Centrum voor Wiskunde en Informatica

REPORTRAPPORT

INS

Information Systems

INformation Systems

INS

Transforming Abstract Document Structure to a Hypermedia Presentation

R.C.E. van Velthoven

REPORT INS-E0407 JULY 2004

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

Transforming Abstract Document Structure to a Hypermedia Presentation

ABSTRACT
Automatically generating presentations from content available on the World Wide Web, mainly through multimedia databases, is made possible with the advent of the Semantic Web and the increasing availability of multimedia repositories. The Semantic Web enables machines to work with assumptions on the meaning and interrelations of multimedia content. This makes it possible to present information in a more coherent manner, paving the way for the field of automatic multimedia presentation generation. Among the different issues to be solved in this field is layout generation. When combining separate parts together into a single presentation, new options and dangers present themselves. In this report we look at two options that present themselves and the related dangers. These options are analysis of the presentation tree in order to find interesting structures and adding to the system the option to use emphasis elements when presenting media elements.

1998 ACM Computing Classification System: I.7.2, H.5.1
Keywords and Phrases: Presentation generation; structure analysis
Note: This is a Master's thesis at the Technical University of Eindhoven

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computing Science

CENTRUM VOOR WISKUNDE EN INFORMATICA

MASTER'S THESIS
Transforming Abstract Document Structure
to a Hypermedia Presentation

by
R.C.E van Velthoven

Supervisor: Prof. dr. L. Hardman

Eindhoven/Amsterdam, May 2004

Acknowledgement

This thesis would never have reached its present form, without the contribution of many people. I would like to take this opportunity to thank some of them.
First of all Lynda Hardman who gave me the opportunity to work in a world class research environment at the Centrum voor Wiskunde en Informatica. I can only hope the final result doesn't disappoint you.
Jacco van Ossenbruggen for his valuable input in the more technical matters of my work and for setting my mind at ease whenever Lynda managed to unnerve or confuse me.
Also my other colleagues as CWI, Joost, Lloyd, Martin, Stefano, Steven, Katya, Frank, Jana and Yulia. Also those students that left before me, Amit and Michiel.
Special thanks to my roommate Matthijs, with whom I could not but agree that Johnny Depp deserved an Oscar for his role in *Pirates of the Carribean.*
The Rijksmuseum for supplying the images and CWI for funding the research.
My parents for always supporting me through my studies and had the endurance to keep it up until the very end. I hope to receive similar support from them with whatever goals I choose to pursue in my future life.
Finally my sister, Mirjam, for wonderful dinners and being an excellent example. And of course all of my friends for presenting opportunities to escape everyday work. To those who know who they are, I would just like to say: *I'm not the last!*

Rob van Velthoven, Amsterdam/Eindhoven 2004

Abstract

Automatically generating presentations from content available on the World Wide Web, mainly through multimedia databases, is made possible with the advent of the Semantic Web and the increasing availability of multimedia repositories. The Semantic Web enables machines to work with assumptions on the meaning and interrelations of multimedia content. This makes it possible to present information in a more coherent manner, paving the way for the field of automatic multimedia presentation generation.

Among the different issues to be solved in this field is layout generation. When combining separate parts together into a single presentation, new options and dangers present themselves. In this report we look at two options that present themselves and the related dangers. These options are analysis of the presentation tree in order to find interesting structures and adding to the system the option to use emphasis elements when presenting media elements.

Contents

List of Figures

Chapter 1

Introduction

The need for user adapted information presentation is desirable from both a technical and a personal point of view. The technical point of view shows that there is a, still growing, number of different devices that connect to the World Wide Web. Each of these devices comes with its own technical capabilities and limitations. The personal point of view makes it clear that each user is an individual and as such has individual interests and a unique background knowledge. This too acknowledges the need for user adapted information presentation.

One approach to user adapted information presentation is to present the user with a multimedia presentation built up of tailored information items. This approach is made possible with the advent of the Semantic Web [BLHL01]. When media elements are annotated and relations between them typed, then a machine process can reason about these interrelationships and make decisions on how to present them as a coherent whole. Thus, the Semantic Web technologies enable user adapted information presentation through automatic multimedia presentation generation.

1.1 Problem statement

Even when it is technically possible to extract the best-matching data items from multimedia repositories scattered around the Web and structure them together in an abstract presentation format, this is still a big step from actual final form presentations. As advocated in earlier work [Geu02] generating the final form presentation requires a complex transformation process, that can be performed not by simple document transformation but requires complex constraint solving in a multi-layered iterative transformation process.

1.2 Scope and contribution

In previous work [Geu02] it has been argued that when generating multimedia presentations using constraint logic programming, both quantitative and qualitative constraints are needed. In his work Geurts mainly presents an extensive survey of the quantitative constraints, whereas the qualitative ones are only briefly touched upon. In particular where decisions on the actual filling in of those qualitative constraints need to be made.

This work attempts to fill this gap in the use of constraints by defining rules that apply high-level reasoning to the creation of the final layout of the presentation and translate these decisions into qualitative constraints. Here layout includes link-based interactivity as well as temporal and spatial layout. It will be argued that in order to generate qualitative presentations an extensive rule-base is required from which the qualitative constraints can be derived and a first step towards such an extensive rule base is presented.

1.3 Outline

The remainder of this thesis is structured as follows:

Chapter 2 gives an in depth discussion of the problem described in section 1.1, along with an example scenario.

Chapter 3 proposes a solution to the problem described in chapter 2.

Chapter 4 introduces a number of technologies, such as Structured Progression and Hypermedia Formatting Objects, used to implement the proposed solution, as described in chapter 5.

Chapter 5 presents the implementation of the solution.

Chapter 6 discusses the implementation and gives an overview of future work.

Chapter 2

Problem and Example Scenario

This chapter gives an in depth description of the problem, of defining high-level reasoning rules to fill in the layout of the final presentation. It does so by presenting an example scenario as it passes through a realistic transformation process and identifying the places where improvements can be made. First, a short introduction is given into both the transformation process and the example scenario. Then the scenario is described as it passes through the transformation process.
This chapter also introduces a number of concepts that are used throughout the remainder of this report.

2.1 Introduction

In order to generate a presentation we need the data to be presented and a transformation process, which takes a set of related media elements and transforms them into a coherent presentation. One possible implementation of such a process is introduced in section 2.3. We describe the given scenario as it passes through the presented transformation process in the final section of this chapter, to indicate where problems arise and how our approach will help in solving these problems.

2.2 An Example Scenario: The Biography

The scenario described in this section is concerned with presenting a biography, more specifically a biography of a painter. Biographies of painters have previously been used in related work by Bateman et al., [BKKR01]. They describe a system (DArt$_{bio}$) that generates artist biographies based on a domain model containing information about several thousand artists. The

presentation is created following presentation plans, which are a type of template expressed in terms of Rhetorical Structure Theory [MMT89]. The one we base our scenario on is taken from Geurts et. al, [GBvOH03].
This scenario is a *Biography of Rembrandt van Rijn*. Rembrandt van Rijn is one of the most famous Dutch painters and lived in the 17th century. His most well-known work is no doubt the *Nachtwacht*.
The example presentation contains a description of his personal life, his career and his private life.

2.3 The Transformation Process

There are different ways in which one can approach the problem of transforming a set of media elements into a coherent presentation. Independent of the approach, however, each has to deal with a number of decisions to arrive at the final form presentation:

- Which media elements are to be included in the presentation?
- How should the media elements be grouped together into larger parts of the presentation?
- How should the media elements be positioned relative to each other, both in space and time?
- How should the (relative) importance of the media elements within the chosen positioning be expressed?

A typical transformation process might be divided into a number of transformation steps, each of which deals with one particular decision.

2.4 The Scenario passing through the Transformation Process

Now that we have described an example scenario and the transformation process we will use to generate a presentation, we can focus our attention on the actual transformations taking place as the scenario passes through the transformation process. A sketch of this processing of the scenario is presented in figure 2.1, taken from [GBvOH03].

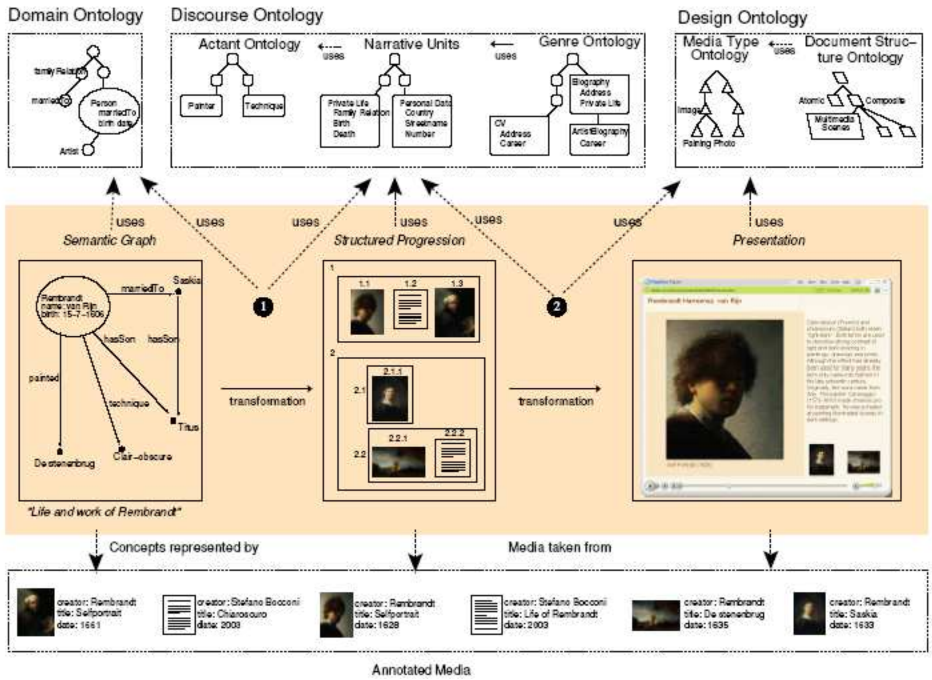

Figure 2.1: Sketch of the processing of the example scenario.
The middle layer shows an example semantic graph (left) that is first transformed to a biography structure (middle) and then to a multimedia presentation (right). The upper layer shows the ontologies involved, while the lower layer represents the annotated media items used.

2.4.1 From Semantics to an Abstract Presentation

First the system passes the user query, in this case *Biography* and *Rembrandt van Rijn* to its multimedia information retrieval back-end, which in turn queries its annotated multimedia database system for matching media elements. In our case this is a repository containing images of samples from the art collection of the Rijksmuseum Amsterdam [Rij]. This query results in a set of approximately 15 media elements. For a description of the Resource Description Framework (RDF, [W3C99]) semantic graph that is the primary input of the transformation process, we refer to [GBvOH03].

Media Element: Media Elements are the atomic building blocks of the presentation. There are four basic types of media elements: audio, image, video and text.

On the semantic level a presentation is constructed around the concepts *Biography* and *Rembrandt van Rijn* based on the available media elements and their annotations. This presentation is expressed as a ***Structured Progression***. The Structured Progression is seen solely as a grouping and ordering of media elements. The ordering of elements within a grouping element is expressed through the use of ***importance*** values, called priority in Structured Progression terminology.

Structured Progression: the structured progression is a tree representing an abstract presentation schema. It defines the presentation in terms of abstract scenes, not dictating where or when media elements should be displayed, but only grouping them into larger building blocks. The term and data structure follow Rutledge et al. [RAB+03].

Importance: the importance values assigned to the scenes of a presentation or another scene represent an ordering that needs to be maintained during further transformations. It is derived from the semantics relating to the media elements and higher levels in the transformation process are unable to reason about these, for them hidden, semantics behind the grouping and ordering. Equality, or absence of importance values, is interpreted as the scenes being elements within an unordered grouping. This means that any possibly present ordering, for example numerical or alphabetical, need not be maintained at all costs.

Based on the above criteria the structural analysis, as described in chapter 3, is performed. Based on these criteria, this analysis can be considered to be domain-independent, as it makes no use of semantics.

In our scenario this results in three major scenes: *Personal Life*, *Career* and *Private Life*. The *Career* scene is subdivided into a *Peter Lastman* and a *Chiaroscuro* scene. The latter is subdivided into another four scenes: *Rembrandt's and Hendrickje's daughter*, *Titus van Rijn*, *Rembrandt's Mother* and *Saskia Uylenburgh*.
These scenes are ***groupings*** of media elements into larger presentation building blocks, called ***composite elements***.

Composite Element: Composite elements are the groupings of multiple media elements, which we also call scenes. Composite elements have no inherent properties, such as size, duration and inherent emphasis, but derive these from their constituent media elements.

Grouping: Grouping is used to indicate that a set of media elements are related by some concept. Composite elements are formed from media elements based on groupings. All the elements within a composite element share at least one grouping, but not all elements from a single grouping need to appear in the same composite element.

2.4.2 From Abstract Presentation to Presentation Format

The Structured Progression needs to be transformed further into a document format that takes the properties of the media elements into account. These properties include the inherent media properties, such as duration and size, but also properties relevant to the structuring process, such as importance.
Based on the properties of media elements within the Structured Progression tree decisions can be made on how to present these elements. An example decision is to use ***emphasis*** to indicate the importance of an element in the presentation.

Emphasis: Emphasis is the degree to which an element draws attention relative to other elements. This applies both to media and composite elements.

These decisions are available to the transformation system in the form of rules. An example of such a rule might be that placing a label below an image is more desirable than placing it in another position relative to the image, because the label is less important than the image itself.

Currently these rules are mostly based on the properties of a single element viewed in separation from all the other elements. This makes it impossible to make decisions involving multiple elements or even entire ***subtrees*** of elements.

Subtree: A subtree is defined as a non-root, non-leaf node of a tree along with all of its descendants. Subtrees are said to be on the same level if their base nodes are siblings, i.e. if they have the same grandparent.

Distinct Subtree: It is assumed that sibling subtrees generally exhibit a certain degree of similarity. If a certain subtree presents no such similarity, it is called a Distinct Subtree.

Examples of such decisions are the following:

- Decisions to apply emphasis to an element, to express its difference from the other elements, can be based on its importance. Emphasis is something which is a relative measure between two elements, one can be more emphasized than the other.

- Decisions to structure the input. In larger presentations it becomes desirable to structure the entire presentation in sections rather than presenting everything as one long sequence. To make a decision about this the process needs to be aware of the entire presentation tree.

In the next chapter we will address this issue and present rules that enable the transformation process to look beyond single elements and make decisions involving multiple elements within the Structured Progression.

The resulting tree is a transformation of the abstract scenes in the Structured Progression to presentation elements expressed independently of an output format, which we call the Presentation Format.

2.4.3 From Presentation Format to Document Format

In the final step of the transformation process, an XSL Transformation, [Cla99] is applied to convert the Presentation Format into the final document format to be presented to the user. This last step converts the output independent format into a document format specifically suited for a particular user using a certain presentation device.

If the system were allowed to generate presentations of ten times the current size, consisting of 100+ media elements, we would be able to see that the generated presentation is a long continuous presentation, with no options to quickly traverse it to the most interesting information, from the user's point of view. This leads us to the final issue we will address in this report, the lack of structuring mechanism and navigation options through the presentations. The most common structuring mechanism is the menu and the most common navigation option in hypermedia is the hyperlink.

Chapter 3

Theory: Structural Analysis

In order to apply high-level reasoning to the choices made about the layout and structure of the final presentation, we need to be able to apply this reasoning to the Structured Progression introduced in the previous chapter. This chapter argues that this can be achieved through structural analysis of the Structured Progression. This structural analysis entails transforming the high-level reasoning concepts into rules that can be applied to the structure of the Structured Progression.

Chapter 2 presented a number of concepts we will now transform into structural analysis rules. The concepts we deal with are mutually dependent. The three concepts to deal with are Emphasis, Distinct Subtrees and Structuring Mechanisms.

Emphasis: The concept of emphasis was introduced in chapter 2. In this chapter we will detail the concept and present a simple rule set that enables us to apply emphasis to elements within the presentation.

Distinct Subtrees: Distinct Subtrees were also introduced in chapter 2. In this chapter we will describe how distinct subtrees can be identified in the Structured Progression and how we deal with them making use of Emphasis rules.

Structuring Mechanisms: The final concept deals with structuring mechanisms. It is best applied to larger Structured Progressions, enabling the creation of presentations consisting of as many as 200 elements, a factor 10 increase with current presentations. We will describe rules to decide whether or not to apply these mechanisms to the Structured Progression and how to apply these mechanisms when they are used.

3.1 Emphasis

3.1.1 Definition of Emphasis

Emphasis can be divided into two different parts. The first is the inherent emphasis each media element possesses, simply due to its nature, which we express as an absolute value.The second is the perceived emphasis, which is a relative value. This perceived emphasis concerns the way the element is actually presented in the presentation and covers aspects such as position, size, background colors, etc. We use background colors as a means to influence the way distinct subtrees are presented.
The use of colors should be applied with considerable care, as is discussed in detail in [Man03]. Such issues would also apply to our use of colors in varying the background color of elements and creating borders. For our current use, however, we abstain from these issues and only use a simple set of colors to vary background and create borders. They are used more as a shading option, as if the colors were viewed in shades of grey.

3.1.2 Using Emphasis in Presentations

When perceived emphasis is used to influence the presentation of various elements, this cannot be done without keeping inherent emphasis in mind. If two elements have similar perceived emphasis, then it is likely that the one with the greater inherent emphasis will be viewed as the most important by the user. To circumvent these situations we will define the actual emphasis level as a combination of both the inherent and perceived emphasis levels.

Another problem is dealing with composite elements, groups of elements that contain multiple atomic elements of possibly different inherent emphasis. To deal with this, composite elements are given an inherent emphasis which is the average of the inherent emphasis of its constituent elements. The perceived emphasis of the composite, which is a relative value, is determined as a whole. In effect, the perceived emphasis is determined relative to the other composites of the same level.

Taking the above assumptions into account we can then derive a number of rules that will guide the system in the transformation process.

We will start by defining an ordering of the inherent emphasis of elements based on their element type. In this ordering video elements are assigned greater inherent emphasis than image elements. A more extensive discussion of the aspects of media types can be found in [Mar02]. Image elements in turn are assigned greater inherent emphasis than text elements. Finally, as mentioned previously, composite elements receive an inherent emphasis that averages the inherent emphasis of its constituent elements.

We have identified four perceived emphasis aspects, more specifically position, size, background color and borders. We do not claim this is a complete list, but for our purpose these four suffice. Now we define the added perceived emphasis for each of the perceived emphasis elements:

- **Position:** Position can be determined in the temporal dimension and two spatial dimensions, horizontal and vertical. If two elements are presented simultaneously, they receive no difference in perceived emphasis in the temporal dimension, but it then makes sense to look at the two spatial dimensions. If two elements aren't presented simultaneously the one coming first receives higher perceived emphasis.
  The two spatial dimensions are treated equally. An element left of or above another element receives higher perceived emphasis. An element both above and left of another element receives both increases in perceived emphasis.
  These rules are based on the common reading direction in the western world, from top to bottom and from left to right. In locales with other reading directions, these rules should be adjusted appropriately.

- **Size:** Size can be used both in a temporal and spatial dimension. The spatial dimension is straightforward: any element taking up more space in terms of screen size as another element receives higher perceived emphasis.
  The temporal situation is slightly more complex. The complication comes from the fact that users need more time to register the content of some elements, e.g. a text of the same size as a simple image takes considerable more time to register in the human mind, see also [Mar02]. Therefore perceived emphasis is only applied to similar elements presented in a sequence. For example, two images are presented in sequence, and one is displayed longer, then that one receives higher perceived emphasis. This implicitly assumes that similar media elements require similar time to register in the human mind, which is also not necessarily true (e.g. a very simple image versus a complex detailed image).

- **Background color:** Varying background color of media elements is an example of perceived emphasis through supporting aspects. For our purposes we only view the variation of background colors as a means of using a shaded box to set some elements apart from the larger presentation around it.

- **Border:** Borders are another example of perceived emphasis through supporting elements. Placing a border around an atomic element or a group of elements clearly adds extra emphasis to the bordered elements. The possible unwanted effects a badly chosen border might have on disconnecting the bordered elements from the presentation by visually grouping them apart from the other elements are beyond the current scope.

Next we make a distinction in the strength of the emphasis aspects. The reason for this distinction is the way users perceive the emphasis aspects. Position and size are perceived as part of the inherent emphasis of the elements, even though they can be influenced, and are thus stronger means of emphasizing elements then the others. The latter two, background color and borders, are cognitively interpreted as means for the creator of the presentation to emphasize elements and may thus be cognitively be ignored by the user, especially if these emphasis elements are known to be commonly used to digress.

The distinction made above can be used to our advantage when applying emphasis to elements. By applying the lesser emphasis aspects, border and background color, to an element in a positive manner, while de-emphasizing it with the stronger aspects, size and position, we are able to create a situation in which elements are set out from other elements, yet are not conceived as most important by the user. This approach can be strengthened by making repeated use of it. A good example of this approach is the positioning of advertisements on the front page of newspapers — elements that are often bordered and have different backgrounds, yet are perceived as less important nonetheless as they are small and placed at the bottom. Another good example is the use of sidebars in magazines and textbooks. We will make use of this approach when dealing with distinct subtrees, as described in section 3.2.2.

3.1.3 Complicating Issues

In our approach for dealing with emphasis, we have simplified a number of situations. In this section we will identify these simplifications and indicate what future extensions should be made to deal with such limitations in the future.

The first simplification deals with our approach towards inherent emphasis. We have stated that for our use we assume that each element of the same type (except for composite elements) is treated as having the same inherent emphasis value. In reality this is not true. For most of the media element types we can distinguish inherent emphasis differences between instances. For example text element instances differ in inherent emphasis based on the font size and style. Likewise image element instances differ in inherent emphasis based on their content, complexity and use of colors.
To deal with this the levels of inherent emphasis might need to be adjusted, allowing for differences within a single element type and possibly even result in overlap from one type to another. For example large bold font text elements may have greater inherent emphasis than monochrome, simple images.

Another simplification was made in the use of colors, both with borders and background colors. As is discussed in [Man03], colors complicate matters in a far greater way than we have taken into account. The choice of the color for a border, in relation to the most prominent background color, may either bring the bordered element to the front or send it to the back. The same holds when using multiple background colors. In future work, the work presented here should be merged with the work presented in [Man03] to make optimal use of both pieces of research.

Finally, we would like to mention the aspect of synergy. This means that the combination of various emphasis elements may lead to unexpected and possibly unwanted synergy effects, where one media element provides emphasis to another element. A good example of this are titles within texts. A title is used to emphasize the elements below it. Likewise the position of an image surrounded by text usually gives increased emphasis to the text closest to it.

3.2 Distinct Subtrees

This section describes rules to identify parts of presentation trees, which we term distinct subtrees. First we will define distinct subtrees, then we provide the rules to identify them and finally we discuss a number of complicating issues.

3.2.1 Definition of Distinct Subtree

A distinct subtree is a subtree that is considered to be different from the other subtrees on the same level — those subtrees taking one of the siblings of the distinct subtree's base node as base.

It only makes sense to talk about distinct subtrees when there are more than two subtrees on a given level. If there is just one it can never be distinct, since there is nothing to compare it with. If there are two, they may differ in the dimensions described below, but it is impossible to make a decision about which one is the normal case and which one the special. Hence, only sets of three or more subtrees on the same level are taken into consideration.

There are three basic properties in which one subtree can differ from the others: size, importance and structure.
We thus define a distinct subtree as a subtree that differs with all of the other subtrees in at least two of these dimensions, while no two other subtrees differ in two or more dimensions among each other. This choice is rather arbitrary, distinctness in only one dimension is encountered too often and distinctness in all dimensions is encountered not often enough to prove workable.

Whether or not a distinct subtree is more or less important than the others is not the primary issue. Rather that it is a part of the presentation that stands out from its siblings. This is what should be reflected in the final presentation.

3.2.2 Identifying Distinct Subtrees

Based on the definition of distinct subtrees we derive a number of rules to identify them in a Structured Progression. To do this we describe the three properties — size, importance and structure — presented in the previous section in more detail and present criteria for setting the measures of difference

that a distinct subtree should satisfy.
Before any analysis is done on the subtrees, each subtree undergoes the same process for its own subtrees, i.e. any distinct subtrees present in the subtrees are removed for the comparison. The following rules are then used to determine whether there is a distinct subtree in any of the three properties:

- **Size:** The size property is one of the easiest to deal with when trying to identify distinct subtrees.
  First we define the size of a subtree as the number of media elements contained within that subtree. At first one might expect that most subtrees will differ in size, however often a scene will present a list of elements in a similar way. This enables us to set a factor by which a distinct subtree should differ from the other subtrees in size, while all pairs of the other subtrees should present no such difference.
  Alternatively we might define an absolute value by which a distinct subtree should differ from all others.
  We have chosen the first approach as it proved to generate the best results and is better suited to fit presentations and subtrees of varying sizes.

- **Importance:** The importance property is dealt with in a similar manner as the size dimension.
  Again we first make a number of assumptions. The importance of composite elements, those consisting of multiple media elements, is considered to be the average of the importance values of the constituent media elements in case no value is defined. The range of importance values any element can have is consistent throughout the entire presentation. A media element that is not assigned an importance value either receives the importance value of its encompassing composite, or, if this is unavailable, the default value, being the average of the range of importance values available.
  Now we can define a subtree to be distinct with its importance property in the same manner as we did for the size property. Again, as with the size property, we have the choice to define distinctness in importance by a factor or by an offset value.
  In contrast to the size property, we have opted for the offset approach with the importance property. This was done because we deem a subtree more likely to be distinct if it has an importance gap with all the others, rather than a factor, which might prove too narrow within a limited span of importance values.

- **Structure:** The structure property is by far the most difficult to deal with, to such an extent that we are still not fully pleased with the current approach.
  The current approach to deal with structure is a relatively simple one. First we compare the subtrees for complete similarity in structure. If this does not lead to a conclusive answer, usually because more than three subtrees are mutually different in structure, a more thorough comparison is made. The system will identify the building blocks used to make up the subtree. These building blocks are the length of the list and the of media elements in the composite. Any subtree can then be defined as a nesting of lists and composites.
  This results in an abstraction from the similarity comparison in that lists of differing length, but made up from the same composite building blocks can be considered as similar. The same holds for lists of equal length, but made up of composites of differing size. Further abstraction comes from flattening a list of lists.
  To achieve better performance of the transformation process, the structure property is only analyzed if one of the previous properties, size and importance, indicate the possible presence of a distinct subtree, but not yet present a conclusive answer, i.e. one indicates a certain subtree as being distinct, whereas the other does not.

3.2.3 Complicating Issues

In identifying distinct subtrees we encounter a number of complicated aspects.

By far the most complicating aspect is how to deal with the analysis of subtree structure. To obtain a good set of rules, that are general enough to be used with a large diversity of structured progressions, the rules would need to be fine tuned.

Another complicating aspect is the presence of distinct subtrees on lower levels when analyzing the subtrees on a higher level. Currently we keep distinct subtrees within a subtree out of the analysis. However, it might very well be that the presence of distinct subtrees on lower levels is another indicator that the subtree itself is distinct as well. Going even further, a nested sequence of subtrees might even indicate an important theme that runs through the presentation as a whole. These things are worthy of future analysis.

3.3 Structuring Mechanisms

This section presents rules for dealing with larger presentations. The generated presentations currently consist of a maximum of 20 atomic elements and have durations of less than 2 minutes. When these become larger, it becomes desirable to have tools in place to structure the content, effectively offering mechanisms for users to narrow their search further while watching the presentation.

We first describe some general structuring mechanisms. Then present an approach that is implemented in the Cuypers framework. Finally we discuss some complicating issues when structuring the presentation and how we have dealt with them.

3.3.1 Definition of Structuring Mechanisms

This section presents structuring mechanisms that can be used to structure larger presentations and details the one we have implemented in the system.

In our view, structuring mechanisms are mechanisms that add interactivity to the presentation in order to provide the user with a means of more quickly browsing through the contents of the entire presentation. With this definition contextual linking is also a form of structuring. We will not address this type in this report, as we feel much work in generating contextual links has already been done and such work could in the future be merged with the work presented here. A good overview of generating contextual links is presented in [CHBG01].

The most important and general structuring mechanism is the menu. Ever since graphical user interfaces have become the de facto standard (and even to some degree before that), structuring through the use of menus has been used extensively. An in depth discussion of the use of menus and positioning of elements in the field of web-sites can be found in [Cza02].

3.3.2 Identifying Structures in Presentations

As described in the previous section, menus are among the most widespread structuring mechanisms, therefore the use of menus was a logical first choice to apply to our presentations. The way we have done this is to convert each top level scene into an element of the menu. While doing this, we kept in mind the possible presence of distinct subtrees.

The final approach meant that the top level scenes are analyzed by the system, just as we would do when looking for a distinct subtree, as described in the section 3.2. In this case, when a distinct subtree is found, this subtree is ignored in the decision to create a menu and left out as a menu item if a menu is created. Then the remaining subtrees are compared to one another, if their size and importance is all similar, a menu is created. An exception is made in the case that no distinct subtree is present and there is no congruity between the subtrees either, then the only requirement they should meet is to be of appropriate size.

3.3.3 Complicating Issues

The most dire consequence of using interactive elements within a timed presentation is the fact that the duration of the presentation becomes undetermined. The way we dealt with this was to set the total duration of the presentation to the total duration of composing elements, assuming that the user might watch everything, but nothing twice, and add to that a interactivity value, which depend upon the number of options the user has or needs to interact with the presentation.

Of course, different approaches might be considered, especially when the duration of the entire presentation becomes much larger. In such a case the user might not have any desire to watch the entire half hour of the presentation, but only select the elements from the menu he deems most interesting or best suiting his query. In such a case a total duration of the presentation, estimated on the likeliness the user will watch a certain percentage of the total content may be better. However, the limited number of test presentations and their relative short duration, made such decisions impossible to make and they therefore remain for future research.

Chapter 4

Technologies

This chapter introduces a number of existing technologies that are required to implement the solution described in chapter 3. The first technology described is the Prolog logical programming language. The other technologies are a number of document formats. They are presented in the order in which they are encountered in the transformation process, starting with the Structured Progression in section 4.2 and then the Hypermedia Formatting Objects, section 4.3, and the SMIL standard in the final section.

4.1 Prolog

To implement our analysis rules we used the Prolog language, which is already used within the Cuypers system to transform the presentation from Structured Progression, see section 4.2, to Hypermedia Formatting Objects, section 4.3.

The actual Prolog implementation used is the ECL^{i}PS^e version [WNS97]. ECL^{i}PS^e offers as an added benefit the possibility to use constraints and constraint solvers to solve them. Extensive use is made of these constraints to get to a presentation that makes the best use of the vertical, horizontal and temporal space available to the system. The constraints used for these three dimensions are based on the temporal constraints presented by Allen [All83].

4.2 The Structured Progression

The Structured Progression intermediate format is an abstract representation of the final presentation. It represents the presentation without the issues of interactivity and spatial and temporal layout.

The Structured Progression consists of an abstract presentation which consists of one or more scenes. The term scene is only used here to represent a group of media elements that are grouped together in order to ensure that they are presented in the final presentation as a consistent whole. Each scene can by itself be built up from various sub-scenes, and so on. As such a scene may consist of media elements of various types and various levels of relevance pertaining to the subject of the overall presentation. In the current use of importance, defined as priority, of elements within the structured progression, these are assigned as a chapter numbering, meaning there are no importance gaps between scenes on the same level.

An example of a Structured Progression is given in figure 4.1. The scenes that build up the presentation are clearly indicated in the figure. There are three main scenes *Personal Life*, *Career* and *Private Life*. The latter of these is much larger than the other two and since it is the last in the sequence its importance is the lowest, as discussed in the previous paragraph. The *Career* scene has two sub-scenes: *Peter Lastman* and *Chiaroscuro*. The *Private Life* scene even consists of four sub-scenes: *Rembrandt's and Hendrickje's daughter*, *Titus van Rijn*, *Rembrandt's Mother* and *Saskia Uylenburgh*.

4.3 Hypermedia Formatting Objects

The Hypermedia Formatting Objects intermediate format is a representation of the final presentation that takes spatial and temporal constraints into account, but abstracted from the final form document format.

The Hypermedia Formatting Objects format provides for time-based multimedia presentations equivalent functionality that we are used to for text transformations. In [vOGRH03] the need for such a multimedia styling and formatting vocabulary is motivated and the following requirements are identified for it (based on [vOGRH03]):

- **Visual layout** The formatting vocabulary needs to be able to specify the *sizes* and *positions* of the areas in which the content is to be presented. Both *absolute* and *relative* positioning of elements should be available to authors.
- **Visual style** In addition to the visual layout mentioned above, it should also be able to specify the visual style of a presentation in terms of colors, fonts, paddings, borders, etc.
- **Temporal structure** The formatting vocabulary should have instruments to support not only the basic orchestration of presentations

```xml
<ps:presentation xmlns:ps="http://www.cwi.nl/~media/ns/ps#"
      priority="[1]" title="Biography of Rembrandt van Rijn">
      <ps:scene priority="[1, 1]" title="Personal Life">
            <ps:scene priority="[1, 1]" title="Rembrandt van Rijn">
                  <ps:composite priority="[1, 1, 1]">
                        <ps:composite priority="[1, 1, 1, 1]">
                              <ps:ps_media priority="unknown" media-type="text"/>
                        </ps:composite>
                        <ps:composite priority="[1, 1, 1, 2]">
                              <ps:ps_media priority="unknown" media-type="image"/>
                              <ps:ps_media priority="unknown" media-type="image"/>
                        </ps:composite>
                  </ps:composite>
            </ps:scene>
      </ps:scene>
      <ps:scene priority="[1, 2]" title="Career">
            <ps:scene priority="[1, 2]" title="Peter Lastman">
                  <ps:composite priority="[1, 2, 1, 1]">
                        <ps:ps_media priority="unknown" media-type="text"/>
                  </ps:composite>
            </ps:scene>
            <ps:scene priority="[1, 2]" title="Chiaroscuro">
                  <ps:composite priority="[1, 2, 1, 1]">
                        <ps:ps_media priority="unknown" media-type="text"/>
                  </ps:composite>
            </ps:scene>
      </ps:scene>
      <ps:scene priority="[1, 3]" title="Private Life">
            <ps:scene priority="[1, 3]" title="Rembrandt's and Hendrickje's daugther">
                  <ps:composite priority="[1, 3, 1, 1]">
                        <ps:ps_media priority="unknown" media-type="text"/>
                  </ps:composite>
            </ps:scene>
            <ps:scene priority="[1, 3]" title="Titus van Rijn">
                  <ps:composite priority="[1, 3, 1]">
                        <ps:composite priority="[1, 3, 1, 1]">
                              <ps:ps_media priority="unknown" media-type="text"/>
                        </ps:composite>
                        <ps:composite priority="[1, 3, 1, 2]">
                              <ps:ps_media priority="unknown" media-type="image"/>
                        </ps:composite>
                  </ps:composite>
            </ps:scene>
            <ps:scene priority="[1, 3]" title="Rembrandt's Mother">
                  <ps:composite priority="[1, 3, 1]">
                        <ps:composite priority="[1, 3, 1, 1]">
                              <ps:ps_media priority="unknown" media-type="text"/>
                        </ps:composite>
                        <ps:composite priority="[1, 3, 1, 2]">
                              <ps:ps_media priority="unknown" media-type="image"/>
                              <ps:ps_media priority="unknown" media-type="image"/>
                        </ps:composite>
                  </ps:composite>
            </ps:scene>
            <ps:scene priority="[1, 3]" title="Saskia Uylenburgh">
                  <ps:composite priority="[1, 3, 1]">
                        <ps:composite priority="[1, 3, 1, 1]">
                              <ps:ps_media priority="unknown" media-type="text"/>
                        </ps:composite>
                        <ps:composite priority="[1, 3, 1, 2]" discourse="denotative">
                              <ps:ps_media priority="unknown" media-type="image"/>
                        </ps:composite>
                  </ps:composite>
            </ps:scene>
      </ps:scene>
</ps:presentation>
```

Figure 4.1: Scenario Structured Progression

(which items plays when), but also be able to deal with synchronization relations and non-linear temporal structure, as may appear in interactive presentations.

- **Temporal style** Next to the visual style it also makes sense to talk about the temporal style, part of which is already determined by the orchestration of the document.

- **Support for top-down transformations** A template approach is required to allow the extraction of media items from a known collection to be placed in a consistent layout.

- **Support for bottom-up transformations** Bottom-up transformation implies that the choices for a composite formatting object depend on the layout of its children. Something which is useful when generating multimedia presentations, when the exact composition of media elements is not known until runtime.

- **Abstract from target output format** The formatting model should be able to express spatial, temporal and interactivity aspects without requiring information about the intended final presentation format.

- **Support for resource-constrained presentations** Multimedia presentations are often constrained in various dimensions (space and time) for various reasons (device and user). The system has more liberties, in space and time, as compared to paper-based documents, to move media elements around, but needs more information on layout configurations to know which ones do not violate the intended semantics of the presentation.

- **Support for generating consistent presentations** A multimedia formatting model makes it possible for authors to create consistent presentations.

All of the above requirements resulted in the development of the Hypermedia Formatting Objects, which is currently still work in progress. At this time it allows for the stacking of media and/or composite elements along a horizontal, vertical or temporal dimension through the use of encompassing boxes. These boxes are respectively called *hbox*, *vbox* and *tbox*. Also elements, media and composite, can be played in parallel using the *par* element. Pre-defined presentation schemes, such as slideshows, are being added to be used as templates for presentations, modelled in the *slideshow* element.

```xml
<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<hfo:root xmlns:hfo="http://www.cwi.nl/~media/ns/hfo#" title="Biography of Rembrandt van Rijn">
      <hfo:vbox ps_options-function="Top level vertical stacking of title with main content">
            <hfo:text ps_options-function="Main title of presentation"/>
            <hfo:tbox ps_options-function="Top level sequence of scenes of the presentation">
                  <hfo:hbox>                          <!-- Personal Life >
                        <hfo:hbox begin="0" end="6">        <!-- Rembrandt van Rijn >
                              <hfo:text begin="0" compduration="6"/>
                        </hfo:hbox>
                        <hfo:hbox begin="0" compduration="6">
                              <hfo:img begin="0" compduration="3"/>
                              <hfo:img begin="3" compduration="3"/>
                        </hfo:hbox>
                  </hfo:hbox>
                  <hfo:slideshow begin="6" end="45">  <!-- Career >
                        <hfo:hbox begin="6" end="30">       <!-- Peter Lastman >
                              <hfo:text begin="6" compduration="24" end="30"/>
                        </hfo:hbox>
                        <hfo:hbox begin="30" end="45">      <!-- Chiaroscuro >
                              <hfo:text begin="30" compduration="15" end="45"/>
                        </hfo:hbox>
                  </hfo:slideshow>
                  <hfo:slideshow begin="45" end="84"> <!-- Private Life >
                        <hfo:hbox begin="45" end="47">      <!-- Rembrandt and Hendrickje's daugther >
                              <hfo:text begin="45" compduration="2" end="47"/>
                        </hfo:hbox>
                        <hfo:hbox begin="47" end="68">      <!-- Titus van Rijn >
                              <hfo:hbox begin="47" end="68">
                                    <hfo:text begin="47" compduration="21" end="68"/>
                              </hfo:hbox>
                              <hfo:hbox begin="47" end="68">
                                    <hfo:img begin="47" compduration="21" end="68"/>
                              </hfo:hbox>
                        </hfo:hbox>
                        <hfo:hbox begin="68" end="80">      <!-- Rembrandt's Mother >
                              <hfo:hbox begin="68" end="80">
                                    <hfo:text begin="68" compduration="12" end="80"/>
                              </hfo:hbox>
                              <hfo:hbox begin="68" end="80">
                                    <hfo:img begin="68" compduration="6" end="74"/>
                                    <hfo:img begin="74" compduration="6" end="80"/>
                              </hfo:hbox>
                        </hfo:hbox>
                        <hfo:hbox begin="80" end="84">      <!-- Saskia Uylenburgh >
                              <hfo:hbox begin="80" end="84">
                                    <hfo:text begin="80" compduration="4" end="84"/>
                              </hfo:hbox>
                              <hfo:hbox begin="80" end="84">
                                    <hfo:img begin="80" compduration="4" end="84"/>
                              </hfo:hbox>
                        </hfo:hbox>
                  </hfo:slideshow>
            </hfo:tbox>
      </hfo:vbox>
</hfo:root>
```

Figure 4.2: Scenario Hypermedia Formatting Object Tree.

All types of media elements are modelled within a single *media* HFO object type.

An example of a HFO tree, representing the scenario identified in chapter 2, is given in figure 4.2. The correlation between this tree and the Structured Progression shown in figure 4.1 is immediately clear. Except for the added elements at the top of the tree — the title and stacking it with the content — there is a one-to-one correspondence between the scenes in the Structured Progression and those visible in the HFO tree.

The Hypermedia Formatting Objects format lacks however in terms of interactivity in its current implementation [vOGRH03], even though all of the intended output formats (SMIL and HTML+TIME) offer interaction capabilities. As described in chapter 2, this report tries to address this gap and has hence led to an extension of the HFO vocabulary.

4.4 The SMIL 2.0 Presentation Format

The SMIL 2.0 format is the World Wide Web multimedia standard developed by the W3C [Coh01]. It is one of the possible output formats of the Cuypers engine, and the one used in this report.

Currently the Cuypers engine makes use of only a limited set of the available options in SMIL 2.0. Based on the Amsterdam Hypermedia Model [Har98] it is also able to offer a wide array of interactive elements. The basic building blocks for presentations, such as *par* and *seq*, for parallel and sequential presentation of elements respectively, were used. But elements that allow for interactivity, such as *excl* for the exclusive presentation of a single element from a set at any one time, were not yet used. An example of a SMIL tree of the *Biography* scenario can be found in figure 4.3 below. Again the correlation with the Structured Progression (figure 4.1) and the HFO tree (figure 4.2) can clearly be seen. Besides the additions already present in the HFO tree, SMIL 2.0 separates the positioning of the elements from the actual content, by making use of *region* declarations in the *head* section. The remaining scenes again correspond to those in the previous trees on a one-to-one basis.

The resulting SMIL presentations were displayed using RealNetworks RealONE player, [Rea], and also checked to work with the latest RealPlayer 10 released in 2004. A screenshot of the *Biography of Rembrandt van Rijn* presentation can be found in figure 4.4 below. This means that text elements could be presented using RealText. Furthermore it meant that the final SMIL presentations had to be generated in such a way that they are compatible with RealONE. The implications of this are mentioned where appropriate.

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<smil>
<head>
<layout>
      ... contains region information required in SMIL 2.0 ...
</layout>
</head>

<body>
<par id="root1">
      <par id="vbox1">
            <text id="title"/> <!-- Main title of presentation >
            <seq id="tbox1">        <!-- Top level sequence of scenes >
                  <par id="hbox3">        <!-- Personal Life >
                        <par id="hbox1">        <!-- Rembrandt van Rijn >
                              <text id="hfo_text-25470"/>
                        </par>
                        <seq id="hbox2">
                              <img id="hfo_img-47825"/>
                              <img id="hfo_img-26412"/>
                        </seq>
                  </par>
                  <seq id="slideshow1">   <!-- Sequence of Career scenes >
                        <par id="hbox4">        <!-- Peter Lastman >
                              <text id="hfo_text-71208"/>
                        </par>
                        <par id="hbox5">        <!-- Chiaroscuro >
                              <text id="hfo_text-86948"/>
                        </par>
                  </seq>
                  <seq id="slideshow2">   <!-- Sequence of Private Life scenes >
                        <par id="hbox6">        <!-- Rembrandt and Hendrickje's daughter >
                              <text id="hfo_text-46820"/>
                        </par>
                        <par id="hbox9">        <!-- Titus van Rijn >
                              <par id="hbox7">
                                    <text id="hfo_text-99783"/>
                              </par>
                              <par id="hbox8">
                                    <img id="hfo_img-93487"/>
                              </par>
                        </par>
                        <par id="hbox12">       <!-- Rembrandt's Mother >
                              <par id="hbox10">
                                    <text id="hfo_text-28580"/>
                              </par>
                              <par id="hbox11">
                                    <img id="hfo_img-65214"/>
                                    <img id="hfo_img-22867"/>
                              </par>
                        </par>
                        <par id="hbox15">       <!-- Saskia Uylenburgh >
                              <par id="hbox13">
                                    <text id="hfo_text-17038"/>
                              </par>
                              <par id="hbox14">
                                    <img id="hfo_img-84357"/>
                              </par>
                        </par>
                  </seq>
            </seq>
      </par>
</par>
</body>
</smil>
```

Figure 4.3: Scenario SMIL 2.0 Tree.

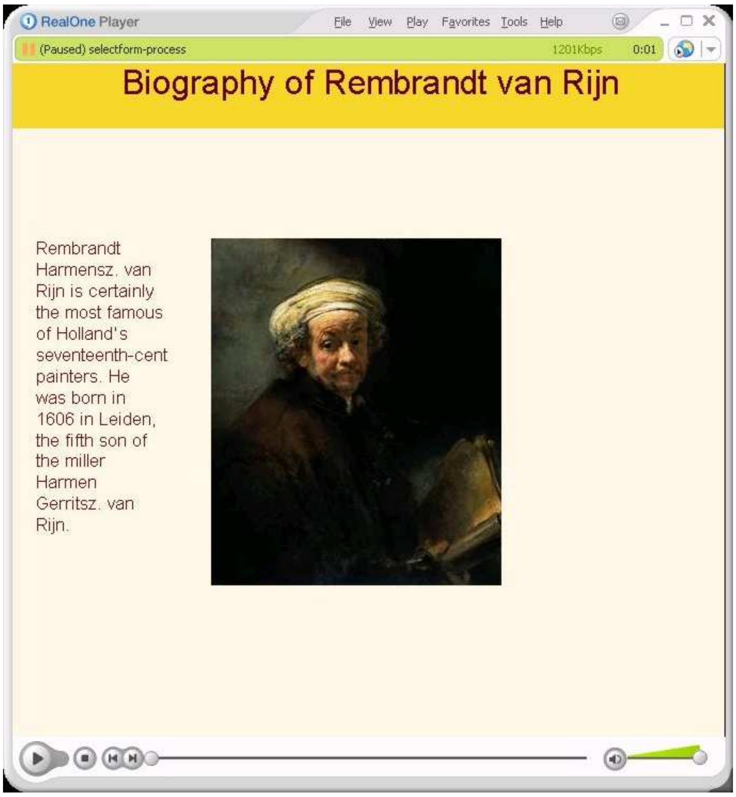

Figure 4.4: Screenshot of the generated *Biography of Rembrandt van Rijn* presentation in RealONE

Chapter 5

Realization and Implementation

This chapter derives the actual implementation of the solution first described in chapter 3 using the technologies described in the previous chapter. It presents the various elements in the same order as was done in chapter 3, starting with a description of the implementation of emphasis elements, followed by the distinct subtrees in section 5.2 and finally the structuring mechanisms.

5.1 Emphasis

In this section we will describe how we have included the use of the emphasis concept into our system. First we will start with inherent emphasis, then we discuss perceived emphasis.

5.1.1 Inherent Emphasis

Including the concept of inherent emphasis into the system was done by checking the type of the element. Each media element type was assigned an inherent emphasis value stored in a configuration file. Video elements received the highest value, followed by image elements and finally text elements received the lowest inherent emphasis value. For each composite element, a requested asking it for its inherent emphasis resulted in request to all of its children for their inherent emphasis, after which the average was taken.
In the future it might be desirable to take the inherent emphasis out of the configuration file and subsequently the querying of elements for their type out of the transformation process and store the inherent emphasis as

an attribute value with each object. In this manner the actual decisions on inherent emphasis values could be moved up in the transformation process, allowing previous transformation steps to make decisions about these values.

5.1.2 Perceived Emphasis

In order to work with perceived emphasis, all the elements identified should be available. This meant that the Prolog engine and the Hypermedia Formatting Objects (HFO) Vocabulary had to be extended to provide a means of creating all of the emphasis types.

- **Position:** Before the changes made with this work, the elements were positioned either vertically (*vbox*), horizontally (*hbox*) or temporally (*tbox*) in the order in which they appeared in the *priorityList* element of the parent element.
  After the changes, it is possible to change this presentation order, by shifting the order of the elements within the *priorityList*.

- **Size:** Before the changes the elements were presented in the default size they required and media elements such as images were never scaled. Now it is possible to decide to scale media elements, such as images. This is done by forcing the system to attempt to generate a presentation where the constraints for the element to be scaled are at first limited to a smaller range. Effectively, we try to force the system to present the image in a smaller or larger area than it would normally require. Only when this fails, we resort back to the original approach and try to generate a presentation with the element in its unscaled size.

- **Background color:** In the original approach, when all the HFO elements were generated based on their corresponding Structured Progression elements in a bottom-up manner, all elements were assigned the default background color. The only exception to this rule was the separately created title element, which was assigned a different background color to set it apart from the content of the presentation.
  With the changes, the distinct subtree analysis (see section 5.2) is done before the assignment of background colors. When this analysis finds a distinct subtree, which needs to be made distinct in the manner of presentation, the system then assigns a different background color to this distinct subtree, whereas the others all receive their background color according to the original approach. Examples of this can be found in figures 5.4 and 5.5. Actually, the latter is a bit of a dubious example,

Figure 5.1: Mock-up of a bordered element in a presentation.

as the relative large size of the shaded element still tends to draw too much attention. As was described in section 3.1.2 this is due to the fact that size is a stronger emphasis element than background color.
Currently, as we abstain from complicating color issues, the differing background color taken is the background color used for the title element, just as a means to prove the viability of the approach. Were we to use a color selection mechanism, such as described in [Man03], we could pick a less attentive color for the shaded box in figure 5.5 to have it draw less attention.

- **Border:** Before our work, there was no possibility to create borders around elements.
  The approach taken is relatively straight forward. We take an element of either *hbox* or *vbox* type and create a new element of the same type around this element. The new element receives a different background color from the existing one, effectively creating a border as long as the assigned size of the new element is bigger than that assigned to the original element. The new element replaces the original element in the *priorityList* of its parent. The difficulty was in assuring the new element became larger than the original. This was done by requiring the system to assign a certain minimum value to the new elements *border* property whenever possible. A result of a bordered element can be seen in figure 5.1.

Note that we have not supplied the option of creating borders around *tbox* elements.

5.2 Distinct Subtrees

In order to be able to apply the rules we identified for distinct subtrees, we first needed to implement the rules we defined to identify such subtrees in a Structured Progression. This resulted in two Prolog predicates, each designed to analyze (a part of) the Structured Progression for the presence of distinct subtrees based on a particular property.

The first predicate deals with the size property. It makes an indexed nested list of the size of the tree and all lower elements. Then it compares the sizes of the various subtrees. The actual implementation can be found in figure 5.2.

The next predicate deals with the importance property. Due to the current implementation of priorities in the Cuypers system, as discussed in section 3.2, the only option left for implementation was to check whether the possible distinct subtree was either the first element (most important) or the last element (least important) in the list. The predicate is shown in figure 5.3.

As described in section 3.2.2 we will make use of emphasis elements to express a distinct subtree's distinctness. We distinguish between two cases, based on the importance of the distinct subtree, which is either lower than that of the other subtrees or at least equal to it.
In the case the distinct subtree has an importance at least equal to the other subtrees, we apply any number of emphasis elements to the subtree to guarantee that it has a greater total emphasis than the other subtrees. An example of this is shown in figure 5.4.

In the other case, when the distinct subtree is of lower importance than the other subtrees, using emphasis at first doesn't seem the logical approach. However, as we have advocated in section 3.1.2, by making consistent and subtle use of emphasis elements, it can also be applied to emphasize elements that are less important. A distinct subtree of lower importance is therefore first emphasized using one of the lesser emphasis elements, either a border or a differing background color. Then the system will guarantee an overall lower emphasis of the distinct subtree, by maneuvering the position and size of the distinct subtree relative to the others in such a way that it lowers the perceived emphasis of the subtree. An example of this approach is given in figure 5.5.

```
:-protected(analyzeSize/3).
analyzeSize([], _, _).
analyzeSize([HIn|TIn], PriorityList, Result):-
% STEP 1: Size comparison. For each subtree do leafNodeCount, use factor 2 for now
% input is a list from makeTreeIndex, so take each element in list and call leafNodeCount
% [HIn|TIn] = makeTreeIndex list of numbers for subtree, first element is number of children, so ignore
% PriorityList = list of PS objects, the same subtree

       ( foreach(X, PriorityList), foreach(Y, OutList) do
              (
              X::priorityList(Prior),
              X::makeTreeIndex(Prior, InBetween),
              X::leafNodeCount(InBetween, Y),
              )
       ),

       MaxCount is max(OutList), MinCount is min(OutList),

       % if there is no factor 2 difference between these, than no need to look further
       *(MinCount, 2, TwiceMinCount),
       ( (MaxCount >= TwiceMinCount) ->
              ( % continue, first find out which one is the distinct value,
                     setval(count, 0),
                     ( foreach(Z, OutList), param(MaxCount) do
                            ( % first max test
                                   *(Z, 2, TwiceZ),
                                   ( MaxCount >= TwiceZ -> ( getval(count, TempX),
                                                             incval(count),
                                                             ) ;
                                                             true
                                   )
                            )
                     ),

                     getval(count, MinusMaxLength),
                     length(OutList,TotalLength),

                     ( TotalLength is MinusMaxLength+1 ->
                            % Max is distinct, get index of MaxCount, look no further
                            (
                            setval(tempresult, 0),
                            (fromto(OutList, In, Out, []), param(MaxCount) do
                                                  (
                                                  In=[X|PossibleOut],
                                                  *(X,2,TwiceX),
                                                  ( (MaxCount >= TwiceX) ->    ( incval(tempresult),
                                                                               Out = PossibleOut,
                                                                               );
                                                                               ( incval(tempresult),
                                                                               Out = [],
                                                                               )
                                                  )
                                                  )
                            ),

                            getval(tempresult, TempResult),
                            );

                            % Similar code for min value if max wasn't distinct
                                   ...
                                   ...
                            );
                            % neither Min or Max is distinct, return 0
                            TempResult is 0,
                            true
                            )
                     )
              );
              true
       ),
       Result is TempResult.
```

Figure 5.2: The Prolog predicate analyzing the size property.

```
:-protected(analyzeImportance/3).
analyzeImportance([],_,_).
analyzeImportance(PriorityList, TempResult, FinalResult) :-
% PriorityList: the ordered priority list of the tree to be analyzed
% TempResult: the temporary result of analyzeSize, the index of the most likely distinct subtree

      length(PriorityList, ListLength),
      % if TempResult is either 1, the first subtree, or ListLength, the last subtree,
      % then keep distinct, else return 0
      ( (TempResult is 1) ->
                              FinalResult is TempResult
                              ;(
                              (TempResult is ListLength) ->
                                    FinalResult is TempResult
                                    ;
                                    FinalResult is 0
                              )
      ),
      true.
```

Figure 5.3: The Prolog predicate analyzing the importance property.

Figure 5.4: Mock up of a distinct subtree with greater importance. The shaded box represents a definition given as primary information of greater importance than the example images.

Figure 5.5: Mock up of a distinct subtree with lower importance. The shaded box represents a definition given as additional information of lower importance than the core presentation content.

5.3 Structuring Mechanisms

The implementation of the menu structure is straightforward and relies heavily on the *excl* construct found in the SMIL 2.0 vocabulary. This *excl* element makes it possible to play at most one element from the set of children at any one time in the same spatial area. Thus, we created elements, the menu items, that each activate the appropriate child of an *excl*-like element, which we called the HFO *xbox*. This *xbox* then requires a spatial area large enough to display the content of each of its children appropriately. To achieve the complete menu structure, the menu (either an *hbox* or *vbox*) needs to be nested with the *xbox*, that contains the content. A menu created as an *hbox* needs to be nested vertically with the content in a *vbox*, i.e. a horizontal menu is placed above the content. Likewise, a *vbox* menu comes left of the content through nesting in an *hbox*.

In order to create the menu items, a similar approach was used as was done for the title element of the presentation. For each menu item the title of the scene corresponding to that menu item was taken as the text displayed on the menu item. Likewise, the menu items were assigned the same background color as the title. Finally all menu items were added as children to the *hbox* or *vbox* representing the entire menu, as described above.

The resulting HFO constructs can be seen below in figure 5.6, where the

```
<hfo:vbox id="vbox1"
      left="0" right="450" top="115" bottom="676" begin="0" end="84"
      ps_options-function="Vertical stacking of menu and content"
      alignTime="true">
      <hfo:hbox id="hbox1"
            left="4" right="450" top="115" bottom="133" begin="0" end="84"
            ps_options-function="Top level horizontal menu of the presentation"
            alignTime="true">
            <hfo:text id="menu-item-1"
                  left="4" right="189" top="115" bottom="133" begin="0" end="84"
                  ps_options-function="Menu Item = title of scene"
                  type="text"
                  activates="tbox3">
            </hfo:text>
            <hfo:text id="menu-item-2"
                  left="189" right="279" top="115" bottom="133" begin="0" end="84"
                  ps_options-function="Menu Item = title of scene"
                  activates="slideshow1">
            </hfo:text>
            <hfo:text id="menu-item-3"
                  left="279" right="450" top="115" bottom="133" begin="0" end="84"
                  ps_options-function="Menu Item = title of scene"
                  activates="slideshow2">
            </hfo:text>
      </hfo:hbox>

      <hfo:xbox id="xbox1"
            ps_options-function="Top level scenes of the presentation"
            interrupt="ignore"
            onFinish="return">
                  <hfo:tbox id="tbox3">               <!-- activated by menu-item-1 >
                        <hfo:tbox id="tbox1">
                              <hfo:text></hfo:text>
                        </hfo:tbox>
                        <hfo:tbox id="tbox2">
                              <hfo:img></hfo:img> <hfo:img></hfo:img>
                        </hfo:tbox>
                  </hfo:tbox>
                  <hfo:slideshow id="slideshow1">     <!-- activated by menu-item-2 >
                        <hfo:tbox id="tbox4">
                              <hfo:text></hfo:text>
                        </hfo:tbox>
                        <hfo:tbox id="tbox5">
                              <hfo:text></hfo:text>
                        </hfo:tbox>
                  </hfo:slideshow>
                  <hfo:slideshow id="slideshow2">     <!-- activated by menu-item-1 >
                        <hfo:tbox id="tbox6">
                              <hfo:text></hfo:text>
                        </hfo:tbox>
                        <hfo:tbox id="tbox9">
                              <hfo:tbox id="tbox7">
                                    <hfo:text></hfo:text>
                              </hfo:tbox>
                              <hfo:tbox id="tbox8">
                                    <hfo:img></hfo:img>
                              </hfo:tbox>
                        </hfo:tbox>
                        <hfo:tbox id="tbox12">
                              <hfo:tbox id="tbox10">
                                    <hfo:text></hfo:text>
                              </hfo:tbox>
                              <hfo:tbox id="tbox11">
                                    <hfo:img></hfo:img> <hfo:img></hfo:img>
                              </hfo:tbox>
                        </hfo:tbox>
                        <hfo:tbox id="tbox15">
                              <hfo:tbox id="tbox13">
                                    <hfo:text></hfo:text>
                              </hfo:tbox>
                              <hfo:tbox id="tbox14">
                                    <hfo:img></hfo:img>
                              </hfo:tbox>
                        </hfo:tbox>
                  </hfo:slideshow>
      </hfo:xbox>
</hfo:vbox>
```

Figure 5.6: HFO structures representing a menu. The relations between menu items and content sequences are indicated through the use of boldface.

```
<par id="vbox1" region="vbox1-region" begin="0" dur="84">        <!-- Code representing menu items >
      <par id="hbox1" region="hbox1-region" begin="0" dur="84">
            <text id="menu1" region="menu1-region"/>             <!-- menu item 1 >
            <text id="menu2" region="menu2-region"/>             <!-- menu item 2 >
            <text id="menu3" region="menu3-region"/>             <!-- menu item 3 >
      </par>
</par>

....

<excl id="xbox1" begin="0" dur="84">                             <!-- excl encompassing all content >
      <seq id="tbox3" begin="menu1.activateEvent" dur="14">      <!-- content behind menu item 1 >
            <seq id="seqtest1" end="tbox3.end" dur="14">
                  <text id="hfo_text-25470"/>
                  <img id="hfo_img-47825"/>
                  <img id="hfo_img-26412"/>
            </seq>
      </seq>
      <seq id="slideshow1" begin="menu2.activateEvent" dur="39">  <!-- content behind menu item 2 >
            <seq id="seqtest2" end="slideshow1.end" dur="39">
                  <text id="hfo_text-71208"/>
                  <text id="hfo_text-86948"/>
            </seq>
      </seq>
      <seq id="slideshow2" begin="menu3.activateEvent" dur="51">  <!-- content behind menu item 3 >
            <seq id="seqtest3" end="slideshow2.end" dur="51">
                  <text id="hfo_text-46820"/>
                  <text id="hfo_text-99783"/>
                  <img id="hfo_img-93487"/>
                  <text id="hfo_text-28580"/>
                  <img id="hfo_img-65214"/>
                  <img id="hfo_img-22867"/>
                  <text id="hfo_text-17038"/>
                  <img id="hfo_img-84357"/>
            </seq>
      </seq>
</excl>
```

Figure 5.7: SMIL code representing the menu in the *Biography* scenario. The first part is the SMIL code representing the menu and the menu items, the second represents the content. For clarity and RealONE compatibility nested **tbox** structures from Figure 5.6 were collapsed into a single **seq** structure.

example scenario has been transformed into a presentation structured using a menu structure as presented above.

This lead to a number of extensions of the Hypermedia Formatting Objects (HFO) vocabulary. The first came in the form of a new *xbox* element, or Exclusive Box. The next came in the shape of the desire to link certain content behind and interactive element, resulting in the *activates* property, which indicates the identifier of the element it activates, which is usually the child of an *xbox*. The resulting SMIL code and a screenshot of a presentation containing a menu can be seen below in figure 5.7 and figure 5.8 respectively.

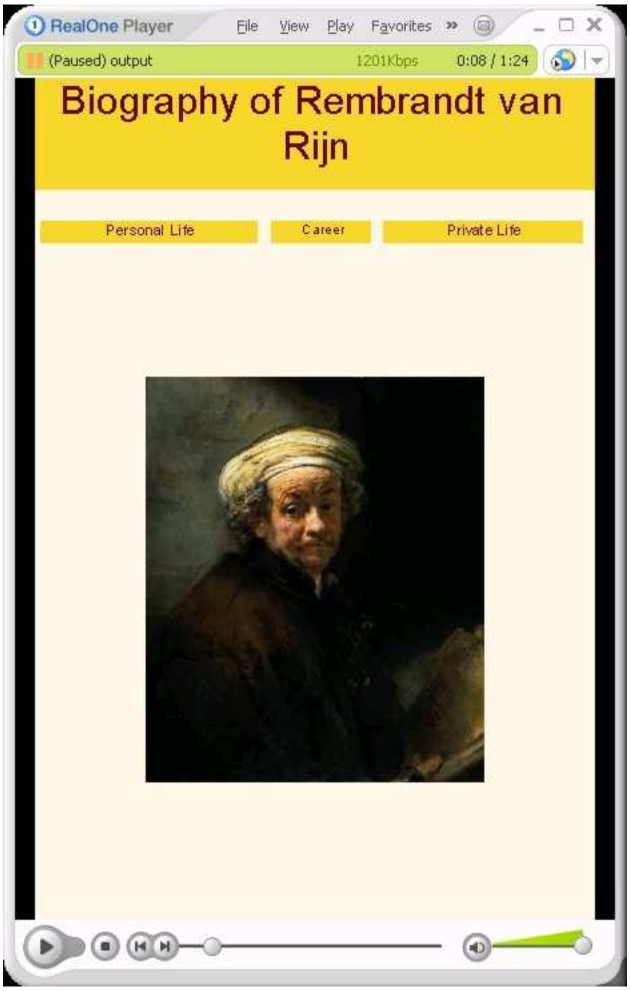

Figure 5.8: Screenshot of the generated *Biography* presentation with menu structure in RealONE.
The menu can clearly be seen positioned below the main title of the presentation.

Chapter 6

Discussion

6.1 Conclusion

Applying structural analysis on an abstract presentation schema, such as the Structured Progression, is a powerful domain-independent approach to make decisions about the layout of a final presentation.

Our approach has proven particularly applicable in structuring the presentation and identifying irregularities. This resulted in the possibility to generate presentations containing two new building blocks: structuring mechanisms and emphasized elements.

The effort to include structuring mechanisms as building blocks in the generated presentations resulted in the creation of menu elements. Menus are valuable to the generated presentations especially when these presentations become larger. Even though it is an achievement to be able to generate presentations containing hundreds of media elements that take an half an hour or more to run from start to end, there is a real danger that such presentations lose user interest. This may especially happen when presentation generation is used as a front-end to information retrieval, in which case a user may be uncertain about his actual query. In this case it is desirable for the user to be able to browse through the presentation to the scenes that present the information he needs most.

The option to emphasize elements, both in an attention intrusive (background colors and borders) and less intrusive manner (position and size), is a valuable addition. Especially the combination of emphasis with distinct subtrees seems to offer valuable solutions, as it enables us to express the

distinctness of the subtree in the final presentation. The attention intrusive emphasis aspects allow us to set the distinct subtrees apart from the larger presentation around it, while we can maintain the correct perceives importance by making use of the less intrusive aspects.
When adjudicating the means by which we have chosen to emphasize elements, the less intrusive options, position and size, can be used to great lengths and pose little nuisance to the user, while possibly strengthening the intended message of the presentation. The intrusive options, background colors and borders, should be used sparingly and with care, or they are quickly likely to become a nuisance to the user and thus undermine the meaning of the presentation as a whole.
Possible uses of the intrusive emphasis elements in future presentation generation systems might include using it to indicate elements tailored specifically to a user based on a user profile. This might include such elements as extra explanations for users with a limited grasp of the information presented.

6.2 Future Work

The majority of the future work comes from the complicating issues identified for the three main aspects of our work, *Emphasis*, *Distinct Subtrees* and *Structuring Mechanisms*, in the chapter detailing the theory, chapter 3. Below we will give an overview of the most important issues identified.

The first issue concerns the simplifications made when dealing with inherent emphasis of media elements. We have stated that for our use we assume that each element of the same type is treated as having the same inherent emphasis value. In reality this is not true. For most of the media element types we can distinguish inherent emphasis differences between instances. To deal with this the levels of inherent emphasis might need to be adjusted, allowing for differences within a single element type and possibly even result in overlap from one type to another. Current research done by Bacharova will most surely offer insights in how to deal with this complicated matter. As mentioned the work of Martinez [Mar02] already presents insights into the nature of the various media element types.

We have not delved into the issues related to the use of various colors within a single presentation. In future work, the work presented here should be merged with the work presented in [Man03] to make optimal use of both research efforts in the field of automated presentation generation.

In our discussion of distinct subtrees, we were forced to admit that we have no satisfying means of analyzing the structure of subtrees. In order to come up with a good set of rules to analyze subtree structure, and even decide whether or not such analysis is a viable approach, we would need to study numerous presentation trees and base our rule set on the common structures encountered in those subtrees. Such research wasn't possible at this time, as we were only able to work with a very limited number of structured progressions and no definitive conclusions could be drawn about common structures within such trees.
Only when these rules are in place and the structure analysis has proven viable, does it become possible to decide on expansions of the application domain of the distinct subtree concept.

The most dire consequence of using interactive elements within a timed presentation is the fact that the duration of the presentation becomes undetermined. The way we dealt with this was to set the total duration of the presentation to the total duration of composing elements, assuming that the user might watch everything, but nothing twice, and add to that an interactivity value, which depend upon the number of options the user has or needs to interact with the presentation.
This of course is by no means a satisfying solution. Especially when one considers the addition of other interactive elements, for example through contextual links. The best approach to come up with a desirable solution to this issue is through experimentation. As a relative new issue there is little related research available and only application in the real world can confirm the way in which users will respond to interactivity in temporal presentations.

After this work we are left with a mixed feeling. On the one hand we have done a good exploration of the available possibilities to enhance the layout and structure of the generated presentations. On the other we haven't come up with a satisfying means to make decisions on when and how to use those possibilities. The structural analysis approach hasn't proven itself as the definitive answer yet. Further research, both into the proposed structural analysis and possible other approaches is needed in order to be able to make optimal use of the enhancements to the generated layout described in this work.

Bibliography

[All83] James F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–844, November 1983.

[BKKR01] J. Bateman, J. Kleinz, T. Kamps, and K. Reichenberger. Towards constructive text, diagram and layout generation for information presentation. *Computational Linguistics*, 27(3):409–449, September 2001.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, May 2001.

[CHBG01] L. Carr, W. Hall, S. Bechhofer, and C. Goble. Conceptual linking: Ontology-based open hypermedia. 2001.

[Cla99] James Clark. *XSL Transformations (XSLT) Version 1.0 W3C Recommendation*. W3C, November 16 1999. Available from World Wide Web: http://www.w3.org/TR/xslt/.

[Coh01] Aaron Cohen. *Synchronized Multimedia Integration Language (SMIL) 2.0 Recommendation*. W3C, August 7 2001. Available from World Wide Web: http://www.w3.org/TR/smil20/.

[Cza02] K. 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.

[GBvOH03] J. Geurts, S. Bocconi, J. van Ossenbruggen, and L. Hardman. Towards ontology-driven discourse: From semantic graphs to multimedia presentations. In *Second International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, October 20-23 2003.

[Geu02] J. Geurts. Constraints for multimedia presentation generation. Master's thesis, University of Amsterdam, 2002.

[Har98] L. Hardman. *Modelling and Authoring Hypermedia Documents*. Phd thesis, University of Amsterdam, 1998. ISBN: 90-74795-93-5, also available at http://www.cwi.nl/ lynda/thesis/.

[Man03] A. Manniesing. Creating harmonious and legible colour schemes in the automated generation of multimedia presentations. Master's thesis, Delft University of Technology, 2003.

[Mar02] O.R. Martinez. Design dependencies within the automatic generation of hypermedia presentations. Master's thesis, Technical University of Catalonia, June 30 2002. Published as CWI technical report INS-R0205.

[MMT89] W.C. Mann, C.M.I.M Matthiesen, and S.A. Thompson. Rhetorical structure theory and text analysis. Technical Report ISI/RR-89-242, Information Science Institute, University of Southern California, 1989.

[RAB+03] L. Rutledge, M. Alberink, R. Brussee, S. Pokraev, W. van Dieten, and M. Veenstra. Finding the story — broader applicability of semantics and discourse for hypermedia generation. In *Proceedings of the 14th ACM conference on Hypertext and Hypermedia*, Nottingham, UK, August 26-30 2003.

[Rea] RealNetworks, Inc, http://www.real.com/. *RealPlayer*.

[Rij] Rijksmuseum Amsterdam. *Rijksmuseum Amsterdam*. Website: http://www.rijksmuseum.nl/.

[vOGRH03] J. van Ossenbruggen, J. Geurts, L. Rutledge, and L. Hardman. Towards a multimedia formatting vocabulary. 2003.

[W3C99] W3C. *Resource Description Framework (RDF) Model and Syntax Specification*, February 22 1999. Available from World Wide Web: http://www.w3.org/TR/REC-rdf-syntax.

[WNS97] M. Wallace, S. Novello, and J. Schimpf. *ECLiPSe: A Platform for Constraint Logic Programming*, 1997.