A GENERIC ARCHITECTURE FOR AUTOMATED CONSTRUCTION OF MULTIMEDIA PRESENTATIONS

FREDERIC BES, MURIEL JOURDAN AND FARID KHANTACHE

Unité de Recherche INRIA Rhône-Alpes
ZIRST, 655 avenue de l'Europe, Montbonnot, 38334 St Ismier CEDEX
France
E-mail: {Frederic.Bes,Muriel.Jourdan,Farid.Khantache}@inrialpes.fr

We are interested in applications that automatically generate a dynamic multimedia presentation adapted to the needs of the user from a database of XML fragments of basic elements (video, images, paragraphs, etc.). In this paper we present an architecture which goal is to support developers of such applications. This architecture takes benefits from both the use of transformation sheets and the use of a constraint solver. Transformation sheets are used to provide different renderings that depend on discrete parameters; whereas the constraint solver aims to handle continuous parameters and to optimize both content selection and formatting instructions taking into account global criteria like the total duration of the generated multimedia document. We illustrate the use of this architecture by presenting an application that builds a personalized TV sport news automatically.

1 Introduction

TV-like multimedia presentations will be more and more used to communicate information: training courses, medical reports, news, on-line shopping catalogues, etc. However, designing high-quality multimedia presentations is known to be a complex, time-consuming and error prone task [3,5,10] whatever is the authoring tool used. Moreover, the information overload problem associated with the Internet and the frequent updates of some information like news or medical reports raise a crucial need of information services that are customized for each individual user [4,15]. Typical examples of such information services are: educational courses adapted to the student's knowledge, automatic design of medical reports, personalized TV-news.

Whereas many works have been done in those areas on the run-time adaptation of the navigational structure of a hypermedia document [8], few works focus on the automatic generation of multimedia presentations. Moreover existing works often propose solutions linked to some particular area and it is not easy to evaluate how these specific experiments can help to implement other kind of applications. This is why we propose an architecture to help the development of personalized multimedia information services.

Starting from the twofold analysis that using transformation sheets is today the best way to associate different presentations to the same set of objects depending on discrete parameters; and that global criteria, like a user preference on the total

duration of the multimedia document get at the end, are easy to handle by using optimization techniques, the architecture takes benefits from these two technologies.

We illustrate the use of this architecture by presenting an application that builds a personalized TV sport news taking into account, among others parameters, the favorite teams of the user, the style of the summary she/he wants to see.

The second section introduces the area of automatic document generation. The third one is devoted to a structured presentation of related works that explains why we base our application on the use of transformation sheets and optimization techniques. The fourth section makes an overview of the architecture before presenting it in details through the development of a working example in the last section of this paper.

2 A KIND OF DECISION PROCESS

We consider in this paper that a personalized multimedia presentation system is an application that takes some user parameters (content preferences, capabilities...) and technical parameters (like screen size or bandwidth) and generates a dynamic multimedia presentation from a set of XML fragments. Dynamic multimedia presentation means that objects are organized both in time, space and hyperspace (hyperlink structure). We do not make any hypothesis on how user parameters are obtained: either by asking the user to answer some questions or by using some user modeling techniques, like collaborative filtering [20].

The two following examples will help the reader to understand the kind of applications we are interested in:

- personalized sport report: deliver a TV-like report after a championship day to a user who can choose the total report duration, the teams for which she/he wants to have detailed information (her/his favorite teams), the kind of the report: an exhaustive one (at least some minimal information about each match is given) or not (the maximum information about each favorite team is given). This is the example detailed later on.
- personalized medical survey: deliver a multimedia presentation that describes the main patient surgeries by using synchronization between a graphical representation of a patient body, some speech and textual informations [5].

The development of such kind of applications can be thought as a decision process that has to handle two kinds of decision:

content selection: which objects must participate to the final multimedia presentation ?

object organization: how objects are organized in time, space and hyperspace dimensions ?

One difficulty encountered in these applications is to know in which order content selection and content organization (through the three dimensions) must be done. For

instance, deciding if an object with an intrinsic duration d has to be display in the presentation which whole duration must be D (this is a user's request) strongly depends on the temporal organization of this presentation. On the other hand, the temporal order between two objects may depend whether another object is present or not in the final presentation. Actually, there is no strong argument in favor or against one solution: in some situations it is better to select organization before content (in order to perform a relevant selection), while in other ones the content has to be selected first, for instance to adapt the spatial organization to the selected content. The example of articles presentation from a catalog can illustrate this problem.

3 Related works

In order to understand the choices made in the architecture we propose, we present different existing methods that are used to develop automatic generation of presentation. We have group them together in the five following approaches.

3.1 Programming based approach

It is obvious that programming languages like JAVA can be used to develop a system that generates personalized multimedia presentation. On the Web, this approach is mainly used by CGI scripts. Well known disadvantages of this approach are: their lack of independence between the programming code and the piece of information; the difficulty to reuse some parts of an application inside another one; …

3.2 Using templates and selection instructions

Thanks to the use of more descriptive languages like XSLT [25], Northolk [16] or more imperative ones like PHP [17], JSP [12], ASP [14] it is possible to write virtual documents (they are dynamically generated on demand) that merge static data, i.e. information that does not change, with dynamic ones, i.e. information dynamically extracted from external data sources. This approach can be thought as the design of a template in a presentation language (HTML for web pages or SMIL [23] for multimedia presentation) that contains the static part of the document and some selection instructions that express how to collect the dynamic data (like query instructions in a database). It suits well to applications in which content selection can be split into several local database requests. But this approach does not allow to handle easily global content selection criteria like: choose the set of videos that maximize the user interest and such that the sum of video duration is lower than D. Moreover even if the developer implements global criteria through successive requests it is done at the expense of time performances of the application.

3.3 Using a set of transformation sheets

Transformation sheets are a well known way to associate different presentations (like different layout organizations) with the same set of objects. These presentations are usually deduced from a logical structure (for instance a DTD) associated with the objects. Personalization can be achieved by associating one transformation sheet with each parameter value that controls the application.

An example of such kind of architecture can be found in [26] in which the user can choose between three different styles to present the same set of objects. The obvious limitation is that it is only possible to handle discrete parameters.

In [19] more flexible transformation sheets are presented. They are based on constraints use. For instance the document layout can be expressed in the style sheet by using high-level constraints (centering, alignment, etc.) that will be solved by using a constraint solver. Doing so, the application has some possibilities to adapt the document to the size of the user's screen. However some drawbacks remain, since it does not provide the designer with a powerful way to select information, only filtering on some local criteria can be easily expressed.

3.4 Optimization algorithms

A lot of works in the area of automatic generation application put the emphasis on its content selection step and formalize it as an optimization process [7,13,15].

For example, in [13] the system tries to create a TV news program that fits the user interests (a lot of international political news followed by few sport news for instance) and a maximal duration constraint. The database contains a set of individual news classified by categories. Moreover a numerical value is associated with each news to express its degree of importance. The content selection problem is then equivalent to find a news subset that maximize the user interest while respecting the duration constraint. Then the authors propose a heuristic that makes the resolution of this NP problem more efficient. Once the content selection has been done, the TV news program simply consists in concatenating the selected elements (i.e. sequential presentation). A similar approach is used in [15] to generate music program (a concatenation of music) adapted to user preferences (at least 30% female-type voice or at most 20% "Jazz style" for instance).

These works seem to be very efficient for selecting content on global criteria but they mainly address some specific applications (the proposed heuristic strongly depends on addressed problems) and do not propose a more general solution to ease the design of various kinds of generation applications. Moreover, they only consider a simple and unique way to temporally organize the selected content (spatial and hyperlink organization are not addressed at all).

Only the approach described in [7] takes the temporal organization issue into account but in such a way that the content selection is done without any knowledge on the future organization of the objects (content selection is performed before temporal organization decision). As a consequence the selected content is often not

adapted to the user request. For instance, ever if the user wants a multimedia document of 30 min, she/he may get something significantly shorter than the wished 30 min.

3.5 Knowledge based approach

People from knowledge based area make some propositions to use their experiment to solve “automatic generation like” problem [1,4,5,18]. The idea is to use different knowledge bases (on the content properties, on some principles for temporal organization, on graphical rules…) in order to be able to answer to a large variety of requests (even those not planned when the system was designed) by decomposing the initial request in some sub-goals to reach. Thus, the content selection and the objects organization decisions are broken into several steps and are distributed through the application. This distributed way used to handle decisions makes global criteria difficult to take into account.

3.6 Synthesis

Application based on transformation sheets or templates have the same kind of advantages and limitations: they suit well to associate different well-fashioned rendering (depending on discrete parameters) to a set of existing objects. However, they fail to easily express global criteria. On the contrary, formalizing the automatic generation application as an optimization problem provides the designer with a powerful way to drive content selection upon global criteria but it keeps open the problem of content organization. Moreover, only specific applications have been yet handled by such an approach.

These two approaches use a two steps decision process but in the opposite order. Whereas the content is selected first while using optimization constraints and then an organization is set on the selected contents, approaches based on transformation sheets select first an organization and then the content that fits well with the selected organization. Concerning this aspect knowledge based applications use an interesting alternative solution since they break each decision into several entities and thus are able to adapt the decision order to local situations.

Another thing that we have learnt through this study is that few works propose generic solutions able to help anyone that wants to design and maintain an automatic generation application. This is why it is important for us to propose both a methodology and a software architecture that are not tuned for a particular application. The architecture we propose combines the advantages of the use of transformation sheets and optimization tools described above and is also an attempt to break the granularity of the decision steps (we have seen for instance in the "knowledge approach").

4 Architecture Overview

We only give here an overview of the architecture. The details of each component will be given in the next section through the description of the personalized sport news example.

The main principle of our architecture is that a transformation sheet is used first to build a coarse structure of the final document and then an optimization step is used to refine this structure. In these two steps, objects are selected and some decisions about objects organization are taken. The idea is that the transformation step selects more objects than the wished final set while leaving the optimization step to remove objects from this first selected set. Concerning the organization of the presentation the principle is that the transformation sheet gives a global structure of the document while keeping unspecified some formatting values (objects duration, objects size…). This is one goal of the optimization process to fix these formatting values.

The proposed architecture is described in Figure 1. An application built on this architecture takes as input an XML database, a set of parameter values, several transformation sheets and sets of constraints. The application generates a formatted document through three different components. The first component (named **Analyzer**) aims to take into account the values given by the user for each input parameter in order to select:

- one transformation sheet TSi ;
- the constraints set CSjk associated to the each parameter value Vjk (value Vj of Parameter Pk).

Then, the selected transformation sheet is used by the second component (named **Transformer**) in order to apply it on the XML sources. Finally, the third component (named **Optimizer**) takes into account the constraints sets selected by the first component plus the set of global constraints, in order to decide which objects from the intermediate document must be removed and which exact values must be associated with open formatting parameters.

The developer of a generation application that wants to use our architecture has to define:

- The input parameters that control the application (P1, …, Pp), including both the user choices and the parameters coming from the environment (screen size for instance);
- Several transformation sheets that are associated with the values of the input parameters (TS1, …, TSt). For instance, the developer can write different transformation sheets for each kind of target devices. These transformations have to match the XML structure given as input of the application.
- Several sets of constraints that may depend on each value of the input parameters. Suppose for instance, that the user can choose to see a multimedia document without any audio, then a constraint that specified that the duration of each audio is equal to zero must be associated to the value “without” of the

parameter untitled “with or without sound”. It means that this constraint will be selected by the Analyzer component if and only if the “without” value is selected by the user.

- A set of global constraints that does not depend on the input parameters value. These constraints express some information about the application domain. For instance, we can suppose that in a sport news application, showing the summary of the match is always more important than showing some interviews.

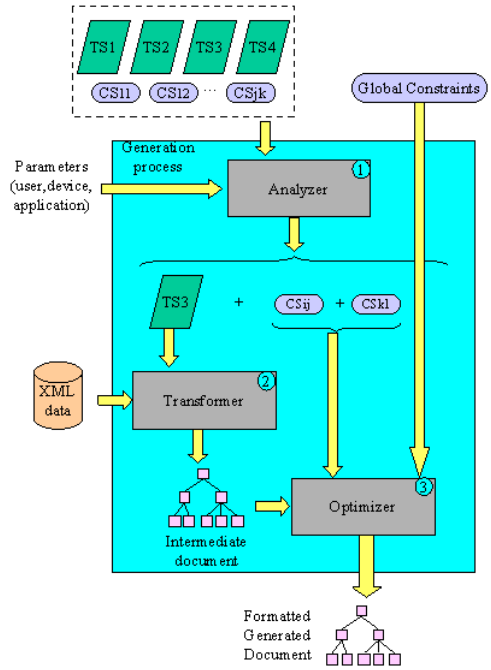

Figure 1. Architecture for automated construction of multimedia presentation

The main feature of this architecture is the use of a constraint solver both to globally optimize the content selection and to perform some parts of the formatting phase in order to handle continuous parameters like global duration or screen size. Moreover we choose to select the transformation sheet before selecting the content in order to constrain this selection with some information deduced from the transformation sheet.

5 The architecture in use

We present in this section the architecture in details by describing how we develop the personalized sport TV-news example by using this architecture. The language target for the generated presentation is the SMIL 1.0 language. This choice has been done because of the need of the temporal dimension. But SMIL 1.0 is one of the simplest language to describe multimedia documents. Even if it can describe a quite complex temporal organization, it does not provide any operators for the spatial dimension. This point is important for us to evaluate the power of our constraint language.

5.1 The personalized sport TV-news example

The application presented to illustrate the use of the architecture is aimed to generate a personalized TV-like sport news report about the results of a football championship day. The user can express her/his favorite teams and can choose to see either a report that focuses on these favorite teams or an exhaustive report of the day. The user can also set a constraint on the total duration of the report and decide to have or not have any audio in the document. Finally, the application is able to take into account the kind of device used by the user (large screen, pocket PC…) and thus to produce a multimedia document adapted to the resources of this device.

a) b)

Figure 2. Snapshot of a long summary a) and a short summary b)

Whatever the user choices are, the designer of this application decides to build a document that has the same temporal structure: it basically presents sequentially several matches by using either a long summary of the match or only a short summary depending for instance on the total duration of the document. These two

kinds of summaries do not use the same spatial organization (Figure 2). After this sequence some global information about the championship day (results and ranking) are presented. In fact the user choices modify first which matches are presented in the main sequence and in which order then how long is the time associated with each summary and finally which media types (video, sound or only text and picture) used.

5.2 The Analyzer component

This component analyses the input parameters in order to select the transformation sheet that will be used by the Transformer component. It also builds the constraints set linked up to these values and that will be merged with the global constraints in order to be used by the Optimizer component.

In our example, the user of the application has to set four different values:

1. The duration of the report. She/He has to choose a value between 5mn and 60mn.
2. Her/His favorites team.
3. The kind of report she/he wants to see: an exhaustive report or a focused report on her/his favorite teams.
4. A report with audio object or a report without any audio.

Moreover, one environment feature is taken into account by the application: the screen size (only small or large screen values).

(Parameter, value)	**Constraints (not formally expressed)**
(P1, d)	- total duration equals d
(P2, t1,t2,…,tn)	- use teams t1,t2,… , tn as favorites teams
(P3,exhaustive)	- each match is presented during approximately the same amount of time.
(P3,focused)	- priority for favorite teams
(P4,without)	- duration of audio objects must be null - duration of objects which type is interview must be null
(P5, small)	- no scroll bar - use time sequencing for large text presentation

Figure 3. Constraints associated to each parameter.

Once these five values have been specified, the component selects a transformation sheet TSi. Its choice is simply based on the description made by the developer of which set of parameter values corresponds to which transformation sheet. This description is contained in the header of the transformation sheet. In our example, the developer decides to write four transformation sheets that depend on both the first user parameter (kind of report) and the screen size value. It means that

a first transformation sheet handles the case of an exhaustive report shown on a large screen, a second one handles the case of an exhaustive report on a small screen and so on. The second goal of the Analyzer component is to select the set of constraints associated with the effective parameter values. We give in Figure 3, the intuitive meaning of each constraint associated with their parameter values. The exact language used to express them will be presented while describing the Optimizer component.

5.3 The Transformer component

Once the transformation sheet has been chosen, the Transformer component applies the transformation to the XML data. The transformation aims to build the coarse-grained structure of the final document: selecting the maximal set of objects that can be useful inside the final document and deciding the global objects organization in each dimension (for instance, to decide that the document presents in sequence a set of subparts, each of them being a parallel composition of some basic objects). Indeed, to adapt the document in a finer way, we let the Optimizer component refining the document resulting from the transformation: by removing some objects and specifying some formatting values (like objects duration and objects size) that the transformation process has not yet specified.

```
<team name="OM" logo="om.gif">
  <picture>OMteam.jpg</picture>
  <picture>VelodromeStadium.jpg</picture>
  <player name="bakayoko">bakayoko.gif</player>
  <player name="maurice">maurice.gif</player>
  <player name="leroy">leroy.gif</player>
  <player name="belmadi">belmadi.gif</player>
  <player name="skoro">skoro.jpg</player>
  <player name="gallas">gallas.gif</player>
</team>

<match numero="1" home="OM" visitor="AS Monaco">
  <score>2-0</score>
  <goal team="home">
    <minute>75</minute>
    <striker>bakayoko</striker>
  </goal>
  <goal team="home">
    <minute>85</minute>
    <striker>bakayoko</striker>
  </goal>
  <summary type="short">OM_ASMshortvideo.mpg</summary>
  <summary type="long">OM_ASMlongvideo.mpg</summary>
  <interview team="home">OMcoatch.mpg</interview>
  <interview team="home">OMpresident.mpg</interview>
  <selection team="home">
         <titular player="bakayoko"/>
    <titular player="maurice"/>
        <titular player="leroy"/>
        <titular player="belmadi"/>
        <titular player="skoro"/>
        <titular player="gallas"/>
        <titular player="camara"/>
    </selection>
    <selection team="visitor">
        <titular player="simone"/>
        <titular player="marquez"/>
        <titular player="giuly"/>
        <titular player="nonda"/>
        <titular player="leonard"/>
        <titular player="porato"/>
  </selection>
</match>
```

Figure 4. An example of the XML sources (team and match).

An example of XML sources are shown in Figure 4. The transformation result shown in Figure 5 corresponds to the case of an exhaustive report presented on a large screen. It means that the user is mainly interested in some favorite teams but want to see all the match summaries. So for each match that does not involve a favorite team, the transformation produces only a short summary (it should last less than 5mn) whereas for each match involving one favorite team, the transformation produces both a short summary and a long summary that contains all information included in the XML data. These summaries are always built for large screen only.

In the "match" element of Figure 4, we can see that there is two "summary" elements. Both will be transformed in the target language (elements with id equals to "summary1" and "summary2" in Figure 5).

```
<seq id="summary">
    <par id="match1">
      <text src="html/OM.html"/>
      <text src="html/1-0.html"/>
      <text src="html/AS_Monaco.html"/>
      <img src="images/om.gif"/>
      <img src="images/monaco.gif"/>

      <par>

        <!— long summary —>
        <seq id="longsummary">
          <par endsync="first">
            <audio src="sons/entree.wav"/>
            <seq>
              <img src="images/OMteam.jpg"/>
              <img src="images/bakayoko.gif"/>
              ...
              <img src="images/gallas.gif"/>
              <text src="html/Homeselection.html"/>
              <img src="images/teamMonaco.jpg"/>
              ...
              <img src="images/porato.gif"/>
              <text src="html/selectionVisiteuse.html"/>
            </seq>
          </par>
          <video id="summary1" src="videos/OM_ASMlongvideo.mpg"/>
          <video id="interview1" src="videos/OMcoatch.mpg"/>
          <video id="interview2" src="videos/OMpresident.mpg"/>
        </seq>

        <!— short summary —>
        <par id="shortsummary">
           <text src="html/selectionH.html"/>
           <text src="html/selectionV.html"/>
           <seq>
             <video id="summary2" src="videos/OM_ASMshortvideo.mpg"/>
             <video id="interview3" src="videos/OMcoatch.mpg"/>
             <video id="interview4" src="videos/OMpresident.mpg"/>
           </seq>
        </par>
      </par>
    </par>

    <par id="match2">
    ...
    </par>

    ...
    <text dur="5s" src="html/feuille1.html"/>
</seq>
```

Figure 5. Intermediate SMIL document, result of the transformation process.

The final choice between the short and long summaries is under the responsibility of the Optimizer component. In our example, only one of the two

summary elements will be display in the final presentation (the element with id equals to "summary1" in Figure 8). Thus, the developer has first to express this alternative using the constructions proposed by the target language and then to express some constraints that will be used by the Optimizer to select only one object contained inside the alternative. As we developed our example using SMIL as target language of the transformation, we used a parallel construction between the long and the short summary and a constraint (one of the global constraints in Figure 7) that expresses the mutual exclusion between these two kinds of summaries. Note that we can't use here the "switch" element of the SMIL 1.0 language. Indeed, the test made on the elements it contains to display them in exclusion manner can only be computed on simple discrete parameters (like language, bitrate or screen depth) whereas we would need more global criteria or complex functions. Here, the choice is given by the computed duration that best fit the given parameter.

The global temporal structure given by the transformation process is a sequential presentation of each match that begins with the matches involving at least one favorite team (Figure 6).

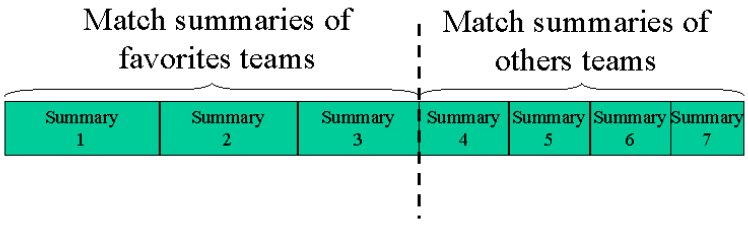

Figure 6. Temporal organization of the generated presentation.

5.4 The Optimizer component

The Optimizer component works on the document resulting from the transformation process and builds the final multimedia document. Its first goal is to decide which objects have to be removed and its second goal is to set a value to each unspecified formatting values. In our example, the main selection will be the choice between the long summary and the short summary for each match that proposes both. The formatting and content selection decisions are controlled both by the set of global constraints and by the ones selected by the Analyzer component.

Thus the role of the developer according to this component is to express the constraints that will guide the solver in the formatting and the content selection phases. From intuitive ideas like "each match is presented during approximately the same duration", she/he has to express real constraints. To do so, the architecture needs to provide her/him with a language. This one must be able to express some classical needs when dealing with formatting issues (for instance, instructions to format a table) like done in XSLFO [24] or CSS [22] languages. Of course, it must

also integrate the multimedia dimension of the document (by proposing for instance instructions to distribute a global duration through a temporal sequence) like it is done in Madeus [9].

But it has also to express different strategies for achieving these formatting instructions. For instance, the developer needs to express that a global duration must be distributed through the objects of a temporal sequence by allowing the formatter either to remove objects with some level of priority or to remove objects beginning by the end of the sequence. In order to perform some experiments with our architecture, we have currently defined a first basic language that should be extended in the future to be more expressive. We present an overview of this language by presenting its three main features and showing how some constraints expressed in Figure 3 and some global constraints useful for our working example are expressed. Finally, the constraints expressed by the developer of our working example are summarized in Figure 7.

(Parameter, value)	Constraints (not formally expressed)	Explicit Constraints
(P1, d)	- total duration equals d	- duration(element[id="Summary"]) = d
(P2, t1,t2,…,tn)	- use teams t1,t2, …, tn as favorites teams	- priority((element[id="team_t1"],1),(element[id= "team_t2"],2)… (element[id="team_tn"],n))
(P3,exhaustive)	- each match is presented during approximately the same amount of time.	- time_well_balanced(element[id="Summary"])
(P3,focused)	- priority for favorite teams	- remove_last_element_first(element[id="Summary"])
(P4,without)	- duration of audio objects must be null - duration of objects which type is interview must be null	- duration(element[media="audio"])=0 - duration(element[type="interview"])=0
(P5, small)	- no scroll bar - use time sequencing for large text presentation	- scrollbar="no" - cut(element[media="text"]) =time_sequence

Global Constraints
- mutual_exclusion(element[id="LongSummary"], element[id="ShortSummary"])
- priority((element[id="MatchSummary"],1),(element[id="Interview"],2))
- remove_low_priority_first(element[id="Summary"]) - remove_low_priority_first(element[id="Match"])

Figure 7: Constraints needed for the example

Addressing

One feature of our architecture is that the constraints are separated from the document on which they apply. So the first need for this language is to be able to

address some parts or elements of the documents. For instance in the constraint duration(element[media="audio"])=0 associated with the constraint "duration of audio object must be null" addresses all audio objects of the document.

Formatting strategies

One key point of this language is to provide the developer with some possibilities to control the formatting phase performed by the Optimizer. For instance, the constraint "each match is presented during approximately the same amount of time" is explicitly expressed by the constraint time_well_balanced(element[id="Summary"]) where Summary is a sequential composition produces by the transformation sheet. The time_well_balanced instruction asks the solver to give children of this sequence approximately the same duration (as long as possible).

Removing strategies

The developer has also to guide the Optimizer when it fails to fulfill a constraint and thus has to remove some objects. For instance, while failing to format a temporal sequence with a fix duration, there are a lot of possibilities to reduce the set of objects: removing last objects first, removing objects with low priority, etc. Thus, the language has to enable the developer to express these strategies and also some priority levels. To do so, she/he can use the following instructions : priority(element(),1)(element(),2))…(element(),n)) to set priorities and remove_low_priority_first(element()) or remove_last_element_first(element()) to decide how to remove objects into a temporal sequence. Another thing that the developer should be able to express is some kind of mutual exclusion in order to solve alternatives between elements that are still present after the transformation step. To do so, we provide her/him with a mutual_exclusion(element(), element()) instruction that can be used in our example to express mutual exclusion between short summaries and long summaries.

It is important to understand that all formatting instructions given by the developer have not to be satisfied by the solver, but this one has to find a solution that is the best compromise between these (sometime contradictory) orders. To do so, the Optimizer needs to know which level of priority must be associated with each language instructions. Even if it is possible to let the developer of the application to set these levels of priority, we consider that it is a too difficult task and we prefer to associate a default priority level with each instruction.

Given these constraints, expressed by the developer of the application, the first thing done by the Optimizer is to translate both the semantic of the document

produced by the transformation and the high level constraints given by the developer into constraints understandable by the solver. This sub-step is handled by the Translator component that takes place before the Solver inside the Optimizer. The semantic interpretation corresponds to the translation of the each object into variables and each relation into constraints between those variables following the semantic of the target language (in our application: SMIL). The second translation, that handles the high-level constraints, strongly depends on the temporal and spatial organization of the document resulting from the transformation process. For instance, the constraint duration(element[id="Summary"])=d is translated into the sum of all objects included in the document if and only if these objects are organized in sequence. Given the Transformer tasks presented above, it is easy to understand that the Translator component is specific to a language used to describe multimedia document.

```
<seq dur="900s" id="summary">
    <par  dur="182s" id="match1">
      <text dur="182s" src="html/OM.html"/>
      <text dur="182s" src="html/1-0.html"/>
      <text dur="182s" src="html/AS_Monaco.html"/>
      <img dur="182s" src="images/om.gif"/>
      <img dur="182s" src="images/monaco.gif"/>

      <par dur="182s">

        <!— long summary —>
        <seq dur="182s" id="longsummary">
          <par dur="104s" endsync="first">
            <audio  dur="300s" src="sons/entree.wav"/>
            <seq dur="104">
              <img dur="2s" src="images/OMteam.jpg"/>
              <img dur="2s" src="images/bakayoko.gif"/>
              ...
              <img dur="2s" src="images/gallas.gif"/>
              <text dur="4s" src="html/Homeselection.html"/>
              <img dur="2s" src="images/teamMonaco.jpg"/>
              ...
              <img dur="2s" src="images/porato.gif"/>
              <text dur="4s" src="html/selectionVisiteuse.html"/>
            </seq>
          </par>
          <video dur="53s" id="summary1" src="videos/OM_ASMlongvideo.mpg"/>
          <video dur="25s" id="interview1" src="videos/OMcoatch.mpg"/>
        </seq>
      </par>
    </par>

    <par dur="170s" id="match2">
    ...
    </par>

    <text dur="5s" src="html/feuille1.html"/>
</seq>
```

Figure 8. Result of the optimization on the previous document

Once this translation phase has been done, the Solver computes a solution: it sets a value to each variable (the best compromise between the set of constraints). Theses values are then integrated into the document as object attributes. This is the last step of our architecture and in our example it gives the SMIL document in the Figure 8. In this document, all the duration attributes of the elements have been computed. For instance, the duration of the global sequence is formatted to 900 seconds. The choice between the short and the long summary has been fixed in

favor of the long summary and we can see in it that the video of the second interview has been suppressed.

Before concluding this paper, we briefly present the technical state of the architecture we propose. Its implementation is based upon the use of the Kaomi toolkit [11]. This one aims to ease the design, the transformation and the presentation of multimedia documents. Kaomi integrates a XSLT [25] Xalan [27] processor that we use to develop both the Analyzer and the Transformer. Of course, the core element of our architecture is the Optimizer. It is also the most complex one. Kaomi contains a SMIL parser that we use to build the Translator sub-component. Moreover, it provides the developer with some basic formatting supports that are based on the use of the Cassowary constraint solvers [2], a time-efficient solver for linear constraints. We currently experiment its use to perform some kind of optimizations problems that we encounter in automatic generation applications. For instance, how to format a temporal sequence while following some removing strategies. In parallel, we are connecting the Scilab [6] scientific software package for numerical computations into the Kaomi toolkit in order to express more general constraints and to take advantages of Scilab optimization support.

6 Conclusion

After a detailed analysis of existing works on automatic generation of multimedia documents that highlights the absence of general proposition to handle generation applications that deal with global criteria, we propose both a methodology and a software architecture to help the development of such kind of application. This architecture takes benefits from both the use of transformation sheets and the expressive power of constraint solvers to easily and efficiently associate different rendering to the same set of objects and to handle global criteria during the generation process. Taking both content and organization decisions allows to build multimedia documents having nice rendering (like with template approaches) while choosing the best content to fit global criteria. Moreover, thanks to the use of constraints some properties that would be tedious to express by using only a transformation sheet, are easily expressed in terms of constraints. For instance, the “with-or-without audio" parameter can be handled by using some ‘switch’ rules in each transformation sheet, but it is much easier to add the constraint: duration(element[media="audio"])=0 if necessary. Beyond the implementation tasks described above, our main goal is to extend the language used to describe high-level constraints in order to cover more generally the needs of documents generation applications.

7 Bibliography

1. André E. and Rist T., "Generating Coherent Presentations Employing Textual and Visual Material", *AI Review* **9** (1995) pp. 147-165.
2. Badros G. J. and Borning A., "The Cassowary Linear Arithmetic Constraint Solving Algorithm: Interface and Implementation", *Technical Report UW-CSE-98-06-04* (1998).
3. Boll S. and Klas W., "ZYX - A Multimedia Document Model for Reuse and Adaptation", In Transactions on Knowledge and Data Engineering, DS-8 Special Issue, IEEE (2000).
4. Bordegoni M., Faconti G., Feiner S., Maybury M., Rist T., Ruggieri S., Trahanias P. and Wilson M., "A Standard Reference Model for Intelligent Multimedia Presentation System", Computer Standards and Interfaces, Special Issue on Intelligent Multimedia Presentation Systems, **18**(6-7) (1997) pp. 477-496.
5. Dalal M. *et al*., "Negotiation for Automated Generation of Temporal Multimedia Presentations. ", *Proceedings of ACM Multimedia'96*, ACM Press (1996) pp. 55-64.
6. Gomez C., "Engineering and Scientific Computing with Scilab" (INRIA - Rocquencourt Ed., France, 1999) 512 pages; *http://www-rocq.inria.fr/scilab*, May 2001.
7. Hakkoymaz V., Kraft J. and Ozsoyoglu G., "Constraint-Based Automation of Multimedia Presentation Assembly", *Proceedings of ACM Multimedia'99*, ACM Press (Orlando, USA, 1999).
8. Henze N. and Nejdl W., "Adaptivity in the KBS Hyperbook System", *2nd Workshop on Adaptive Systems and User Modeling on the WWW*, (Toronto, 1999).
9. Jourdan M., Layaïda N., Roisin C., Sabry-Ismail L. and Tardif L., "Madeus, an Authoring Environment for Interactive Multimedia Documents", *ACM Multimedia'98*, ACM (Bristol, UK, 1998) , pp. 267-272.
10. Jourdan M., Roisin C., Tardif L. and Villard L., "Authoring SMIL documents by direct manipulations during presentation", *World Wide Web*, Balzer Science Publishers, **2** (4) (1999).
11. Jourdan M., C. Roisin and L. Tardif, "A Scalable Toolkit for Designing Multimedia Authoring Environments", *Multimedia Tools and Applications Kluwer Academic Publishers*, vol. **12**, num. 2/3 (2000), pp. 257-279.
12. JSP, *http://java.sun.com/products/jsp*, May 2001.
13. Merialdo B., Lee K.T., Luparello D., Roudaire J., "Automatic Construction of Personalized TV News Programs", *Proceedings of ACM Multimedia'99*, ACM Press (Orlando, USA, 1999).
14. Microsoft ASP, *http://www.microsoft.com*, May 2001.
15. Pachet F., Roy P. and Cazaly D., "A Combinatorial approach to content-based music selection", *IEEE Multimedia* (2000).

16. Paradis F. and Vercoustre A.-M., "A Language for Publishing Virtual Documents on the Web", *in International Workshop on the Web and Databases* (Valencia, Spain, 1998).
17. PHP, *http://www.php.net*, May 2001.
18. Rutledge L., Hardman L., van Ossenbruggen J. and Bulterman D.C.A., "Adaptable Hypermedia with Web Standards and Tools", *The Active Web -- A British HCI Group Day Conference* (London, 1999).
19. Rutledge L., Davis J., van Ossenbruggen J. and Hardman L, "Inter-dimensional Hypermedia Communicative Devices for Rhetorical Structure", *In: Proceedings of International Conference on Multimedia Modeling 2000 (MMM00)* (Nagano, Japan, 2000) pp. 89-105.
20. Sarwar B, Karypis G., Konstan J. and Riedl J., "Item-Based Collaborative Filtering Recommendation Algorithms", *The Tenth International World Wide Web Conference (WWW10)* (Hong-Kong, 2001).
21. Villard L., Roisin C. and Layaïda N., "A XML-based multimedia document processing model for content adaptation", *Digital Documents and Electronic Publishing (DDEP00)* (2000).
22. W3C, CSS2 Recommendation (1998); *http://www.w3.org/TR/REC-CSS2*, May 2001.
23. W3C, SMIL Recommendation (1998); *http://www.w3.org/TR/REC-smil*, May 2001.
24. W3C, XSL Candidate Recommendation (2000); *http://www.w3.org/TR/xsl*, May 2001.
25. W3C, XSLT Recommendation (1999); *http://www.w3.org/TR/xslt.html*, May 2001.
26. Weitzman L. and Wittenburg K., "Automatic Presentation of Multimedia Documents Using Relational Grammars", *Proceedings of 2nd ACM Conference on Multimedia*, ACM Press (San Francisco, California, 1994) pp. 443-451.
27. Xalan, *http://xml.apache.org*, May2001.