboration Working Group. He currently is Deputy Division Manger for National Intelligence Division and Director of MITRE's Advanced Information Systems Center, where he leads a set of seven strategic technology sections spanning the disciplines of intelligent information systems, collaborative applications, distributed computing, networking, and data and information management.

Dr. Thomas Rist is a Senior Researcher at the department Intelligent User Interfaces of the German Research Center for Artificial Intelligence (DFKI). Dr. Rist studied computer science and mathematics at the University of the Saarland. He undertook his doctorate in the area of knowledge-based graphics generation and received the degree from the University of the Saarland. He was involved in various projects funded by the German Ministry for Education and Research, industries, and the EU. One of these projects, WIP, was a winner of a 1995 Information Technology Award (ITEA). As a Senior Scientist he was also in charge of management tasks in a number of industrial projects, and is currently coordinator of the European LTR Project Magic Lounge within the i3 initiative on Intelligent Information Interfaces. His areas of interest and experience include: multimedia/multimodal communication, intelligent user interfaces, automated presentation systems, knowledge-based graphics generation, life-like characters, and user modelling. Dr. Rist has served as a programme committee member for national and international conferences and workshops, and was on the organization board of a number of workshops on multimedia systems. In 1995 Dr. Rist became a member of the ERCIM Computer Graphics Network Task 2 Group.



Salvatore Ruggieri is presently a PhD student at the Department of Computer Science, University of Pisa, where he also received his MS in 1994. In 1995 he spent four months at Rutherford Appleton Laboratory in UK, working on the Computer Graphics Network Task 2 Project: design of a reference model for intelligent multimedia presentation systems. His research interests include formal methods, logic programming, multimedia systems, software quality. He has been co-organizer of workshops on those topics, and reviewer of scientific papers for international journals and conferences.



Panos Trahanias is an Assistant Professor at the Department of Computer Science, University of Crete, Greece, and a Senior Researcher at the Institute of Computer Science, FORTH. He holds a BS in Physics from Athens Univ., Greece, and a PhD in Computer Science from the National Technical Univ. of Athens, Greece. He has held positions as Research Associate with the Institute of Informatics, NCSR Demokritos, Greece (1990–91) and with the Department of Electrical and Computer Engineering, University of Toronto, Canada (1991–93). His current research interests are in the areas of multimedia presentation systems, computer vision and pattern recognition, and man–machine interaction. He has participated in many R&D programs at the University of Toronto and FORTH and has been a consultant to SPAR Aerospace, Canada. He is an active member of the HCM ERCIM Computer Graphics Network and the TMR Network VIRGO. He has published over 40 papers and has contributed in two books.

Michael Wilson obtained a bachelors degree in Experimental Psychology from the University of Sussex, and a doctorate in Psycholinguistics from the University of Cambridge. After 1983 he studied how users learn to use windowing systems on a project funded by IBM at the MRC Applied Psychology Unit, Cambridge. Since 1986 he has worked at the Rutherford Appleton Laboratory researching intelligent user interfaces incorporating multiple interaction modes, automatic generation of presentations, ontology based information retrieval, knowledge acquisition, multimedia information retrieval and presentation, and conversational interaction. He is a member of the BCS and ACM, and has served on the programme committees of many international conferences, and several journal editorial boards on HCI and AI. He has acted as a research programme advisor and reviewer for UK and European research programmes, and is currently coordinator of the UK EPSRC programme on Multimedia Networking Applications.