Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

INS

Information Systems

INformation Systems

Design dependencies within the automatic generation of hypermedia presentations

O. Rosell Martinez

REPORT INS-R0205 JUNE 30, 2002

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3681

Design Dependencies within the Automatic Generation of Hypermedia Presentations

Oscar Rosell Martinez
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

ABSTRACT
Many dependencies appear between the different stages of the creation of a hypermedia presentation. These dependencies have to be taken into account while designing a system for their automatic generation. In this work we study two of them and propose some techniques to treat them. During this process, we introduce the partial design of a hypermedia formatting model that we call Hypermedia Formatting Objects.

1998 ACM Computing Classification System: H.5.4, H.5.1
Keywords and Phrases: Hypermedia, Multimedia, Attention Characteristics, Formatting Objects.
Note: The research reported here has been carried out under the "ToKeN2000 I2RP" and "Dynamo" projects.

1. Introduction

Since its creation in the early nineties, the number of people accessing the World Wide Web (the Web) has increased every year. Until this moment, the primary interface to the Web has been the personal computer. However, as explained in [51], presumably the number of devices accessing it will grow in the years ahead, and it will become necessary, for content producers, of being able to provide quality content for this diversity of devices. This will mean that an automatic way of adapting the content to the devices is desirable.

The low impact of the second generation of mobile phones in Europe is mostly explained by two factors that feed each other: the poor capabilities of this kind of device and the small number of documents available on the web for them, making users avoid using them. With the future appearance of the third generation of mobile communications technology, with higher technical specifications, the first factor should be overcome, and the problem will be the second, the availability of compatible content.

Although some of the content can be adapted client-side, the quality of the output cannot be warranted. In the case of a multimedia application, this adaptation is difficult to carry out without decreasing the resemblance to the original. The solution for content providers right now, is to double the design work in order to have a specific version of the content for each kind of device accessing it.

With the increase of different devices, with different specifications, the number of covered possibilities will also have to grow. In consequence, many versions will have to be made in order to satisfy the same audience that now can be served with just one. The companies will have to balance the cost of creating different presentations for each possible device with the loss produced by not allowing the device to access the content (or supplying an unadapted version). Automating the generation process is a solution.

But content adaptation is not only about devices but also, and possibly more important, about people. Not every language is the same, not every culture is the same, not every person is the same. Therefore, not all content should be the same for every person. As mentioned in [52], the content should be adapted to the user's needs (what he wants to accomplish), preferences (how he wants the content), and capabilities (what he is able to do and what he is not), instead of being the same for everybody. The importance of this adaption can be seen in two ways. It is important for the user, as

it makes the task to find what he is looking for easier, in the way he wants and needs, and it is also very important for the content provider, as this tailoring can add value to the content and can lead to the client's fidelity [34].

It is in this context where the automatic generation of tailored multimedia applications can be very appealing, and a successful system would be really useful.

This future blooming of web enabled devices is anticipated by the scientific community. A proof of this statement is the creation of a W3C working group that has designed a framework, CC/PP [40], an RDF [53] based document format and protocol, to solve part of the problem: how the technical specifications of the devices and the personal preferences of the users should get to the servers.

How to treat all this information and generate adapted multimedia applications, however, is still mostly a matter of research.

1.1 Introduction to Cuypers

In the remaining of this section we will explore the architecture of an experimental instance, the Cuypers system [50], developed by the Multimedia and Human-Computer Interaction group at the Centrum voor Wiskunde en Informatica (CWI) in Amsterdam. At the end of the section, we will explain the motivation and the structure of this work.

Cuypers is an experimental software system developed at CWI (Amsterdam) that automatically generates tailored multimedia presentations, trying to fulfill both adaption requirements: technical adaptation (devices, networks, ...) and personal adaptation.

Architecture The system's layered architecture is exhaustively described in [50]. Here we will only make a gentle introduction to it.

Cuypers is conceptually divided into five levels:

1. *Final-form presentation level*: this is the lowest level of the system. In this layer, the internal representation of a presentation in Cuypers is transformed into a real, widely spread, client-side playable multimedia presentation format.

2. *Quantitative constraints level* [26]: An abstraction over the final form. The temporal and spatial layout is determined in a quantitative way. The exact position of the media elements is determined at pixel detail. The adaptation process is mostly concluded: it only remains to write the presentation in a presentation format playable at the client.

3. *Qualitative constraints level* [26]: An example of a qualitative constraint is "object A is at the left of object B". At this level, the layout is specified as relations between the elements that appear in the presentation, not as absolute, pixel-precise positions.

4. *Communicative device level*: Communicative Devices [45] are multimedia design patterns. They describe the presentation in terms of well-known spatial, temporal and hyperlink presentation constructs that try to convey a meaning, a message.

5. *Semantic structure level*: The topmost level of the system, describes the presentation in terms of semantic relations between the elements that will take part in it (media items).

Implementation The key concepts of Cuypers' implementation are:

- *Rhetorical Structure Trees*: From [37]:"Rhetorical Structure Theory suggests that discourse has an ideational structure or hierarchy which depends upon the speaker's communicative intent (the speaker's discourse purpose). To capture this ideational structure discourse is marked-up with ideational relations (rhetorical predicates)". Therefore, a rhetorical structure tree (RST) is a tree that tries to convey this semantic hierarchy. Although Rhetorical Structure Theory was created for text analysis, it can also be effectively used for generating multimedia presentations as discussed in [44] and [3].

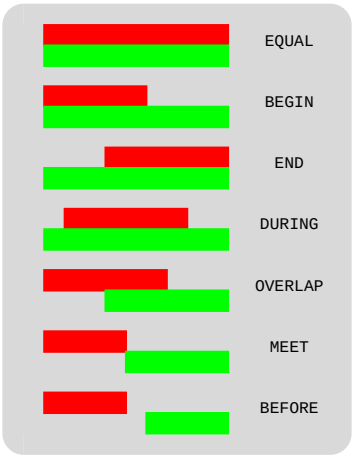


Figure 1: Allen relations. Only seven of the thirteen are depicted, as six of them are just the inverse (e.g., A *during* B $\Leftrightarrow$ B $during^{-1}$ A).

- *Communicative Devices*: use of this abstraction in the implementation, as the RST is transformed into a communicative device tree (CDT). They permit the system to be able to generate very different presentations from the same semantical relations, depending on which communicative devices are chosen and deployed.

- *Allen Relations* [2]: James Allen described in his paper an interval algebra for temporal reasoning. This algebra is based in time intervals and a set of binary relations between them (β). Given two intervals, the relative positions between them can be described with one and only one element of β, where $\beta = \{\ before, before^{-1}, meet, meet^{-1}, overlap, overlap^{-1}, begin, begin^{-1}, during, during^{-1}, end, end^{-1}, equal\ \}$. In Cuypers, the qualitative relations between the media items are described in terms of Allen relation, for the three dimensions (x, y, time). Although, they were originally intended for temporal modeling, they can be successfully applied in layout dimensions (i.e. x,y) [26].

- *Constraint Solving Programming*: Constraint-based methods have been extensively used for automatic layout [36], and Cuypers is not an exception. An introduction to Constraint Solving Programming and an explanation of how it is used in Cuypers can be found in [25]. In a few words, the relative positions between the media items are stated one by one (qualitative constraints) for every dimension. These qualitative constraints are transformed into numerical relations between the starting and ending points of every element, in every dimension (quantitative constraints). For the implementation, ECL^{i}PS^e [56], a general Constraint Logic Programming System, is used in order to check the integrity of the constraint set and, at the end, calculate the exact position of every media item in a three-dimensional space (where time is the third dimension) by solving a finite domain constraint system.

- *Backtracking*: Every time a constraint is added to the system, the constraint-solver checks if the whole set is consistent. That is, if exists an specific disposition of the elements on the screen and on time, that does not violate any of the stated constraints. If at any point, the constraint set becomes inconsistent, the system goes to the last time he chose between options, and chooses another one. Backtracking can make the system go back and reconsider an option in higher layers (e.g. choosing another communicative device). This ends with a more or less modified set of constraints, that are evaluated to check for consistency and so on. The backtracking is a

basic feature of Prolog [13] on which ECL^{i}PS^e is based.

- *Unification*: Unification is a concept closely related with Logic Programming in general, and with Prolog in this case in particular. In Cuypers, it is used as the main branching technique: the data entered is matched (unified) with the existing rules, and the program execution evolves that way, unifying and applying the matched rules.
- *SMIL*: for this first implementation, SMIL was the chosen option for the final presentation format because of its expressive power, that covered everything that Cuypers can generate internally, and the fact that it is a public format recommended by the W3C (World Wide Web Consortium), a non-profit organization.

Problems The creation of a system like Cuypers or, in general, any Intelligent Multi Media Presentation System (IMMPS) [43], is a difficult task that requires merging techniques from different fields of the computer sciences. Database techniques are needed for efficiently storing, querying and extracting the information needed for the media components, artificial intelligence techniques are needed in order to be able to automate all the knowledge and techniques typically used and studied by the multimedia community. These systems are complex, as they try to fulfill many commitments at the same time. The correctness of the generated presentations is evaluated over different axis: aesthetics, informative capacity, usability, interactivity, adaptivity, ... The perfect system, that performs well in all the categories and is still general and fast enough to be really usable, does not exist (yet).

2. Goals of this work

The basic idea on which Cuypers is based is that any presentation has a rhetorical structure that relates the elements that appear in it, and that from this rhetorical information the formatting of the actual presentation can be derived. However, this semantic information of the relations between the elements is necessary but not sufficient to fully design a presentation, and other factors need to be considered.

As stated in [52], there are many interdependencies among the different steps of designing a multimedia presentation. For example, the specific content used can influence the structure and style of the presentation. More concretely, the media types of the elements can suggest and constrain the way they can be displayed. While closely related images tend to be displayed simultaneously on the screen, the same cannot generally be done with videos as the result would be awkward because of the difficulty of paying attention to two videos at the same time. Even if the semantic relation holding between two elements is the same, the nature of the items may change the way they should be displayed. While for a human designer the influence of the media types in the layout is sometimes obvious and is mostly processed at a subconscious level, a machine needs to be instructed to mimic this behavior.

Moreover, these dependencies can also be found within a level. For example, the global style of a presentation determines the formatting of the different parts of it. For efficiency, Cuypers uses mainly an incremental, bottom-up, local algorithm to create the presentations. However, as we have seen, the designing task is not suited for a local approach as many things have to be balanced. A human designer tends to regard the presentation as a whole, not only locally as different parts. In order to maintain a global style and a consistent structure, decisions at any step of the creative process could require modifications in the parts already designed and these should influence the parts that are going to be created. Although we could rely on backtracking for accomplishing global reasoning in Cuypers, by setting constraints affecting the whole presentation, this would be highly inefficient because of the great cost of backtracking and the manipulation of additional constraints. If we want to keep the incremental approach, we require a data structure that holds sufficient information about the parts created in an easily accessible form, and is also flexible enough to be modified as the generation progresses.

Our goals in this work are:

- To set the theory behind media types, media types combinations and infer the information needed to reason at this level.
- To find a method for applying this theory to the system. We need some information about the constructed presentations and rules to infer which layouts should be avoided.
- To find the requirements of a data structure to hold the information about the formatting of a presentation construction.
- To design a feasible option for this data structure.

2.1 Approach taken
This document is structured as follows:

Section 2 We study the information we need about the media types. We make an overview of the main characteristics of each media type and we study their optimal combination in a presentation, all this from the perspective of attention. At the end of the section, we study the application to Cuypers, analyzing the rules and information structures needed and the options we have to implement them.

Section 3 We introduce the concept of a formatting model, and explain how it can be used to describe the layout of the generated structures. We also detail some problems found in the previous version of Cuypers, and how they could be mitigated by implementing an adequate formatting model.

Section 4 We show a partial implementation of a hypermedia formatting model, that we call Hypermedia Formatting Objects, explaining the key elements and reasoning about its benefits. We detail how this implementation helps to solve the problems described in Section 3 and state some difficulties we found while implementing.

Section 5 We make an evaluation of the outcome of our work, commenting the way the techniques developed in our work can help the system to generate better presentations and the shortcomings of our solutions.

3. Attention and Media Types

Our primary goal in this section is to show the importance of attention in a multimedia application, while stating the commonly agreed upon principles of human attention that are relevant for paying attention to a multimedia presentation. These principles will be the rules that will enable us to decide about which combinations of media types are convenient and which ones should be avoided because they require too much effort from the user. In order to do that, we show before some of the main psychological and cultural aspects of each media type, especially the ones related with attention. At the end, we will explain which options we have for including this knowledge into Cuypers in order to avoid these harmful combinations and generate only correct presentations.

We would like to issue a warning at this point. The attention issue, as any psychological one, is complex and, although we have tried to be rigorous, we are aware that there are some things that could be argued against and some aspects missing.

3.1 Multimedia
The term multimedia started to become commonly used in the beginning of the nineties, closely related to the widespread use of sound cards, enabling the computers to generate high-quality, stereo sound, and the popularization of the CD-ROM. The CD-ROM was a key factor as it was a huge improvement from the previous generalized data support, the floppy disk, multiplying by five hundred the capacity of the latter. This permitted the appearance of multimedia creators, that were able to distribute, in a

cheap way, applications including high-quality media: video, sound and high-resolution images. The multimedia concept is, however, much more than the mere union of these different media types.

Multimedia stands for the integration of multiple media (audio, video, ...) into one presentation format. The key of multimedia is not in a simple and plain usage of different media, but in the interaction between them in order to get something better. As the Gestalt psychologists say, "things are affected by where they are and by what surrounds them... so that things are better described as more than the sum of their parts" [6].

As a song is more than the sum of its notes, a multimedia application should be more than the sum of the media elements contained in it. For each thing we want to communicate the most suitable media type should be used. Each media type should be used only for what it is good for, and the combination of them should result in a benefit for the viewer in terms of enjoyability, and information transfer.

The multimedia paradigm is (from [1]):

> " the dominant conviction that adding multimedia functionality to information systems leads to improved information and knowledge transfer to people ".

Psychological studies have proved that when receiving stimulus of only one media type, only part of our brain is working. Therefore multimedia should be the dynamite to break this barrier, allowing us to learn using our capacities to a full extent.

Efficiency of multimedia

Even before the term multimedia was coined, many studies [38] had been made trying to show if multiple-channel communication (the basis of multimedia) can improve knowledge transfer and learning. Many of these studies ended with results that suggested that multiple-channel communication, although more amusing (a factor that should not be belittled), could not show a superiority over single-channel communication in its communicative capacity. Even worse, many of these results showed that using multiple-channel systems could have a negative effect in the learning capacity of the user [38], resulting in less effective instruction than traditional text.

Many other studies [38] have shown that multimedia (or multiple-channel) systems can really be used successfully, incrementing the retention capacity of users and the arousal, and leading to a more satisfactory communication.

The conclusion of these discrepancies should be that a multimedia application can be better than a unimodal one but, for this to be true, the application has to be very well designed; "each medium should be used where it is the best solution for the conveying and processing of information"[35], and the relations between the media have to be very well thought out.

3.2 Attention in Multimedia

In [5] attention is defined as:

> "A process which involves the use of mental effort in selectively processing information from a range of different sources, both internal (including your own thoughts and dreams) and external (including visual and auditory stimuli)"

The implications of attention in multimedia are clear. The combinations of different sources of information, of different media types, that are combined and displayed at the same time are highly demanding for the attention of users [5].

Although the number of theories about human attention is large, due to the fact that the attention capabilities and traits can vary from one person to another, there are some commonly agreed rules that can be applied to most people. Also from [5]:

1. People are able to focus their attention on particular stimuli.

In general, we can focus our attention at will. In a multimedia application that means that if two static items are displayed at the same time, the viewer will be able to pay attention to one, without being disturbed because of the existence of the other.

2. People are able to pay attention to more than one stimulus at any one time.

   Psychological studies have shown that is easy to combine different activities if:

   - The activities are dissimilar.

     Example 3.1 *Listening to the radio or even talking while driving a car is normally easy, because for talking (or listening) the primary sense involved is the aural, while for driving the primary sense is the visual.*

     The Multiple Resource Theory [57] explains it this way: resources are divided in different pools associated with particular input modalities (auditory/visual), processing codes (spatial/verbal) and response modes, and performance is better if the tasks do not require the same resources. This theory is reinforced by some physiological evidence: for different tasks, different parts of the brain are involved.

   - The activities are highly trained.

     Example 3.2 *Reading subtitles is something annoying for a non-trained person but natural for a trained one.*

   - The activities are simple, and do not require large amounts of cognitive resources.

     Example 3.3 *Chewing chewing gum is a classical example of an easy task that does not require the use of any cognitive resources and thus, it can be done while performing complex tasks without a loss of performance.*

3. Attending to information requires mental effort.

   The power of the combination of different media elements of different media types carries a danger. If too much information is presented to the user at the same time, she will be unable to capture it all (this effect is named cognitive overload in [30]).

   As depicted in [38], Hartman distinguishes four possible relations between two media cues: redundant, related, unrelated and contradictory. If they are unrelated or contradictory, they compete with each other, resulting an information interference. On the other hand, if they are redundant or related they will complement each other and will improve comprehension and learning.

   These relations can be summarized in the concept of *level of congruence* [30]:

   > "The level of congruence is the degree to which different information types are used redundantly to express the same ideas"

   "Disruption is one of the negative cognitive side effects multimedia systems may have if information types are used incongruently"[30], and has a great impact in the user's capacity of paying attention to a presentation and understanding it completely.

   In summary, it will be easier for the user to understand a presentation with mixed media types, if the corresponding media elements refer to the same concept.

3.3 Media types
In this study we will follow the media type classification described in the Multipurpose Internet Mail Extensions (MIME) RFC [22].

The five top level discrete MIME media types are: text, image, audio, video, application. However, we will not develop the last type, as its exact meaning is given by the application that handles the format. We will base our study on the other four, that have meaning by themselves.

Text textual information. It can be plain text or enriched [22].

Image requires a graphical display.

Audio requires an audio output device.

Video requires the capability to display moving images.

In the MIME specification [22] it is accepted that a video could include (although it is not recommended) synchronized audio. Despite that, as our work in this section will be conceptual, we will only consider simple [23] media types: video without sound, images without text in them, etc...

We make a previous distinction between static and dynamic media types. Static media is the media with no inherent temporal dimension, they do not change over time: text and images. Dynamic media is the media with an inherent temporal dimension, they evolve during time: audio and video.

3.4 Psychological aspects of each media type
In this section we analyze the the more important psychological aspects that apply to each media type. We analyze factors such as cognitive resources needed, attention centering and memorability. We will also make a short analysis about what every element is suitable for in a multimedia presentation.

Text

Dimensions

For our purposes we will consider text as a 2-dimensional entity (x, y), as it has to be displayed on a screen and, in Cuypers, text pieces are treated as rectangles.[1]

It has no inherent temporal duration (Static Media).

Psychological issues

- Text is tokenized: that is, in a text there are smaller parts (tokens) that have meaning for themselves (and they are text themselves). The reader, in general, gets the complete message of the text by sequentially capturing the meaning of every one of these tokens.
- Text requires reasoning: in order to comprehend a text it is not enough to see the letters, but a reasoning process is needed. Every token is processed inside the reader's mind, and some representation of the whole text is created.
- Reading time: in consequence, text needs time to be read and understood. This time depends on the reader, the complexity of the text and its length. The complexity of a text can be found in the shape of the letters and layout of the text (visual complexity), in the structure of the text (syntactic complexity), and in the subject developed (semantic complexity). A deeper analysis of text complexity can be found in [29].
- Text is read, not only viewed. Most of the information is not in the font, or in the size, or in the colors. Text is a communication medium just like the air vibrations in a speech communication. As a medium, it has no inherent information. The information is in the meaning of the words.

[1] Although this can be seen as a broad simplification, most multimedia formats use this approach.

- Text can, however, carry information in the visual aspect. With the use of fonts and colors, some information can be carried in the physical aspect of text.
- Less memorable than other visual media: text is less appealing for the human senses than images or a video, and studies have demonstrated that people remember for a longer period what they see than what they read [4]. This is generally ascribed to the fact that while images are remembered in a dual way, visually and as concepts, text is generally remembered just as concepts, easily forgetting the exact words.
- More memorable than speech: a textual representation of speech seems to be, in general, more memorable than the speech itself. This phenomenon is generally attributed to the user's capacity of controlling the timing in text, so a reader can adjust his reading velocity, or reread parts of the text, when he needs to clarify it. With recorded speech, this cannot be done [48].
- Memorized as concepts: once the text has been read, and goes to the long-term memory, the text will be stored in our memory as the ideas explained, not as the actual text.

Cultural issues

Text is written in a language: the capacity of the reader to comprehend the message is directly related with the knowledge she has of the specific language the text is written in.

Reading order: closely related with the language is the way we write (and the way we read). Although in most western languages the order is left to right and top to bottom, in other languages of other places, the order changes. At this point, it is important to say that this factor can affect also the way a person "reads" an image, or a whole presentation.

Use in presentations

Text has been used in documents, long before multimedia existed, and through the history, it has been the principal communicative medium used as documentation.

Although it can be used to explain all kinds of things, in a multimedia application text is best used to explain abstract concepts or ideas, to present data, and to clarify other media elements.

Despite being the most polyvalent of all the types, reading from a screen is tiring for most people, so the amount of text in a multimedia application should be the minimum necessary.

Images

Dimensions

Two spatial dimensions (x, y).

No intrinsic temporal dimension (Static Media).

Psychological issues

- Images are comprehended at a glance. If the image is small enough to fit in the visual focus of the user, he will get a general idea of all the parts of the image at once.
- Viewing time. Although images can be comprehended at a glance, the viewer needs a certain amount of time (normally low) to process the information of an image, which grows with the complexity of the image.
- Images are examined. After a first general impression, the user can concentrate on specific parts of the image.
- Images are viewed, and the information itself is gained from the act of viewing. An image cannot be fully expressed in other formats such as text or speech. It is true that many images can be explained so that the user can get an idea of what is in the image, but he will never grasp

completely the whole information. In art works, the thing is more complex as in general the image is intrinsically related to the message, so the explanation is impossible (how to explain a color to a blind person?)

- Increases retention: studies show that people's visual memory is better than other types of memory. People remember better what have they seen than what they have read or heard [4][12].
- Require little cognitive processing: we are very used to view and to process what we see. Viewing implies cognitive processing mostly at a sub-conscious level.

Cultural issues

Cultural aspects are less important when understanding an image than in the case of text, but the direction of reading in our languages affects the way we view large images that do not completely fit in our vision focus, and the way we examine images.

Images, in general, are not affected by the language problem except if there is something written in the image.

Use in presentations

Images are very useful from an informative point of view as they convey a lot of information quickly. They are really handy for presenting spatial information, as this kind of information is inherently visual and results more intuitive in a visual form. However, they are unsuitable for conveying abstract information [8].

In [4] is stated the classification of the role of an image in an instructional document (classical paper) made by Levie and Lentz. They distinguish four image functions (in illustrated text):

1. *Attentional*. They can be used to attract attention. This is one of the utilities of using icons.
2. *Affective*. Enhance enjoyment and affect attitude. This is specially true with children, for whom reading text results hard.
3. *Cognitive*. Give new information and improve comprehension and retention.
4. *Compensatory*. To help poor readers to understand a text (or, in general, any other element).

These functions can also be applied to the use of images in the multimedia field.

Audio

Dimensions

For our purposes, where we consider two visual dimensions and the temporal one, audio is just one-dimensional, represented by an inherent temporal interval (Dynamic Media).

Interference

Two audio elements played at the same time will interfere between themselves at the physical level. This means that the listener could have problems in distinguishing each thread of sound. The relative intensity (volume) of the sounds will determine which audio element is more affected by the interference.

Psychological issues

With the word audio, we are only referring to the sense modality used in perceiving it (input modality). As we cannot explain the properties of text, images and videos just as visual elements, it is also impossible to deal with the attention problem only considering sound in general. We need to make distinctions between types of audio.

In this study, we will distinguish three types of audio media, which we think are the basic three types of audio we can find in multimedia applications:

- *Music*

  In general it does not require to be processed by the listener. The listener has to do almost no reasoning at all. A short interference shouldn't be very harmful in order to process the music, as there is a strong continuity component in music.

- *Speech*

  Similar to text, the listener must do certain amount of reasoning. Unlike text, in this case the listener is passive. Just as a passenger in a train, the user cannot choose the velocity of the travel, when it starts or when it ends. This lack of control, and the absence of the possibility of rereading means that a long, strong interference could cause the listener not to understand the message.

- *Sound Icons*

  Defined as a (in general) short duration sound that has no rhythm (is not music) and no words (is not speech). Normally, we can associate them with an object or idea, like an animal sound that we can associate with the animal itself. They should be considered as the union of Gaver's auditory icons [24] and Blattner's earcons [9].

  Example 3.4 *Examples of sound icons are most desktop sounds. They are short duration sounds that want to convey a message: error, question, desktop changes, etc.*

  Their main function is to capture the attention of the listener (like the visual icons).

Cultural issues

- Music. Non-vocal music can be grasped by everyone, although the culture marks, in some measure, the musical tastes.
- Speech. As speech and text are one to one related, it carries all cultural problems of text. Even worse, the "it's now or never" aspect of speech makes it more difficult for a non-native speaker to follow a speech than to understand a text passage.
- Sound Icons. Mostly culture independents in the way of interpreting them. However, the instances used, the sound icons themselves, can vary its meaning or not having meaning at all depending on the person, when the sounds refer to real things not existing in that person's environment.

Use in presentations

> "The main benefit of audio is that it provides a channel that is separate from that of the display."[39]

As stated on page 7, the dissimilarity of the stimuli helps to make the combination easier for the user.

> "Speech can be used to offer commentary or help without obscuring information on the screen."[39]

Speech can also be used as a helpful complement. Accompanying a text, it can help the user to apprehend the content using the dual modality repetition and consequent reinforcement of the message. It can also complement any other visual media, offering an auditory option for people with visual disabilities while enhancing the multimedia experience for non-disabled users.

> "Music is probably the most obvious use of sound. Whenever you need to inform the user about a certain work of music, it makes much more sense to simply play it than to show the notes or to try to describe it in words."[39]

Apart from this informative goal, background music is used many times to make the multimedia experience complete, making the application more enjoyable:

> "Audio can also be used to provide a sense of place or mood."[39]

> "Non-speech sound effects can be used as an extra dimension in the user interface to inform users about background events: for example, the arrival of new information could be signaled by the sound of a newspaper dropping on the floor. . . ".[39]

As said before, sound icons can be used to attract user's attention.

In general, audio media has also the property of being omni-directional [8], being very suitable for a one-to-many communication.

Video

Dimensions

Three dimensional media:
Two spatial dimensions (x,y).
Intrinsic temporal dimension (Dynamic Media).

Psychological issues

- Captures interest: no other medium can attract attention and hold an audience's interest as well as video does. Motion attracts visual attention in a very strong way. Reminiscences of our hunter's past, "our peripheral vision is highly tuned to detecting changes in motion, and our attention is involuntarily drawn to peripheral areas undergoing motion which distinct from its surrounds" (From [42]).
- In a video the viewer is a passenger. The viewer is a passive element that sees the images one after the other in a motion sequence.
- Increases retention: as stated before, studies show that people's visual memory is better than other types of memory.
- Requires attention: The viewer needs to concentrate on the video in order to gather all the information included in it.

Cultural issues

As a sequence of images, the same cultural questions that appear when viewing an image, can appear when watching a video.

Use in presentations

In an informative environment, the best use of video is to show things that move, and the movement is essential in itself (e.g., a ballet, a football goal).

In general, video and movement attract attention and result very appealing for the user.

	Text	Image	Video	Speech	Music	Sound Icon
Static/Dynamic	Static	Static	Dynamic	Dynamic	Dynamic	Dynamic
Input Modality	Visual	Visual	Visual	Auditory	Auditory	Auditory
Processing Code	Verbal	Image	Image	Verbal	Image	Image
Centers Attention	Not much	Some	A lot	A lot	Not much	No
Requires Processing	A lot	Not much	Not much	A lot	Not much	Barely
Req Cognitive Resources	A lot	Not much	A lot	A lot	Not much	No

Table 1: Characteristics of each media type.

3.5 Summary of characteristics of each media type
As a summary, we present a table of the characteristics of the media types that will be used subsequently in this section.

Static/Dynamic. We defined static media as the media that no evolutes on time, it has no inherent temporal dimensions, while dynamic media does.

Input Modality. Physical method in which the information is transmited: e.g. visual, auditory, haptic, etc.

Processing Code. Verbal processing code is related with linguistic information, while image coding is related with analogical, non-linguistic information. Linguistic information processing is mainly located in the left brain hemispher. Non-linguistic information is processed in both.

Centers Attention. Capacity of the media type to acquire and hold the attention of the user.

Requires Processing. Requires active processing of the information.

Requires Cognitive Resources. Requires attention or active processing from the user.

Of course, these properties are not orthogonal. For example, verbal processing code requires active processing, and the types that require active processing require cognitive resources.

3.6 Psychological aspects of media types combination
In this section we will analyze what happens when two media items are displayed at the same time during a multimedia application. We will center the study in attention aspects, estimating the amount of resources needed from the user and the feasibility of the combinations.

For this study we will combine two approaches. On the one hand, the study of which combinations are normally used and which ones are not. On the other hand, we will try to infer the difficulty of the combinations, using the information presented in sections 3.2 and 3.4.

Text and Image This is one of the most useful and most used combinations of media types. As it is a static combination, that can be used in paper format, it has been used long before any other one. Therefore, this combination has been studied many times [4], and consequently its power is well known. This combination is very common, resulting very easy for the users: they are highly trained in reading illustrated text. However, this is also its flaw in a multimedia environment: people are so used to seeing it in other environments that it is not very appealing; in a multimedia application they expect something more.

We know that:

- Text is a static medium.
- Images are static media.

And:

- As viewers can focus their attention at will, two static media items displayed at the same time do not interfere each other.

We can conclude that displaying an image and a text at the same time will not cause any problem as long as we give the user enough time to read the text and examine the image.

In general, an image will be focused on before a text [21], so in order force the focus to the text first this should be displayed before the image.

Text and Speech This combination can be found many times, but in general as synchronized elements, both using exactly the same words. As we stated on page 11, combining text with speech we introduce a redundancy that can help people with disabilities, and also improve performance. The other way round, text complements speech for international users, as normally it is easier to read a non-native language than to understand it in speech format, particularly if the speaker uses a non typical dialect, has an strange accent, or simply speaks very fast [39].

We know that:

- Text is a static medium.
- Text requires cognitive processing.
- Speech is a dynamic medium.
- Speech requires cognitive processing.
- Speech and text are similar. Although they rely on different input modalities, both use the same processing code.

And:

- It is difficult to pay attention to two sources if they both require cognitive resources.
- It is difficult to pay attention to more than one stimulus if they are similar.

So we can conclude that it will be difficult to pay attention to a text and a speech if they do not say the same.

Speech and text are processed in the same zone of the brain, making it very difficult to gather all the information. Even if the two refer to the same subject, it would be difficult for the user if the spoken words are not exactly the same that appear in the text.

Text and Video Text can complement a video in many ways.

We know that:

- Text is a static medium.
- Text requires cognitive processing.
- Video is a dynamic medium.
- Video requires attention, requires cognitive resources.

And:

- It is difficult to pay attention to two sources if they both require cognitive resources.
- Dynamic media center attention.

We can conclude that this could be a good combination if the user had time to pay attention to the video and to read the text. Normally, the user will center his attention in the video before reading the text. The power of motion is so great that if the text is presented before the video, the user will switch his attention from the text to the video even if he was reading it.

If a text and a video are presented at the same time, the amount of cognitive resources should be minimum, that is, video and text should be closely related and the text and or video should be not very complex (as in subtitling, where the amount of text is minimum, sometimes summarizing the actual spoken words).

Text and Text We know that:

- Text is a static medium
- Text requires cognitive processing.

And:

- As viewers can focus their attention at will, two static media items displayed at the same time do not interfere with each other.

We can conclude that as they are static media, you can show both at the same time without interfering with each other. The user will need sufficient time in order to read both pieces of text.

If both texts share the same style properties (font, size, colors,...) and language, the reading order used by the user will be, in general, the reading order inherent in the language. If within the texts the style properties are not uniform, this could change the reading order. As an example, if the texts have a heading each, and this heading is clearly visible, the user could read the headings in a glance, and then continue with the text element that he thinks that, according to the heading will be more interesting.

If styles are different, then the reading order chosen by the reader will be based on many factors (e.g. colors, sizes,...) [29].

Video and Image Motion attracts attention [21]. The user normally will center his attention in the video, but as the image requires almost no time to grasp, the user will be able to get most information from the image.

We know that:

- A video requires attention, requires cognitive resources.
- Videos are dynamic media.
- An image is viewed at a glance: the required viewing time depends on the person and is normally short.
- Humans are highly trained in viewing.
- An image requires almost no cognitive resources.
- Images are static media

And:

- We are capable of paying attention to various information sources if the amount of cognitive resources involved is not excessive.
- It is easier if the activities are highly practiced.

We can conclude that it will be quite easy for a normal viewer to pay attention to an image and a video at the same time, as the image requires almost no cognitive effort, and we can concentrate on the video. If the image is very complex, this will not be so easy and giving some time to the user to examine the image is advised.

Video and Speech Video is mostly accompanied with all kinds of sound: speech, music and sound icons, and people are highly used to see this combination (e.g. on TV, cinema, etc...). The sound complements the video images. Speech can be used to know what people on the video are saying and can also be used to enhance the comprehensibility of the video, situating it or explaining things that are not evident in it (voice-over).

We know that:

- A video requires attention, requires cognitive resources.
- Videos are dynamic media.
- Speech is a dynamic medium.
- Speech requires cognitive processing.

And:

- It is difficult to pay attention two sources if they both require cognitive resources.
- Video and Audio are dissimilar modalities.

We can conclude that as the input modality is dissimilar (visual vs. auditory), a normal user should be able to pay attention to both elements. But as both require a lot of cognitive resources, the user could have problems to acquire the complete information, if the subjects they are about are different. Only when the combination is highly integrated, with a high level of congruence, all the information will be grasped easily.

Video and Video This combination is virtually inexistent in informative environments. It can be sometimes found in more artistically oriented documents, where the primary goal is to get a visual impact. It can also be used for a direct, visual comparison between two videos.

We know that:

- A video requires attention, requires cognitive resources.
- Videos are dynamic media.

And:

- It is difficult to pay attention to two sources if they both require cognitive resources.
- It is difficult to pay attention to to more than one stimulus if they are similar.

We can conclude that it will be difficult to pay attention to two videos displayed at the same time.

Speech and Image Although not often used alone, the mixture of these two elements is in general similar to text and image.

We know that:

- An image requires almost no cognitive resources.
- Humans are highly trained in viewing.
- An image is viewed at a glance: the required viewing time depends on the person but is normally short.
- Speech is a dynamic medium.
- Speech requires cognitive processing.

And:

- It is easier to pay attention to two sources if one of them requires few cognitive resources.
- It is easier to pay attention to more than one stimulus if they are dissimilar.

We can conclude that in general, it should be easy to pay attention to an image and speech at the same time, as they are dissimilar, and the image is easy to grasp.

Speech and Speech Although this combination could be used in multimedia applications oriented to an artistic impression, it can rarely be found in another, more information centered, multimedia context.

We know that:

- Speech is a dynamic medium.
- Dynamic media require attention.
- Speech requires cognitive processing.

And:

- It is difficult to pay attention to two sources if they both require cognitive resources.
- It is difficult to pay attention to more than one stimulus if they are similar.

We can conclude that as two speeches rely on the same sense, and they require cognitive resources, it will be very stressful for a user to follow both information streams.

In [15] is explained how a person is able to pay attention to a specific conversation even if many are held next to him. We are very trained in this task. However, this does not mean that a person is able to, in a comfortable way, pay attention to more than one speech fragments. The concurrency of two or more speech streams will make difficult to the user to grasp the messages, and therefore it should be avoided in an informative presentation.

Image and Image Very common, and is also very natural for the users, as this combination can be found in printed form.

We have stated that:

- An image is viewed at a glance: the required viewing time depends on the persons and is normally short.
- Humans are highly trained in viewing.
- An image requires almost no cognitive resources.

And we also have stated that:

- Two static media items can be displayed at the same time without interferences (if they are not overlapping, of course), as users can focus their attention at will.

So we can conclude that two images can be shown at the same time as long as we give the viewer a certain amount of time to examine both images. To answer the question as to which image will draw more attention and therefore will examined first, many factors have to be taken into account [42] (e.g. colors, sizes, contents, context,...).

The rest The combinations of a visual items and a non-speech audio are feasible, due to the fact that they are using different sensory resources, and music and sound icons do not require many cognitive resources.

The audio-audio combinations are difficult to categorize in good/bad combinations, as they are very dependent on other things such as the relative volume of the sounds. As a rule of thumb, it seems better not to mix speech and speech (as viewed before), music and music, or two sound icons at the same time, because of their physical and perceptual similarities.

3.7 Summary of characteristics of each media combination
We summarize in Table 2 the values of correctness of showing two different media items in parallel, depending on their media types.

	Sound Icon	Music	Speech	Video	Image	Text
Text	OK	OK	SUBJECT	SUBJECT	OK	OK
Image	OK	OK	OK	OK	OK	-
Video	OK	OK	SUBJECT	WRONG	-	-
Speech	OK	OK	WRONG	-	-	-
Music	OK	WRONG	-	-	-	-
Sound Icon	WRONG	-	-	-	-	-

Table 2: *Correctness of media type combinations.* The table is symmetric as order is irrelevant.

OK means that, in general, the combination is easy for a normal user.

WRONG means that, in general, the combination is hard for a normal user.

SUBJECT means that the combination is easy if both items' subjects are closely related, and hard otherwise.

3.8 Application
In order to generate only correct presentations, that do not require too much effort by the user, we intend to make this knowledge about attention and media types, explicit in Cuypers.

We can introduce into the system the information contained in Table 2 as the rules to address the attention problem. In this case, the description associated with a presentation would be a list of media types that are used inside. However, in order to do this, first of all we have to extend the table to deal with combinations of more than two media items.

Remembering that table, we had OK values, meaning combinations that in general are optimal, WRONG values, for combinations that are wrong in general, and SUBJECT values, for combinations whose correctness is highly dependent on the congruency between the items. Of course, as we are talking about information overloading, adding items to a WRONG combination, which is already overloaded, only makes things worse, so combinations that contain WRONG sub-combinations are WRONG themselves. In consequence we should not play at the same time two or more of any of the following: video, speech, music, sound icons.

Static Media

We can show two texts at the same time, or two images, or an image and some text, but we would like to know if there is any limit to the number of static media that would be correct to show at the same time. In theory, with the attention rules we have set, the number of static media that can be shown at the same time could be infinite. The human capacity of centering the attention at will (page 6) enables us to 'read' a presentation with a large number of simultaneous static data. However,

although that is true, other mental processes are involved when understanding a presentation, and they can make unpleasant a presentation with too many static media shown at once. Despite this, in our case this is rarely a problem as we have a reduced amount of space, determined by the size of the screen, that will be a limiting factor.

Visual Media

The display of more than one video is WRONG in general, therefore setting the maximum number of videos playable at the same time to one. We saw that adding an image to a video is in general OK as the requirements of images are pretty low. The number of images playable at the same time as a video is indeterminate. In general, this depends on the cognitive resources required for both, the video and the group of images, that is the inner complexity of them, and also the amount of congruence among all these information sources. In this case too, the screen size is a limiting factor.

Audio

As the combination of two items of the same audio subtype is generally wrong, and we have only three subtypes defined, we can immediately derive that the maximum of audio elements playing at the same time should not be greater than three: one of each subtype. We have seen that each combination of two different audio sub-types is normally feasible, and we have to consider if the combination of all three of them is still correct. It seems to be this way, as we have empirical evidence in the form of movies and computer games where this combination of sounds is extensively used. In both, we can find background music mixed with dialogs (speech) at the same time that sound effects are played, and the result is still smooth. This feasibility can be explained because the only media of them that requires cognitive processing is speech and humans can discern a specific speech when hearing multiple sounds played at the same time [15]. The continuity aspect of speech, means that even strong interruptions that mask the speech completely, if they are short enough, do not become a problem in understanding the message. However, in this analysis there is a strong implicit assumption: music is not considered a primordial informative element, that would require real attention, but just an accompanying element that tries to create a more enjoyable experience. Otherwise, if the music was a first class, fundamental element of the presentation, the combination of music and other audio should be avoided.

Everything together

Considering images, music and sound icons as non-informative elements, that do not require cognitive processing, we have that we can mix them easily, having at the same time a number of images, music and a sound icon playing together. Normally, one element that requires processing can be also displayed: a speech, a video or some text. Sometimes we can also mix images, music, sound icons, and two or more elements requiring active processing, when the global amount of required processing is not too much or there is a high level of congruence between the information streams(e.g. a movie clip (video) with the synchronized voice (speech), background music, sound effects and subtitles in the same language).

Evaluation

The approach of using a table such as Table 2 has the advantage that it solves all the sound combinations easily, if we set a normal maximum of three. However this approach has several shortcomings, mostly when dealing with the visual elements:

- *It does not scale well*: we can never introduce all the information for all the possible combinations, and any new media type that we want to deal with needs to update the table.
- *It is too discrete*: the table in page 18 has a problem: not all the media items of the same media type require the same cognitive effort. For example, diagrams and icons are both images but diagrams are more complex than icons. Even if we stick to diagrams we cannot generalize; some instances are more complex than others, so they require more cognitive processing. This aspect,

that we could call continuity of attention, cannot be fully covered by a tabular approach as it does not contemplate specific instances.

- *It is too general*: related with the previous two defects, this way of dealing is too general, and strict in this generality. For example, we cannot make a music element non-interruptible, as it considers all the musical elements as "secondary priority" elements, that are not the objective of the presentation.

In order to alleviate these problems, the solution needs to be extended with quantitative information. We need a way to describe the attention requirements of a media element in a continuous way so that elements can be characterized as individuals instead of as groups. Although this can be partially automated by using simple heuristics (e.g. calculating the complexity of an element from its size), it needs human intervention or highly sophisticated artificial intelligence to be complete.

It would be good if the complexity of a presentation, or a partial construction, could be numerically characterized. This way we could set a threshold setting the maximum complexity allowed, depending on the intended audience (e.g. children vs. adults) or goal (e.g. educative vs. recreational). However, to get an algorithm to calculate this value is not an easy task. Even if we were able to calculate the attention value of a presentation, this can never be used alone to derive the value of a combination. We cannot take the attention values of two presentations and, only with that information, reason about the correctness of a simultaneous display. The specific source that generates this attention requirements needs to be considered. For example, it is important to separate the requirements between channels (i.e. aural vs. visual), because, as we have seen, high complexities in different channels are easier to process than if they occur in the same one. Although the problems described so far are large, perhaps the biggest obstacle is related to the congruency issue. As we saw, often the complexity of a combination is related with the level of congruency between them. A deep knowledge about the content of the media items and complex algorithms to do that perfectly are required. Also, in a multimedia presentation, two elements can be displayed sequentially, completely in parallel or overlapping which would add more complexity.

In any case, there are three possibilities in order to incorporate the knowledge about attention into Cuypers:

1. *Not to incorporate it.* This is not really a solution but a method to deal with the problem. It consists in assuming that with the media items we have as input and the design rules in the system, all the possible outputs are correct. The responsability is thus moved out of the system to the designers.

2. *As prerequisites of the design rules.* Before applying a design rule, we can check the elements we are relating in order to see if the combination will be valid. This is a computationally cheap approach but it has many drawbacks. It is an heuristic, as it cannot consider the exact position of the elements in the final presentation. As it is dependent on the design rules, it cannot be incorporated to the core of the system, and any design rules added will have to incorporate these checks.

3. *As global constraints.* We can add some constraints that force the presentation to be attentionally correct. For example, we could state that two videos cannot be played together, they should not overlap in time. This approach has the advantage that is more complete, as it can work constraining the concrete timing of the elements. Also, rules like this can be incorporated in the system independent of the design rules used. However, the use of constraints as such can result computationally expensive (see discussion about non-overlap constraint in [26]).

All the stated problems make the issue of getting a functional attentional algebra a very complex task.

4. Hypermedia Formatting Objects

In this section we introduce the concept of a formatting model, commenting its benefits and drawbacks. We will justify the applicability of a hypermedia formatting model while explaining how a good implementation could help an IMMPS, and Cuypers in particular, to generate better presentations by describing the layout of the parts generated. We present a design of our formatting model called Hypermedia Formatting Objects, and explain how it could help the Cuypers system to overcome some problems it has.

This section is based on the CWI Multimedia and Human-Computer Interaction group's internal discussions and documents by Jacco van Ossenbruggen, Oscar Rosell and Lynda Hardman.[2]

4.1 About Formatting Models

In general, a formatting model is the model that a style sheet [49], or other transformation technique, uses to describe the intended formatting of a presentation.

Benefits of a Formatting Model As explained in Chapter 2.2 of [49], most style sheets do two things intertwined:

1. They specify how your document is going to be presented.
2. They specify how that presentation can be achieved in the chosen target presentation format.

This mix-up can become a problem in two scenarios:

1. *Change of target format syntax.* Every change on output format syntax requires a completely different style sheet.

   Example 4.1 *You have written a style sheet, that transforms your XML[14] document to a presentation format, for example, SMIL [55]. Then you decide that you need the same presentation, with exactly the same behavior and look and feel, but in a different output format, say HTML+TIME [46].*

   In a situation like the one depicted in the example above, a new style sheet has to be created from scratch, because the style sheet language cannot separate the "essential" design decisions from the output-specific syntax details. The style sheets depend on the document and the output formats.

2. *Target format is too low level.* The chosen target presentation format can be a very low-level one or it can have such an arcane syntax (it may even be in a binary format) that defining style rules in those terms becomes unfeasible. In text, for example, it is very inconvenient to write style sheets in terms of the individual bounding boxes of the character glyphs, or even in terms of PostScript [1] or low level TEX[33] commands.

A formatting model can be used as an intermediate step (Figure 2). It describes a presentation in terms of abstract formatting constructs and, as such, the work can be divided into two, serialized steps ([49] page 21). One transformation is needed from the source document to a formatted version following the formatting model, and another one to adapt the formatted document to the final output format. However, it is important to remark that the only transformation that is document dependent is the first one, as it decides the layout of the document which is the author's decision, and it is completely independent of the output format. Since the semantics of the formatting model determine the presentation, the second transformation can be document independent as it just depends on the specific target format.

[2] The author of this work assumes all the responsibility for any wrong statement that may be found in it.

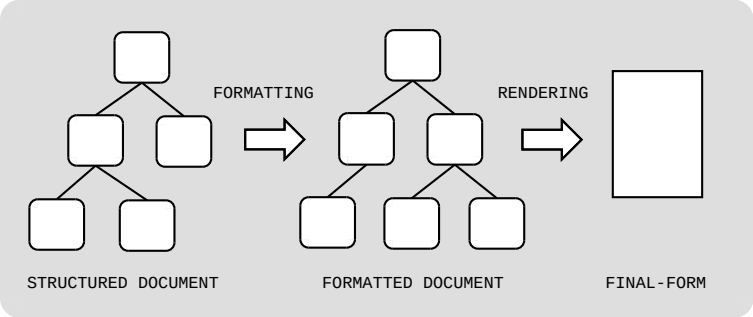


Figure 2: *Separated formatting and rendering processes.* Only the formatting step has to be guided by the author. The rendering step depends only on the syntax and semantics of the formatting model and on the desired output format.

Drawbacks The option of using an abstract formatting model, although it has clear benefits, is not unanimously accepted as a good decision because it has many drawbacks as well. It introduces an overhead since it is another abstraction to understand, another language to learn, and another level to process.

In addition, the final presentation can only contain what can be described in the specific formatting model, so new features of the target format cannot be used unless the formatting model is updated.

Moreover, the fact that the formatting model itself can have a serialization syntax, means that formatted objects can be exchanged over the Web, and these documents contain even less semantic and structural information than HTML or XML+CSS documents (and are thus, for example, a potential nightmare from an accessibility perspective).

Example 4.2 *An HTML h1 element could be formatted as an XSL–FO[54] block object with certain font properties and text-alignment properties. Exchanging such formatting objects over the Web will produce the same visual result on a standard browser, but carries insufficient information for other processing, e.g. "display" on a voice-synthesized interface, or adequate adaptation on a WAP [58] phone.*

An existing example: XSL–FO The eXtensible Style sheet Language (XSL) [54] is a complex W3C specification that comprises two technologies:

1. *XSL Transformations (XSLT)* [16] is a language to create style sheets to transform an XML document.
2. *XSL Formatting Objects (XSL–FO)* [54] is "an XML vocabulary for specifying formatting semantics" and then defines a formatting model.

In XSL–FO, formatting objects are the fundamental building blocks of the XSL formatting model. Each formatting object (FO) is *a well-known presentation abstraction* that has a set of formatting properties that define additional formatting behavior (e.g. font-family is a property of the fo:page-number object). Most formatting objects generate or return areas, where each area represents a rectangular portion of the screen/paper. Actually, each area consists of three nested rectangles, the inner one contains only the content, a second one that is potentially bigger to provide padding space, and the outer one is potentially bigger to provide border space. Finally, each FO defines the composition rules that apply to its children (e.g. "The direct children of an fo:multi-switch object are fo:multi-case objects").

All these Formatting Objects are structured forming a tree (with an fo:root element), that is known as the Formatting Object Tree (FOT), which describes a formatted document.

4.2 Formatting Models and Cuypers
We are focusing on IMMPSs and on Cuypers in particular. The benefits of using a formatting model in this context are twofold:

1. As in any other environment, using it we are achieving all the typically associated benefits that were described in Section 4.1. Specifically in Cuypers, we will be able to describe the style rules in a output independent format, and in a higher level of abstraction than SMIL.

2. In the process of generating the presentation, it can be used to meaningfully describe the inner structure of the presentation or of a partial construction, helping us to decide how to combine them in a more intelligent way.

As we explained in Section 1, one of the objectives we set in this work was to find a data structure to hold the layout of the presentations. As we said, a detailed description of the structure of the parts created can help in order to have a global view of all the parts and to implement complex algorithms.

Considered alternatives The prior implementation [26] was based on ad-hoc created formatting objects that basically formed a tree of 3D (x, y, time) bounding boxes, in which each atomic media item and each composite group was represented by its bounding box. The relative positions of the boxes was set, by asserting qualitative relations (Allen relations[2]) on three dimensions between the boxes, without considering their inner structure. The only accessible information (implicitly from the constraint set) about a box were its dimensions. Although the concept of Formatting Objects appeared in the previous version, in themselves they were only used to guide the SMIL generation, but the constraint process was only based on the bounding box model.
The boxes were created bottom-up in this way:

1. At the lowest level, the media items were a box by themselves. The dimensions of the box corresponded to the natural dimensions (i.e. not scaled) of images, videos and audio items, and they were approximated using heuristics in the case of text.

2. On higher levels, a parent (containing box) of one or more boxes was created, asserting qualitative relations (i.e. Allen relations) among the boxes. The bounding box for this group was implicitly created as the minimum box containing the children. The children of this grouping box could be media items or they could be groups by themselves.

The set of qualitative relations asserted at every stage was determined by:

1. The semantic relations holding between the boxes. These came from the rhetorical structure tree that constituted part of the input.

2. The design rules in the system. These form the mapping from the semantics to the asserted constraints and they should be set by a designer.

3. The feasibility of that set. If the set of relations could not be solved (they were inconsistent with the global constraints), another set of relations would be obtained and evaluated. For example, trying to situate two elements one next to the other, if they do not fit in the screen, the constraints fail and a temporal disposition may be chosen.

This approach was computationally effective but lead to some problems:

- *Absence of consistency*: the same semantic relation can be eventually formatted in different ways depending on the backtracking behavior, because all formatting decisions are local. This can be good as it is easier to find *one* possible presentation, but can end with very heterogeneous presentations even for the same intended meaning. Consistency is a very important aspect of multimedia design [23], as it has an impact on other highly-valued qualities such as predictability.

- *Absence of adaptation*: once the box is created, it is finished (this is not absolutely true because of backtracking). That is, the formatting inside the box does not take into account the situation of the box in the whole presentation or the relations that hold between it and the other elements of the presentation.

  Example 4.3 *An image with a label can be a box, where the only thing stated is that the label's box is next to the image's box. If we add another image at the left of that construction, perhaps it could be good to put the label to the right, in order to keep the idea of a closer relation between the first image and the label. If the label is situated at the left, then there could be some confusion about which image is the label attached to. This kind of behavior is hard to model with the previous approach.*

- *Absence of intra-box relations*: sometimes we want to map a relation between boxes in a set of relations between the contents of the boxes. Sometimes referred as the "Cousins Problem" [36], the data structure should facilitate this behavior.

It can be argued that perhaps it is not the box model that is insufficient, but the algorithms used, the bottom-up approach. However, even with free movement to traverse the tree (i.e. top-down and bottom-up) these problems are difficult to solve. The reason is that once the boxes are created, they all look the same. All the rhetorical and communicative relations between the elements have been lost in the transformation. All the reasoning that lead to that placement has been completely forgotten, and we only have the placement itself.

Example 4.4 *We have an image and an associated label. Once they are converted to bounding boxes they are an image with a close text (or a text with a nearby image). No further adaptation can be accomplished, as we do not know which modifications could change the intended meaning, which adaptations would push the limits too far. All this required information is lost.*

Other considered approaches have been to use existing multimedia models like the Amsterdam Hypermedia Model (AHM) [28] or SMIL. However, we found these to be too low level for our purposes, with the same lack of information problem. They were designed to describe an existing presentation, and although they perform well in that task, their lack of meaningful, formatting constructs makes them unsuited for solving our problems.

Example 4.5 *SMIL basically defines the presentation in terms of media items, their absolute positions, and their temporal relations. However, it does not define any formatting structures: the position of the elements is described but not the relations between them that lead to that distribution. Only the proximity could help us to deduce which elements are closely related, and which ones are not. No meaningful formatting elements are provided.*

On the other hand, XSL–FO describes the presentation in terms of meaningful formatting objects: flows, footnotes, tables, ...

A Hypermedia Formatting Model The term "Hypermedia Formatting Model" is based on the assumption that this model should provide, in the realm of hypermedia, more or less what DSSSL's [31] and XSL–FO's [54] formatting models provide in the realm of text.

Although our requirements are similar to what these models offer, they cannot be directly used. The reason is the huge difference between a multimedia document and a textual, page-oriented one. While text-formatting is essentially based on the linearity of text-flows [49], this doesn't apply to hypermedia. In text the basic elements are letters and words, and what is important when displaying them is to maintain the sequentiality. The exact position in the page is not relevant, while in multimedia the exact position and timing of every element are not only important but fundamental. We need a new formatting model adequate for the specific characteristics of hypermedia.

Although our hypermedia formatting model is likely to be significantly different from text, we can learn some lessons from the text case: the (implicit) formatting objects of HTML/CSS [11] and the (explicit) formatting objects of XSL and DSSSL are all based on well-known presentation abstractions (inline/blocks, pages, floats, etc) that have proved themselves in our long tradition of paper-based printing.

4.3 Requirements for a hypermedia formatting model

This is a list of the main requirements that a hypermedia formatting model should fulfill.

- It should support a wide range of media: audio, video, still images and text.
- It should be able to describe the spatial and temporal layout of a hypermedia presentation and parts of it. In concrete:
  - It should provide full positioning capabilities on a three dimensional space: x, y and time.
  - It should provide absolute positioning on the presentation medium. This is the basic positioning method, as it allows any presentation to be described in this way.
  - It should provide relative positioning with respect to the presentation medium and the other elements.
  - It should provide implicit positioning. The position of some elements should be implied and not explicitly stated. This is very important as it permits the creation of a valid layout without specifying any numerical value.
  - It should provide grouping and positioning of the groups.
- It should provide temporal synchronization facilities.
- It should cater for true non-linear hypermedia presentations. In concrete:
  - It should be able to specify one-way directional linking.
  - It should be able to specify internal and external linking.[3]
  - It should be able to model the AHM atemporal composite node and SMIL's excl element.
- It should provide formatting facilities such as colors, sizes, etc.
  - It should provide grouping and assigning formatting properties to them.
- It should be extensible.
- Its processing should be feasible.

Some requirements we impose on the data structure because it has to be used in Cuypers to improve the system are:

- It should be easy to mix two or more of these structures into another one, giving a relation between them. This mixing should take into consideration the inner structure and presentation behavior of all the structures involved.
- It should result in a reasonably optimal use of screen real estate at presentation time.
- It should be compatible with an automatic adaptation process.
- It should enable us to accomplish certain degree of consistency, adaptation and intra-box relations.

[3] We define internal linking as the link where both termination points are in the same document, and external if not.

4.4 Design

Our design for a Hypermedia Formatting Model is inspired by XSL–FO. Based on this, we define the notion of a *Hypermedia Formatting Object* (HFO) as *"a well-known hypermedia formatting abstraction"*. A hypermedia presentation is then described by a set of HFOs organized in a hierarchical way, forming a *Hypermedia Formatting Object Tree* (FOT) where the leaves are media items and the nodes are HFOs. We call this model *Hypermedia Formatting Objects*.

The fundamental characteristic of an object is its name, its class, as it distinguishes one object from another. Every object defines its own formatting behavior: the way it should be displayed. Every HFO will have a set of children, for which it will decide the relative spatial and temporal layout. In our implementation this will be done by asserting constraints about the spatial and temporal relations they must hold. Each HFO could impose some restrictions about the type and number of its children. For example, the children of a list could be constrained to be list elements. Associated with each HFO, we will have one or more three dimensional boxes containing the boxes defined in its descendants.

Every HFO will have associated a set of formatting attributes, that will refine the behavior of the object specifying formatting details such as fonts to be used, colors, alignments, etc. These attributes could be directly assigned to individual objects but most of the times these will be inherited from their ancestors, in a similar way in which the inheritance of attributes is done in XSL–FO [54] or CSS [11]. For example, a font color specified at the root document is susceptible of being inherited by all the objects.

With this approach, we try to solve the described Cuypers' problems this way:

- *Absence of consistency*: The presentation should be as consistent as possible. As every object class has defined its own behavior, the different instances will be consistent during all the presentation.
- *Absence of adaptation*: Small adaptations can be made to refine the behavior of the object, depending on the relations it gets involved in and its ancestors. The inheritance of attributes will also modify its final behavior. However, the essence of the formatting abstraction the object describes will be conserved.
- *Absence of intra-box relations*: Each object has a reference to its inner structure, and it has certain knowledge about the nature of the children elements, by checking their types. This way, it will be easier for them to map a relation between parents in one or more relations among the children.

If the HFOs describe the formatting, the designer can describe the layout of the presentation by defining the rules that map rhetorical and communicative constructions to HFOs and their properties. The generation of the presentation will proceed by generating the FOT with a bottom-up approach. Now, constraints will not be stated explicitly in the body of the design rules, but they will be implicit from the chosen HFOs. A new HFO layer is added to the architecture of the system, that subsumes the two constraint layers (i.e. qualitative and quantitative) we saw in Section 1.

4.5 Hypermedia Formatting Objects Design

Here we expose a series of objects that we have designed, explaining their formatting behavior and properties that apply to them. This set is not intended to be definitive, but a first collection that, in the future, will be extended and reconsidered as needed.

Root represents the viewport and it is the root of an FOT. Every possible presentation contains one or more objects of this kind, corresponding to one or more possible viewports where the presentation will be displayed.

Slideshow has N children, arranged temporally one before the other. Some of the formatting properties that apply to this object are:

- Vertical Alignment: determines the relative positions of the children on the dimension y. Possible values are: top, bottom and center.
- Horizontal Alignment: determines the relative positions of the children on the dimension x. Possible values are: left, right and center.
- Temporal Padding: time delay between children.

Figure has two children: some content and a text caption. The text is displayed below the content.

- Horizontal Alignment.
- Padding: determines the physical distance between the children.

Title has two children: some content and a text caption. The text is displayed above the content.

- Horizontal Alignment.
- Padding.

Horizontal Box has N children, arranged spatially over dimension x, one before the other. Some of the formatting properties that apply to this object are:

- Vertical Alignment.
- Padding.

Vertical Box Similar to the Horizontal Box but over the y dimension.

- Horizontal Alignment.
- Padding.

Parallel has N children, that are played at the same time. It is used with audio elements and requires at least one child to be of the Audio or Parallel classes.

Media has no children. Represents visual media like a piece of text, an image, etc.

Audio has no children. Represents audio media like music, speech, etc.

4.6 Discussion

We have shown how formatting models are used in text (and hypertext) to describe the intended formatting of documents, by specifying their layout and their formatting while abstracting from output formats and interpreters. In Section 1, we stated as a goal to find a good data structure to dynamically hold the layout of the different parts of a presentation, as they are created by Cuypers. We claim that a formatting model could be that structure, and its use could help the system to improve the quality of the presentations.

At the end of this section, we set the requirements of this model and some design guidelines, including the characteristics of a set of objects. At this point we have to admit that this design is just a tentative approximation to a final solution. For example, at this moment we are not sure about which level of adaptation can be included natively as part of the model. This and other parts of the design are open issues that will be refined after some experimentation has been done.

In the next section, we will expose the details of the implementation in Cuypers of our formatting model, Hypermedia Formatting Objects, and the results of its addition to the system. With this practical experience, we will be able to evaluate the model.

5. Implementation

This section describes the results of a partial implementation of the HFO model described in the previous one. Our goal was to make an exploratory prototype implementation, that demonstrated all the potential the formatting model offered, against a extensive implementation, where the whole model should be implemented. First we expose the inner details of the software implementation. After that, we discuss how the formatting model and the implementation can help us to accomplish consistency, adaptation and intra-box relations. At the end of the section, we make explicit some problems we encountered while implementing the model.

5.1 Implementation

In this section we describe how our experimental formatting model has been partially implemented, explaining the set of objects created.

Since we are working in an automatic presentation generator, not only the formatting objects have to be transformed to an output format, but in addition, the formatting object tree must be generated, it has to be correct from design and communicative perspectives, and adapted to the user environment. This is strongly reflected in the implementation.

Software For implementing the Formatting Objects, the ECL^{i}PS^e system was extended with the addition of Logtalk [18], a lightweight object-oriented extension for Prolog, compatible with any implementation that follows the ISO standard [32]. The reasons are that, besides seeming natural that Object-Oriented Programming (OOP) could be beneficial for the development of Formatting Objects, the use of the OOP paradigm adds many benefits to the system from a software engineering point of view. The main advantages of OOP are:

- *Encapsulation*: data and functions are encapsulated in the object. The access to the data and functions of an object is limited by the public interface that the object provides. This way, programs become more modular; changes in the inner implementation of an object do not propagate to other parts of the program using them, as long as the meaning of the interface does not change. Logtalk supports restrictions in the access to the object "a la C++"[47] with the facility of declaring public, private and protected predicates.

- *Inheritance*: The objects are hierarchically structured. The common behavior of similar objects can be grouped in higher level objects than the ones below inherit from, eliminating redundancy. This way, general solutions can be applied to general problems, reducing the amount of code necessary.

- *Polymorphism*: the final implementation of a general operation is defined at run-time depending on the type of the object that executes the call. Different objects can implement the same function in very different ways, while the interface for accessing the functions from outside remains the same. This way, particular solutions can be applied to general problems, in order to improve the efficiency or when general solutions are not applicable.

  In a way, polymorphism is also a common feature of standard Prolog, and it is part of the basis of the language.

  Example 5.1 *We can define a Prolog rule add/3.*

```
add([],[],[]).

add([X|Xs],[Y|Ys],[Z|Zs]):-
   add(X,Y,Z),
   add(Xs,Ys,Zs).
```

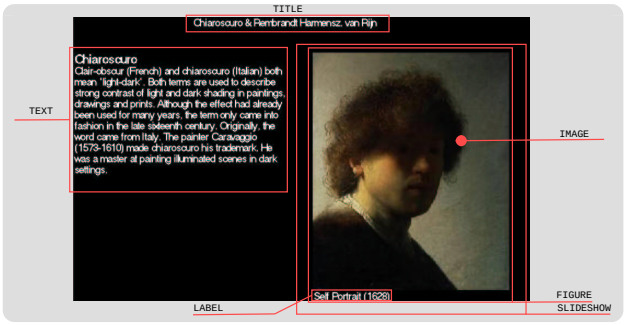


Figure 3: *Chiaroscuro presentation.* The settings are: biggest screen size, high bandwidth and low knowledge. The outlines and the black text are not included in the presentation but are just indications of the elements contained.

```
add(X,Y,Z):-
   Z is X + Y.
```

If we call add(X,Y,Z), the output will be different depending on the specific type of the parameters.

Logtalk simplifies this polymorphism for the programmer, providing a typical object-oriented language implementation. When a function is specified inside an object's definition, it sets the behavior for that object and for its descendants upon receiving that function call. However, this inherited behavior can be redefined in the descendants. Whenever the function is called upon an object, the actions will vary depending on the definition of the function for that object. What Logtalk adds to Prolog is just transparency, as we do not need to check the object type explicitly.

In the end, the visible output of using object orientation is that the applications created are more modular, easier to understand and maintain. Also, but not less important, it helps in the design stage when you are used to object-oriented analysis.

The Demo To start implementing the formatting objects we began setting as our objective to reproduce, in formatting object terms, a demo that appeared in a previous version of Cuypers: the Chiaroscuro demo [26].

The use case of the demo is this:

> "A person interested in art consults the application. He is interested in knowing the meaning of the "chiaroscuro" technique relative to the paintings of Rembrandt van Rijn"

In the application, an HTML form is used to capture all the information about the user: the user's interest and expertise level, quality of the network connection and dimensions of the screen.

Depending on the configuration, the rules applied in the system will vary, ending with different presentations, but all of them fulfilling the user interests in knowing about the term "chiaroscuro" as applied to Rembrandt's artwork.

For example, with the biggest screen size, a high-quality network and a low knowledge of art, the presentation generated is the one with the FOT that appears in Figure 4. This demo shows a title on the top, and the content below. At the left, a piece of text explaining the term chiaroscuro, that is also reproduced in a spoken commentary. At the right, we can see a slideshow[4] of a set of pictures

[4]One after another and, in this case, with a fade-in/fade-out transition effect applied.

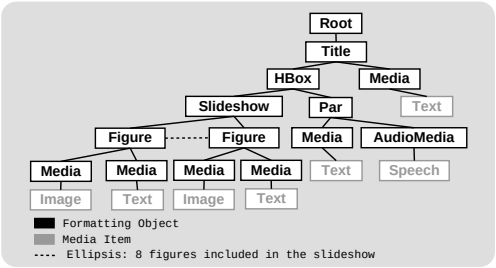


Figure 4: *Chiaroscuro presentation's Formatting Object Tree.* The settings are: biggest screen size, high bandwidth and low knowledge.

in which Rembrandt used the technique. Below each image, the name and date of creation of each picture appears.

With the smallest screen size, things change. As the system attempts to generate the same presentation as before, it gets to a point when the set of constraints becomes inconsistent, since some of the elements cannot be displayed simultaneously because of the reduced screen size. This forces the system to backtrack choosing other rules until the FOT generated is feasible. As a result, the output presentation is different. Now the explanatory text appears alone at the beginning, accompanied by the simultaneous speech and with the title above it. When the speech concludes, the whole screen changes showing the pictures in the slideshow.

The Hypermedia Formatting Objects From the demo, we extracted a series of formatting abstractions that we implemented as our first set of formatting objects. The formatting behavior of these objects was described in Section 4.5. We also created a series of abstract[5] objects to encapsulate shared behavior that other objects inherit. A simplified (only the objects name) UML [41] conceptual diagram is depicted in Figure 5.

Here we describe the abstract objects that were needed and the role of the formatting objects implemented in the presentation.

HFO An abstract object from which all others inherit.

Composite An abstract object. It comprehends all the objects that have children (they are composed by other objects).

Atomic An abstract object. All the objects that do not have children inherit from this object.

Root As stated before, every FOT includes one object of this kind.

Slideshow The paintings and their captions are shown this way, sequentially. It is also used to display the text and the slideshow, when the screen is too small to display them simultaneously.

Figure This object is used to display the pictures and their information.

Title It is used to display the title of the presentation.

Horizontal Box It is used in some versions of the presentation, to situate the slideshow and the figure.

Vertical Box It is used when the screen is too narrow to situate the explanatory text and the slideshow.

[5] An abstract object is one that cannot be instantiated.

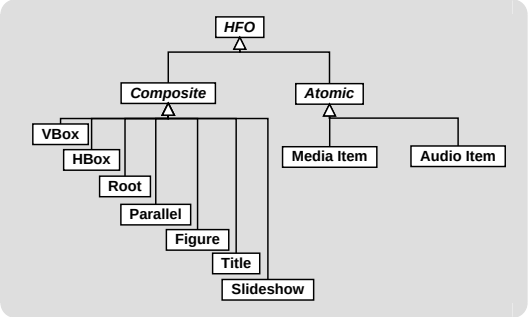


Figure 5: *Conceptual diagram.* Simplified UML conceptual diagram of hypermedia formatting objects.

Parallel In the presentation it is used to synchronize the explanatory text with the voice-over.

Media Represents the visual elements of the presentation.

Audio Represents the voice-over in the presentation.

Details All the objects are implemented as Logtalk's parametric objects (page 17 of [18]). The parameters that they require are:

1. *Id.* Every object is uniquely identified in the system. This Id is dynamically generated when an object is created. Although from the formatting model perspective it is not mandatory that all the objects have their own unique id, having an id for every object simplifies the debugging, as you can easily know which object causes the problem.
2. *Properties.* A set of formatting properties that modify the final formatting of the object.
3. *Children.* The children of the object. A list passed as a parameter on an object's creation.
4. *Coordinates.* Although in the design of the formatting model in the previous section we stated that each object could have zero or more boxes associated, in the current implementation they have always one and only one box. As in the previous version, this box represents the position and size of the object. The coordinates used to define these boxes the begin and end points over each dimension[6]. During the execution, these coordinates are variables with a finite domain associated, that is reduced as the generation evolves.

The objects are not just a repository of information but they offer an interface for making queries and modifications. The most important function that every object supplies is the init function. Upon receiving the call, the object creates a new id for itself, determine the relations between the children depending on the specific type, and generate its own coordinates as a bounding box containing all the children.

Although the parent object determines the position of its children, as relations between them, in the implementation the children are entrusted with the responsibility of asserting the constraints needed. This is because they are the owners of all the information needed to make that assertion, the coordinates, and also because they are thus aware of the relations in which they are.

The constraint labeling process [26] is a per object process now, instead of being a per variable process. This way, the information contained in the object can be used to improve the labeling algorithm.

[6]Sometimes it is more intuitive to reason with the extension over each dimension. We tried another approach using begin and extent but we had problems defining the constraints on these attributes. A solution could be to maintain redundant attributes (e.g. begin, end and extent) as done in [27].

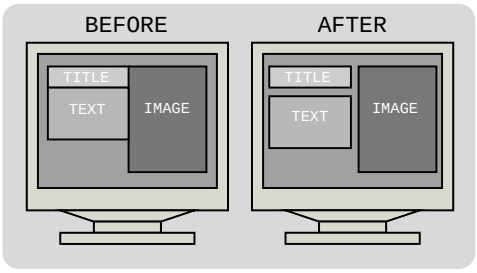


Figure 6: *Padding.* Presentation before and after the padding has been applied.

5.2 Consistency
As stated in the previous section, consistency is important for the quality of a multimedia presentation. It is defined in [23] this way:

> "Consistency measures application regularity,..."

However, to achieve this regularity is expensive. The same way that when we define a constraint we are reducing the number of solutions, if we define a construction to be consistent we are doing the same, we are setting a global constraint.

The use of a hierarchy of objects can help to obtain regularity. Procedures implemented in higher levels of the object hierarchy are shared by all the objects below.

Also from [23]:

> "Treat conceptually similar elements in a similar fashion and conceptually different elements differently."

By implementing procedures in the lower levels of the hierarchy we can accomplish this goal. Similar elements (instances of formatting objects) are formatted finally in a very similar way, as all of them execute the same code. Different objects have different behavior and, consequently, they will be differently rendered. This justifies the necessity of objects like title, figure and vertical box. Although their fundamental behavior at constraint level is the same (they display their children vertically, one before the other) the way this can be refined and their final appearance could vary.

Padding We define padding as the distance between two objects in a presentation. It is important from an aesthetic point of view, as in general results unattractive to have visual elements attached one to the other, with no separation between them. More important than that is the structural and communicative function of padding. Varying the distance between two elements we can associate or disassociate them. A typical example are text headers, where they are physically closer to the text below, thus representing the stronger association between the header and its corresponding piece of text.

We wanted the padding implementation in Cuypers to be an example of consistency across a presentation: that is, we wanted to be able to set a specific padding for all the occurrences of a hypermedia object and, as a concrete possibility, for the whole presentation. However, it is not trivial to implement padding in a system like Cuypers. If we set a global, fixed padding amount from the beginning, we can end with no presentation because the addition of padding requires physical space that we do not have, or it could take a long time because of the caused backtracking. Our position has been treating the padding as a second class element, prioritizing global consistency versus the exact amount of padding used.

Implementation The creation of the padding starts at the root element, which has a padding property that represents the default padding value, applicable through all the presentation. The root element

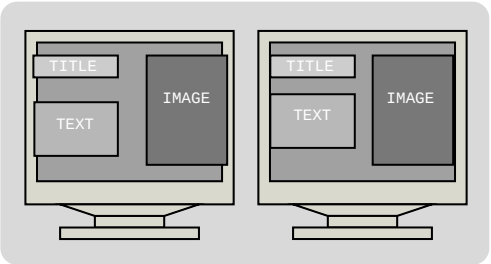


Figure 7: *Padding.* If the value of padding is excessive, then it is reduced until the presentation fits on the screen.

initializes a logical variable, constrained to have a value between 0 and the amount of padding desired. This way, we can achieve the flexibility necessary to avoid backtracking, as the value will have the closest value to the desired one maintaining the consistency. The root element calls a function on its child (setPadding), telling it to set the padding and passing the variable as a parameter.

Every element, upon receiving the order to generate the padding, checks if it has an specific padding attribute defined for itself. If it has not, then it uses the inherited value and makes all the assertions needed, depending on the object type. On the contrary, if the object has its own padding, then it discards the inherited one, and as the root element, sets its own variable that will propagate to its children.

After deciding the variable to be used it executes two instructions:

```
setPaddingForThis(Value),
setPadding(Children,Value)
```

The first one sets the padding for the current element, asserting a distance constraint between the children.

Example 5.2 *The formatting object named Title, consists of some content and a text (the title) above. When it receives the order of generating a padding, as it knows its inner structure, it asserts the next constraint:*

$Content.y1 >= Text.y2 + Padding$[7]

The second instruction, calls the setPadding function for each of the children, passing the variable as a parameter.

Example 5.3 *In the case of the Title formatting object, it calls the function setPadding upon its two children: the text and the content.*

At the end, when all the padding has been set, the actual values are assigned to the variables in the labeling stage. Of course, in all the process, the domain of the variables could have changed, being reduced when constraints are asserted. As all the objects share the same padding variable the consistency of the padding across the presentation is assured above the exact value, that can change if the suggested one is not possible to be maintained for all the presentation.

The system could be easily extended in order to accomplish consistency at the formatting object level. That is, all the objects of a specific type would be displayed with the same padding. In this case, instead of a variable, a list of N variables should be set, where N is equal to the different number

[7]This is an abbreviation of the actual ECL^{i}PS^e code.

of consistent paddings that we want. Each object, when receiving the list, should take the value applicable to itself to calculate the padding, and pass the list to its children.

It is important to see that the exact behavior of the setPadding function varies depending on the class of the object.

Example 5.4 *The formatting object named Vertical Box, consists in N elements, displayed one above the other. When it receives the order to set the padding, it states that the distance between the elements must be equal to the padding, and forwards the call to its children.*

Example 5.5 *The formatting object named Figure consists in some content with a text label below. When it receives the order to set the padding, it just forwards the call to the children. It does not set any distance between the content and the text, signifying in this way that both elements are closely related, and more related between them than with any other element on the screen.*

Comments It is important to see that the consistency that we got is only partial. The algorithm described could increase the distance between two elements, but it will never reduce it.

Consistent Slideshow Another method for obtaining consistency is requiring some elements to be of the same type. For example, we saw that in the Chiaroscuro demo there is a slideshow composed of figures. However, when the figures are created the system does not take into account that they are going to appear in a slideshow. The figures are created independently, from the rhetorical relation between the images and the labels. If the height of the screen is too small, the asserted constraints fail, backtracking occurs, and another rule and thus another layout is deployed. For example, the label, as a secondary element[8] is discarded, and only the image is conserved. If the height of all the images is not the same, some of the figures will be realized and the slideshow created will be inconsistent, mixing figures and images. As the class of the formatting object implies a specific formatting, consistency can be easily assured by requiring all the children of the slideshow to be of the same type.

Comments This method could require backtracking in order to achieve the consistency of the children.

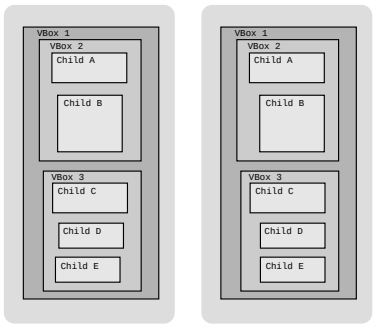
5.3 Adaptation

Adaptation consists in the modification of the formatting objects after they have been initialized, depending on the relations they are involved and the other elements of the presentation. For the formatting model, this is also accomplished mainly by the inheritance mechanism, and by defining parent-children special combinations. Using the inheritance mechanism, elements acquire the properties defined above in the hierarchy. Defining parent-children special combinations, the formatting of an object can be specified depending on its ancestors or children. For example, in our figure object class, the properties of the caption are specified in base of being inside a figure. In the system, this is more complex. As most adaptations influence the positioning of the elements they have to be notified to the objects, so that they are able to adapt the constraints. For example, a bigger font size would require more space and thus, the object should modify its bounding box to check for the consistency of the constraints.

In our global plan of generating optimal presentations, adaptation is important in order to get a certain degree of global reasoning. As the main algorithm for constructing the presentation is a bottom-up one, at the moment of creating the formatting object the only information we have is local information: information about the object itself and its children. The object could need to be refined as the generation progresses and other elements are incorporated.

In the current implementation, different ways of accomplishing adaptation are provided:

[8] We know it is secondary because the rhetorical structure trees are hierarchical.



(a) (b)

Figure 8: *Alignment*. In (a) the alignment is only between the parents, while in (b) the children are aligned too.

- *With an explicit call*. The objects supply a public set of functions to modify its behavior, that can be accessed from the outside to add property values.
- *When they are constrained*. As we stated before, when a relation is asserted between two objects, the objects themselves create the constraints. As a consequence, they are aware of the relations in which they are involved an could react modifying themselves and their children.
- *Using the inheritance mechanism*. Using that mechanism, the objects inherit attributes defined when the ancestors are created. As an example, the attributes defined in the root element as inheritable are susceptible of being shared by the whole presentation.

Padding Again, padding can also be seen as an example of adaptation. The padding to be applied is not determined when the object is created, but it is calculated after the complete structure of the presentation is determined. As seen before, this number depends on the type of the object and on all the other objects that exist.

Alignment Two objects in a presentation can be decided to be displayed aligned. When this alignment is set between two composed objects, there are two options: the alignment is only made between the parents, or the alignment is also applied at lower levels, between the children of the objects aligned. We have implemented both types of behavior.

Example 5.6 *We have the situation depicted in Figure 8. We have two vertical boxes where their children are not aligned. Then we decide to set an outer box, with the style attribute alignment equal to right, which cause the boxes to be right-aligned.*

Implementation
We have a vertical object of type Vertical Box, that asserts constraints to display its children one after the other on the y dimension. It supplies in its interface a function called align. Upon receiving the order to align, it aligns its immediate children, and also calls the function align upon the children. The children will react to that call depending on their type.

Font Color The font color for an object can be set in two ways. It can be set explicitly, as an attribute of the object when it is created, or implicitly, inheriting the value from other elements above in the hierarchy. In the later case, the value is decided after the object is created. How to deal with the case when it has an explicit value, and it also inherits another value, is specific for the object.

Implementation

The implementation of the font color attribute is simple, as it does not require to use the constraint system since all combinations are possible as they do not have an effect on the sizes. The font color can be set at creation time, as an attribute of the object, or later, assigned just before the conversion of the FOT to the final-format as an inherited attribute from its ancestors. In the last case, the object is adapted to the rest of the presentation.

Comments

Fonts could be constrained as well, in order to avoid, for example, certain color combinations that difficult text reading [17]. However, these constraints do not clash with the other set of constraints we have talked about, as they should not modify the position and size of the elements. In consequence, constraints about color would be independent of the constraints related to the positioning.

5.4 Intra-box relations

First, we will show an example where intra-box relations are useful to work with:

Example 5.7 *We have an slide show composed of four images with different sizes. The corresponding heights are: 300, 400, 400 and 300 pixels. Then we have a title for this slide show with a corresponding height of 100 pixels. If we decide to put the title above the slide show, and the height of the screen is 400 pixels.*

If we use the construction method that we have since until this moment, we will set a constraint between the title (with height 100 pixels), and the slide show (400 pixels), resulting in a construction with 500 pixels of height that does not fit inside the screen. On the contrary, if we map the relation in a relation between the title and only the first element of the slide show, we can set them to be one above the other, being played at the same time, and the relation will succeed.

With the HFOs, the reasoning can be accomplished: we have access to all the structures and we can check the types of the elements.

Implementation

We have implemented an option, to correct problems like the one in the example. First it checks if it can use the first option, to title the whole slide show, and upon backtracking it sends a call to the slide show ordering him to title himself. When it receives this call, it creates the necessary relations between the title and the first of its children.

Comments

Although the HFOs offer the necessary information to do intra-box relations, they cannot do it automatically unless they are defined to do it. For example, we could define that every time a title is associated with a slideshow, the title should be related just with the first element. What they cannot do is to define a backtracking behavior, to try one option or another depending on the size of the screen or other constraints. This complex behavior is out of the scope of a formatting model. These different formatting options should then correspond to different HOTs.

5.5 Discussion

During the implementation of our formatting model some problems were encountered that we could not solve. We state them here for future work.

Exception Handling

Due to an incompatibility between ECL^{i}PS^e and Logtalk or a misconfiguration of the local copy used, the fact is that we were unable to catch the exceptions thrown by Logtalk. This made the debugging task more difficult as, for example, the occurrence of singletons is not warned. We expect to get a solution to this in the form of new versions of the software or a proper configuration.

Object Attributes

Although Logtalk offers an interface for creating object hierarchies, it does not add any new features to the underlying Prolog implementation. As a consequence, the variables used are only logical variables and, as such, once they are set they cannot be updated. This has made the implementation of the inheritance mechanisms difficult, as ground variables cannot be overwritten. To solve this problem, it would be advisable to use other programming languages that combine Prolog features with native object-oriented extensions, or mixing ECL^{i}PS^e with functional languages such as Java or C++.

Box Coordinates

As stated before, we tried another version keeping track of the origin point and the extension on every dimension as the coordinates of the boxes. This can be useful, for example, for specifying that two elements must have the same extension, even if they are not aligned. However, we found it difficult to specify some of the qualitative relations in those terms, with the options offered by the finite domain library included with ECL^{i}PS^e . Perhaps, maintaining redundant information about the dimensions of the boxes (i.e, extension, begin and end points), this problem could be solved and we could use each of the coordinates as needed.

In the next section we will evaluate the Hypermedia Formatting Objects model, explaining the differences with hypertext that lead to its current design.

6. Evaluation and Conclusions

The task of creating a presentation is difficult for a human author. A skilled human designer has to take into account many concepts such as communication, aesthetics, complexity, usability, etc. and juggle with them. As a consequence, it cannot be expected that instructing a machine to do the same could be easy. Although much work has been done to investigate how to create a fully capable working system, the day when multimedia designers lose their jobs is still far ahead.

In our work, we looked at some of the interactions that appear in the construction of a multimedia presentation and attempted to solve them. We saw that the media types of the media items used can influence the layout of the whole presentation and situated the concept of attention in the origin of this phenomenum. The rules that follow human attention can be used to derive which combinations of media types should be avoided to be displayed at the same time, and thus can be the basis for a solution. We explained the way this knowledge can be introduced in an specific system: Cuypers.

The interdependences between the different stages of the creation of a presentation make the process inherently non linear. It is a global process, where the parts created influence the way the rest is designed and, the other way round, new parts could require to adapt or completely remake the existing ones. As suggested by Geurts [26], the backtracking feature normally associated with logic programming is naturally suitable for implementing this re-planning feature. However, this is computationally expensive and, as such, should be reduced to the minimum necessary. Another way of accomplishing consistency and global reasoning is using good data structures, that permit the system to analyze the parts created and make more intelligent choices, and also offer the possibility of refining them, to adapt their formatting to the rest. We showed how a formatting model is used in hypertext and suggested that it can be that data structure. We gave the high-level requirements that a hypermedia formatting model should fulfill, and designed and partially implemented our own model: Hypermedia Formatting Objects. The model proved to be successful to describe a hypermedia presentation inside the Cuypers system.

6.1 Attention and Media Types

In Section 2, we showed that attention plays an important role in multimedia. We demonstrated that it is something that requires to be taken in account when designing any multimedia system in general, and to create multimedia presentations “on-the-fly” in particular.

Type selector:

```
H1 { font-family: sans-serif } /* all elements of type H1 */
```

Class selector:

```
*.special { color: green }   /* all elements with class special */
```

Child selector:

```
BODY > P { line-height: 1.3 } /* P elements children of a BODY element */
```

Descendant selector:

```
H1 EM { color: blue } /* EM elements descendant of an H1 element */
```

Sibling selector:

```
H1 + H2 { margin-top: -5mm }
/* H2 elements immediate siblings of an H1 element */
```

Figure 9: *CSS Selectors.* Examples obtained from [11]

We set the axioms of our study: a set of rules of attention and psychological aspects of each media type. It is important to state that the chosen set of traits could be said to be partial or biased. In order to avoid this partiality, we have based the election in texts as closely related with the subject (multimedia) as possible, trying to avoid making inferences from other attentional studies not directly related with this domain. However, the issue of attention, as many psychological fields, is very broad, and there is not a unifying theory, shared by all the scientific community. It also has to be said that the categorization of media types used is pretty coarse-grained. Image, for example, is a very broad category that subsumes very different elements such as icons (which require little attention), pictures and diagrams (which require a lot). The usage of other more detailed taxonomies, such as the very detailed one appearing in [7], would have made the study very complex, because of the number of possible combinations and, more importantly, the practical absence of scientific studies at lower levels of abstraction.

With all these ingredients, we inferred the cognitive requirements of simultaneous displays of two media items, depending on the media types, and studied other properties and factors to be taken into account when dealing with these combinations. This inference process is affected by the relative fragility of the base (the non-unifying attention rules), and the subjectivity of the process (something difficult for one person can be easy for another one). The goal was to get general results, that could deal with most cases.

At the end of the section, we explained the possible options we have for including this knowledge into the system, and we saw that this is not easy. As in many other aspects of presentation generation, it is a process of balancing between final quality and processing time. Complete solutions will take the system too much time in order to give results, while using heuristics the quality is not assured. An implementation should try to find a point of equilibrium between quality and efficiency.

6.2 Hypermedia Formatting Objects

Sections 3 and 4 were devoted to introduce a formatting model, specific for hypermedia, called Hypermedia Formatting Objects.

First of all, our implementation, despite being partial, has proved to be valid for storing the data about the presentations generated in an accessible and modifiable form. In general, this information gives us the necessary basis to make any kind of reasoning and to construct complex algorithms on

top. Specifically, it has been shown that it can help the system to accomplish adaptation, consistency and intra-box relations, three requirements that were difficult to accomplish in previous versions. It is also useful as a bridge between the communicative devices and the final output, hiding the constraint handling and offering a more intuitive abstraction in which to express the design rules than the Allen relations.

The rich expressiveness of the objects makes the HFO model stand further from XSL–FO than we had expected, and becomes more similar to HTML+CSS [11] in this aspect. For example, we have three object classes (i.e. figure, title and vertical box), whose fundamental formatting is the same, they display the elements one above the other. The necessity of having these three different type of elements is more structural, to be able to distinguish three different design constructions, than strictly necessary to solve the formatting. In XSL–FO, there are no specific objects for titles or labels, as they can be subsumed with fo:block elements, by changing the value of their attributes. Our objects are, in this aspect, more similar to HTML elements, which are more structural and, at the same time, unlike XML, they have intrinsic presentation semantics [54].

We can establish more parallelisms between our formatting objects and HTML+CSS. The default formatting of our objects is similar to the default behavior HTML elements have when they are rendered by a browser. CSS also defines an inheritance system, profiting from the tree structure of HTML and XML. Some CSS attributes are inherited, and are susceptible to be overwritten, and others are not. This is similar to the inheritance we have as we have used these specifications as a guide. Consistency is supported in CSS by the possibility of defining style properties for groups of elements instead of individuals, by using type and class selectors (Figure 9). With our padding example, we have shown a very similar approach to define common, and at the same time valid, property values. The philosophy of our adaptation mechanisms is similar to the contextual selectors of CSS. Elements in the presentation are not alone, and their formatting should reflect that. We want to define valid style properties, depending not only on the element but also on the other elements surrounding it. CSS2 defines selectors for assigning rules to the elements depending on their ancestors, their children or their siblings.

Positioning

In our model we have defined two classes of explicit positioning: absolute and relative. Although it is not used by Cuypers, it is contemplated that we can specify the position and dimensions of every element in absolute terms. We can also specify relative placements that are calculated as shifts from the natural position of the elements. For example, we could specify time delays between the elements in the slideshow, as displacements from their original positions. However, the most important part of the positioning model, and of the formatting model itself, is the implicit positioning: the positioning of the elements that does not need to be specified.

The implicit positioning of the elements is basic for a formatting model, as it allows the author to abstract from the *how* and concentrate on *what* he wants to accomplish. As multimedia has its own requirements, the implicit positioning of our model is different from the one used by CSS or XSL. CSS defines three positioning schemas: normal flow, floats, and absolute positioning (page 102 of [11]). The normal flow, as the name suggests, is the basic method, the other two being "exceptions" to the standard rule. Normal flow corresponds to a horizontal and top to bottom position of the elements, which is the "natural" positioning in text-based documents. This flow concept assumes that the sequentiality of the elements in the discourse imply the physical position in the presentation. Given a flow of text, the position of each letter can be deduced from the position of the previous one. However, this assumption does not hold in multimedia and so the concept of flow cannot be applied. In the case of multimedia, discourse sequentiality is defined by a combination of temporal and spatial relations. The position of one element on the screen does not give a hint about the position of the next.

However, the nonexistence of this natural flow does not mean that all the positions have to be stated explicitly. The difference is that the positioning mechanisms required are more dependent

on the specific objects. The relative positioning of the elements has to change depending on the pattern we want to achieve. Instead of having a natural flow, every element defines its own method for arranging its children, every object defines its 'flow rules'. For example, in our model there are objects such as the vertical box, where the position of an element is implicitly calculated from the position of the previous one, in this case, placing them one above the other.

Because of all these factors, the implicit placement and sizing of the elements in our model are determined by their parents instead of by themselves (or in combination). In CSS/XSL the natural flow linearity is assumed, and then the elements can place themselves, deciding how to behave. In this respect, they basically decide to break it (block elements) or not (inline elements). In multimedia, no flow is assumed and the elements must be externally placed.

In text, the flows also imply that the position of an element in the document is irrelevant as long as the sequentiality is maintained. This makes the implicit positioning of the elements flexible and robust. When no more elements fit in a page or in a line, a new one is allocated and the overflow is positioned there. In hypermedia this is not generally applicable as the position and synchronism of the elements are highly relevant. Although parts of a presentation could flow temporally, this needs to be explicitly stated and should be part of the specific formatting defined for the object. Some multimedia parallel constructions can lose all their meaning when their elements are shown in sequence. This makes the implicit positioning of our objects dangerous, as it does not assure by itself that the final presentation will be correct. For example, too many elements inside a vertical box could end with some of them being placed outside of the viewport. This is a serious drawback for a formatting model but, however, there is still a niche for flexibility: the constraints imposed by the objects are not always complete. For example, in the vertical box the elements need to be placed one above the other. The distance between them is not implied by this constraint, giving the system a degree of freedom. We could define the size of the box equal to the viewport, and the padding distance between the elements could be calculated dynamically.

Hyperlinking

Although the model is called Hypermedia Formatting Objects, so far we have only seen multimedia formatting objects. The reason is that internal hyperlinking is difficult to combine with our approach to automatic generation. First, it introduces a source of indeterminism, as we cannot predict at generation time the interaction with the human. Another problem is that links inside the presentation introduce a dual order. While multimedia presentations have a clear temporal order, in a hypermedia presentation the order is double: the default temporal order and the interactions. In hypermedia the semantic relation between two elements imply a temporal proximity, and this makes the incremental construction easier. Introducing links, this discourse proximity can be represented at two independent levels: temporal and navigational. Therefore, we can have two elements closer in time that are less related in the discourse than another two that are far in time and are related via hyperlink.

However, these issues are not related with the formatting model. It is perfectly plausible to introduce hyperlinking in the model in a similar way as it appears in XSL–FO or HTML. However, general interaction methods, such as menus, buttons, etc., are more complex to model and require research on their own.

A Hypermedia Formatting Model

In XSL–FO, the classes of formatting objects "denote typographical abstractions" (page 1 of [54]). We designed our objects with the intention to denote hypermedia abstractions, and we encountered that it was difficult to define a good set of them. Our objects tended to be too general, just arranging boxes, or too specific, and it was very difficult to find a name for them. The problem we have in multimedia is that, perhaps because it is still a recent medium, there is no well-defined formatting language commonly accepted, and then no formatting model can be standardized. The question then is if there can be a defined language for specifying hypermedia applications. Our opinion is that this depends on what we consider multimedia and hypermedia. If, for example, we consider computer

games as multimedia, a formatting model is not feasible. However, we think that it could be possible to define it for a more concrete subset of hypermedia. A formatting model for hypermedia has to cover as much as possible, but it cannot cover everything. However, this is also true for text formatters, with the difference that in text what is normal is more established.

In conclusion, it is clear that a formatting model for hypermedia pose different problems than the ones that can be found in text models, and different approaches are needed in many aspects.

6.3 Final remarks

The origins of this work can be found in the architecture of Cuypers, which was introduced in Section 1.

We call the four bottom layers of this architecture the core. The difference with the upper layer is that, in the current implementation of the system, any decision taken in the core layers is not definitive: it could be later reconsidered by backtracking. The Cuypers core receives two information streams that accepts as correct: the media items that will appear in the presentation, which are the content and the rhetorical structure tree, that provides the basic structure of the presentation. As these are given, the core, during the process of deciding the formatting, can treat only three types of dependencies. The first one, how the formatting depends on the rhetorical structure, was already solved in Cuypers by mapping rhetorical relations to design patterns. Thus we concentrated our efforts in the other two: how the formatting depends on the content and the dependencies between the formatting of the different parts of the presentation. We have partially accomplished this goal.

We analyzed how the content influences the layout, but only from the perspective of media types and attention. Although this is an important part, many other properties of the content can affect the formatting. For example, the colors of a set of images can influence how to arrange them, or could demand a modification on other parts such as the background color.

We worked on how the local formatting depends in the global formatting and proposed a data structure that facilitates future work. We designed a method to describe a presentation construction, but we did not study in depth how this information should be taken into account and which algorithms should be implemented to improve the output.

Moreover, the fact that we cannot treat them within the current implementation of Cuypers, does not mean that other types of interdependencies, such as the ones highlighted in [52] and [10], are not important or already solved. For example, studies should be carried out on how the specific style used could influence the selection of the media items, and how the system should be modified accordingly.

All these issues remain as future work.

References

1. Adobe Systems Incorporated. PostScript. See http://www.adobe.com/products/postscript/main.html.
2. James F. Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–844, November 1983.
3. E. Andre and T. Rist. The Design of Illustrated Documents as a Planning Task. In *Intelligent Multimedia Interfaces* [20], pages 94–116.
4. Gary J. Anglin, Robert L. Towers, and W. Howard Levie. Visual Message Design And Learning: The Role Of Static And Dynamic Illustrations. In *Handbook of Research for Educational Communications and Technology* [19], pages 755–794.
5. Mike Bearne, Sara Jones, and John Sapsford-Francis. Towards Usability Guidelines for Multimedia Systems. In *Proceedings of the second ACM international conference on Multimedia*, pages 105–110, 1994.
6. Roy R. Behrens. *Design in the visual arts*. Prentice Hall, 1983.
7. Niels Ole Bernsen. Defining a Taxonomy of Output Modalities from an HCI Perspective. *Computer Standards and Interfaces*, 18:537–553, 1997.
8. Niels Ole Bernsen. Multimodality in Language and Speech Systems – From Theory to design Support Tool. In B. Granström, editor, *Multimodality in Language and Speech Systems*. Kluwer Academic Publishers, 2001.
9. M.M. Blattner, D.A. Sumikawa, and R.M. Greenberg. Earcons and Icons: Their Structure and Common Design Principles. In *Human-Computer Interaction 4(1)*, pages 11–44, 1989.
10. M. Bordegoni, G. Faconti, M.T. Maybury, T. Rist, S. Ruggieri, P. Trahanias, and M. Wilson. A Standard Reference Model for Intelligent Multimedia Presentation Systems. *Computer Standards & Interfaces*, 18(6-7):477–496, December 1997.
11. Bert Bos, Håkon Wium Lie, Chris Lilley, and Ian Jacobs. Cascading Style Sheets, level 2 CSS2 Specification. W3C Recommendations are available at http://www.w3.org/TR, May 12, 1998.
12. Roberts A. Braden. Visual Literacy. In *Handbook of Research for Educational Communications and Technology* [19], pages 491–520.
13. Ivan Bratko. *Prolog Programming for Artificial Intelligence*. International computer science series.

Addison-Wesley, 2nd edition, 1990.

14. Tim Bray, Jean Paoli, and C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0 Specification, February 10, 1998. W3C Recommendations are available at http://www.w3.org/TR.
15. E. C. Cherry. Some experiments on the recognition of speech with one and two ears. In *Journal of the Acoustical Society of America*, pages 975–979, 1953.
16. James Clark. XSL Transformations (XSLT) Version 1.0. W3C Recommendations are available at http://www.w3.org/TR/, 16 November 1999.
17. Karolina Czajka. Design of interactive and adaptive interfaces to exploit large media-based knowledge spaces in the domain of museums for the fine arts. Master's thesis, University of Applied Science Darmstadt, 2002.
18. Paulo Jorge Lopes de Moura. Logtalk 2.6 Documentation. Technical report, Departament of Mathematics and Informatics. University of Beira Interior., July 2000.
19. David H. Jonassen (Editor). *Handbook of Research for Educational Communications and Technology*. The Macmillan Press, 1996.
20. Mark T. Maybury (Editor). *Intelligent Multimedia Interfaces*. AAAI Press, 1993.
21. Pete Faraday and Alistair Sutcliffe. Multimedia: Design for the 'Moment'. In *Proceedings of ACM Multimedia*, pages 183–192, Seattle, Washington, November 1997. ACM Press.
22. N. Freed and N. Borenstein. Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. RFC 2046, November 1996. Obsoletes: 1521, 1522, 1590 Category: Standards Track.
23. Franca Garzotto, Luca Mainetti, and Paolo Paolini. Hypermedia Design, Analysis, and Evaluation Issues. *Communications of the ACM*, 38(8):74–86, August 1995.
24. W.W. Gaver. *Everyday Listening and Auditory Icons*. PhD thesis, University of California, 1988.
25. Joost Geurts. Constraints for Multimedia Presentation Generation. Master's thesis, University of Amsterdam, 2002.
26. Joost Geurts, Jacco van Ossenbruggen, and Lynda Hardman. Application-Specific Constraints for Multimedia Presentation Geneneration. Technical Report INS-R0107, CWI, May 31, 2001.
27. Susan L. Graham, Michael A. Harrison, and Ethan V. Munson. The Proteus Presentation System. In *Proceedings of the Fifth ACM SIGSOFT Symposium on Software Development Environments*, pages 130–138, Tyson's Corner, VA, December 9-11, 1992.
28. L. Hardman, D. C. A. Bulterman, and G.van Rossum. The Amsterdam Hypermedia Model: Adding Time and Context to the Dexter Model. *Communications of the ACM*, 37(2):50–62, February 1994.
29. James Hartley. Text Design. In *Handbook of Research for Educational Communications and Technology* [19], pages 795–820.
30. Martijn Hoogeveen. Towards a New Multimedia Paradigm: is Multimedia Assisted Instruction Really Effective? In *ED-MEDIA 95 Proceedings*, pages 348–353, 1995.
31. International Organization for Standardization/International Electrotechnical Commission. Information technology — Processing languages — Document Style Semantics and Specification Language (DSSSL), 1996. International Standard ISO/IEC 10179:1996.
32. International Organization for Standardization/International Electrotechnical Commission – JTC 1/SC 22/WG 17 . ISO/IEC 13211-1:1995 Prolog – Part1:General Core.
33. Donald E. Knuth. *TeX: The Program*, volume B of *Computers and Typesetting*. Addison-Wesley Publishing Company, 1986.
34. A. Kobsa, J. Koenemann, and W. Pohl. Personalized Hypermedia Presentation Techniques for

Improving Online Customer Relationships. In *The Knowledge Engineering Review*, volume 16, pages 111–155, 2001.

35. J. Krause. A Multilayered Empirical Approach to Multimodality: Towards Mixed Solutions of Natural Language and Graphical Interfaces. In *Intelligent Multimedia Interfaces* [20], pages 328–352.
36. Simon Lok and Steven Feiner. A Survey of Automated Layout Techniques for Information Presentations. In *Proceedings of SmartGraphics*, 2001.
37. William C. Mann, Christian M. I. M. Matthiesen, and Sandara A. Thompson. Rhetorical Structure Theory and Text Analysis. Technical Report ISI/RR-89-242, Information Sciences Institute, University of Southern California, November 1989.
38. David M. Moore, John K. Burton, and Robert J. Myers. Multiple-Channel Communications: The Theoretical and Research Foundations of Multimedia. In *Handbook of Research for Educational Communications and Technology* [19], pages 851–875.
39. Jakob Nielsen. Guidelines for Multimedia on the Web. Jakob Nielsen's Alertbox, December 1995.
40. Mikael Nilsson, Johan Hjelm, and Hidetaka Ohto. Composite Capabilities/Preference Profiles: Requirements and Architecture. Work in progress. W3C Working Drafts are available at http://www.w3.org/TR, 21 July 2000.
41. OMG (Object Management Group) . Unified Modeling Language (UML), version 1.4, 2001.
42. W. Osberger and A.J.Maeder. Automatic identification of perceptually important regions in an image using a model of the human visual system. In *International Conference on Pattern Recognition*, Brisbane, Australia, August 1998.
43. Steven F. Roth and William E. Hefley. Intelligent Multimedia Presentation Systems: Research and Principles. In *Intelligent Multimedia Interfaces* [20], pages 13–58.
44. Lloyd Rutledge, Brian Bailey, Jacco van Ossenbruggen, Lynda Hardman, and Joost Geurts. Generating Presentation Constraints from Rhetorical Structure. In *Proceedings of the 11th ACM conference on Hypertext and Hypermedia*, pages 19–28, San Antonio, Texas, USA, May 30 – June 3, 2000. ACM.
45. Lloyd Rutledge, Jim Davis, Jacco van Ossenbruggen, and Lynda Hardman. Inter-dimensional Hypermedia Communicative Devices for Rhetorical Structure. In *Proceedings of the International Conference on Multimedia Modeling 2000 (MMM00)*, pages 89–105, Nagano, Japan, November 13-15, 2000.
46. Patrick Schmitz, Jin Yu, and Peter Santangeli. Timed Interactive Multimedia Extensions for HTML (HTML+TIME): Extending SMIL into the Web Browser. W3C Note are available at http://www.w3.org/TR, September 1998.
47. Bjarne Stroustrup. *The C++ Programming Language*. Addison-Wesley, 2 edition, 1991.
48. Steven D. Tripp and Warren B. Roby. Auditory Presentations and Language Laboratories. In *Handbook of Research for Educational Communications and Technology* [19], pages 821–850.
49. Jacco van Ossenbruggen. *Processing Structured Hypermedia — A Matter of Style*. PhD thesis, Vrije Universiteit, Amsterdam, The Netherlands, April 10, 2001. Also available on http://www.cwi.nl/∼jrvosse/thesis/.
50. Jacco van Ossenbruggen, Frank Cornelissen, Joost Geurts, Lloyd Rutledge, and Lynda Hardman. Cuypers: a semi-automatic hypermedia generation system. Technical Report INS-R0025, CWI, December 2000.
51. Jacco van Ossenbruggen, Joost Geurts, Frank Cornelissen, Lloyd Rutledge, and Lynda Hardman. Towards Second and Third Generation Web-Based Multimedia. In *The Tenth International World Wide Web Conference*, pages 479–488, Hong Kong, May 1-5, 2001. IW3C2.

52. Jacco van Ossenbruggen and Lynda Hardman. Smart Style on the Semantic Web. In *Semantic Web Workshop, WWW2002*, May 2002.
53. W3C. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendations are available at http://www.w3.org/TR, February, 22, 1999. Editied by Ora Lassila and Ralph R. Swick.
54. W3C. Extensible Stylesheet Language (XSL) Version 1.0. W3C Recommendations are available at http://www.w3.org/TR/, 15 October 2001, 2001.
55. W3C. Synchronized Multimedia Integration Language (SMIL 2.0) Specification. W3C Recommendations are available at http://www.w3.org/TR/, August 7, 2001. Edited by Aaron Cohen.
56. Mark Wallace, Stefano Novello, and Joachim Schimpf. ECLiPSe: A Platform for Constraint Logic Programming, 1997.
57. C. Wickens. Processing Resources in Attention. In R. Parasuraman and D.R. Davies, editors, *Varieties of Attention*, New York, 1984.
58. Wireless Application Group. WAP-174: WAG UAPROF User Agent Profile Specification, 1999.