Genre Driven Multimedia Document Production by Means of Incremental Transformation

Marc Nanard, Jocelyne Nanard, Peter R. King, Ludovic Gaillard

Genre Driven Multimedia Document Production by means of Incremental Transformation

Marc Nanard
LIRMM, Univ. Montpellier/CNRS
161 rue Ada
34392 Montpellier, F
(33) 467 41 85 17
mnanard@lirmm.fr

Jocelyne Nanard
LIRMM, Univ. Montpellier/CNRS
161 rue Ada
34392 Montpellier, F
(33) 467 41 85 16
jnanard@lirmm.fr

Peter R. King
Department of Computer Science
University of Manitoba
Winnipeg MB, R3T 2N2, CA
(1) 204 474 9935
prking@cs.UManitoba.ca

Ludovic Gaillard
INA, Direction de la Recherche et de l'Expérimentation,
4 avenue de l'Europe,
94366 Bry-sur-Marne, F
(33) 149 83 21 33
lgaillard@ina.fr

ABSTRACT

Genre, like layout, is an important factor in effective communication, and automated tools which assist in genre compliance are thus of considerable value. Genres are reusable meta-structures, which exist independently of specific documents. This paper focuses on that part of the document production process which involves genre, and discusses a specific example in order to present the design rationale of mechanisms which assist in producing documents compliant with specific genre rules.

The mechanisms we have developed are based on automated incremental, iterative transformations, which convert a draft document elaborated by the author into a genre compliant final document. The approach mimics the manner in which a human expert would transform the document. Transformation rules constitute a reusable and constructive expression of certain aspects of genre. The rules identify situations which appear inappropriate for the genre in question, and propose corrective action, so that the document becomes increasingly more compliant with the genre in question. This process of genre conformance iterates until no further corrective action is possible.

This mechanism has been fully implemented. The implementation comprises both a work environment and a rule based language. The implementation relies internally on a general purpose tree transformation engine designed originally for use in natural language processing applications, which we have adapted to handle XML documents.

Categories and Subject Descriptors

I.7 [Document and Text Processing]: Document Preparation – Languages and Systems; H.5.4 [Hypertext/Hypermedia]:

General Terms

Documentation, Design, Human Factors, Languages.

Keywords

Genre, multimedia, transformation, meta-structure, series.

1. INTRODUCTION

In this paper we present an approach to the design and production of multimedia documents based on the notion of *genre* as a re-usable document *meta-structure.* The term meta-structure denotes complex structural, narrative, rhetorical or content based properties, which are shared among a set of documents but which cannot be described in terms of logical structure alone. As an informal example of this notion, we may observe the sharing of meta-structures within familiar multimedia document series, such as television or video series. The darkly comedic look of "perfect American suburbia" in episodes of *Desperate Housewives*, the flat shape assumed by Tom upon crashing into a wall in *Tom and Jerry*, or the sincere voice overtones in *Panorama* are typical design patterns common to documents of the particular series. Documents belonging to a given series share a number of characteristic properties, which serve to define the series. It is apparent however, that no typical logical structure can properly capture such properties and distinguish between series; thus such a series cannot be satisfactorily considered as a document class.

The concept of document genre goes far beyond this notion of document series. Genre characterises one major reusable document meta-structure and encompasses several diverse aspects including narrative constructs, structural and temporal constraints, and rhetorical and visual effects. The concept has been studied by many authors, from antique times with such notions as tragedy, comedy, poetry, speeches, narratives, up to the present day, where we observe such differing TV genres as news, documentary, or sit-com. In our view, the production of documents complying with a given genre currently represents a major challenge for the multimedia document industry. Furthermore, adaptation is an important facet of such work. The contextual adaptation of generic documents to users, such as producing on-the-fly user-adapted news-on-demand [Iksal 2002], consists in producing new documents which have different contents but which are still recognized as being of the same genre.

In this paper we present the design rationale of techniques, originating in the Saphir project[1], which assist in automatically organizing multimedia data extracted from archives in order to produce documents complying with specific genre rules. Such genre rules can be reused for producing document series. We illustrate the approach by an example, describing the production of synthetic TV news digests from multimedia archives.

[1]See: www.riam.org/riam/upload/posters/PosterRIAM2006_SAPHIR.pdf

The remainder of this paper is organized as follows. In section 2 we elaborate on the notion of genre within multimedia documents and discuss the problems involved in producing genre compliant documents. In section 3 we outline the approach taken in this work. Section 4 contains the technical details of our approach, discusses the transformation techniques and the system we use, and discusses in detail their application to the news digest example. Section 5 discusses our work in the broader context of document transformation, and concludes.

2. THE PROBLEM

2.1 Genre

Genre, like layout, is an important factor in communication. Consider layout. The importance of document layout for effective communication is universally recognized. Great care in the fine design of the layout of a web page is now the norm among document producers, professional or otherwise. The distinction between document contents, document structure and document layout is well accepted, and sound techniques and associated tools make it possible to describe a document and its layout separately, and to use presentation rules to map layout onto data.

The genre of a document plays an equally important role in communication [Fowler 1989]. Genre involves a form of categorization on the part of the reader when she perceives a document [Chandler 1997]. When one turns on the TV, one quickly recognizes whether one is watching TV news, a documentary, a thriller, a commercial, a debate etc. Genre is distinct from layout, from appearance, from content and from structure. Consistency is a major requirement for humans in any domain, and genre consistency is a key factor in document understanding. Just as inconsistencies in the layout of a document will perturb the reader, so too will inconsistent respect of the rules of a genre.

A considerable volume of work is focussed on characterizing the notion of genre. Recent work is concerned with document categorization according to genre [Crowston 1997]. Automatic or assisted production of documents which consistently respect the rules defining a given genre is emerging as a topic of considerable research interest, just as the question of eliciting general layout concepts did in the early seventies. Unfortunately, most of the work in multimedia production considers specific genres, such as biographies [Geurst 2003] or virtual dialog [Bocconi 2005]. The first work to emphasize the generic notion of genre as an approach to multimedia production is [Falkovych 2005] where it is argued that production should take account of genre semantics, and that one should adopt a genre model inspired from [Duff 2000] in which genre is defined as a combination of *content*, *form* and *function*. We return to this point later.

In this paper we address the problem of genre driven multimedia document production. Genre *recognition*, that is assigning a given document to a genre, is a complex problem widely discussed in literature. For example, in [Kwasnik 2001] it is shown how web search effectiveness can be improved by identifying document genre. However, recognition is outside our scope. We rather, take the position that examination of sets of documents identified by humans as belonging to a given genre will enable the identification of the shared meta-structures belonging to the genre in question, so that in turn new documents exhibiting these same meta-structures, and thereby being recognized as belonging to the same genre, may be produced. We do not claim that our approach permits the characterization and specification of any genre in general, but rather that the approach can assist in the production of multimedia documents which comply with some strongly typed multimedia genre.

More information about the general notion of genre may be found in [Chandler 1997].

2.2 An Example

Throughout this paper we will base our discussion on a single example[2], namely the synthesis of a *TV news digest*, by reusing richly indexed segments available in the audiovisual archives. TV news is a well-known strongly typed genre. When presenting a news story on an on-going topic, it may be desirable to present first a *digest*, a brief synthetic summary of the most relevant points of the topic to date. Such a digest is far from being simply a concatenation of cuts from old news. A digest needs to resemble a news segment. Such flash-backs are typically constructed from multimedia news archives, but they are completely re-designed to call attention to salient events of the story.

The example has been fully implemented on a real scale instance known as the *De Villepin CPE Digest.* The corpus associated with this example is composed of morning, midday and evening bulletins from French TV news (TF1, France 2 and M6) over a three day period. The sample period is deliberately narrow in order to maintain homogeneity in the set of themes. The main subject during this period concerned the attempted political reformation known as the *Contrat Première Embauche* (CPE). We extracted a set of seventeen news reports relating to this subject, which serve as the base for the digest. Our automated system elaborates various digests to present the context of this event according to the TV news genre. We describe and discuss the production process later in this paper in sections 3 and 4.

2.2.1 The Natural Authoring Process

Let us first consider how an author would produce a genre compliant document without automated help. Her process iterates on the following three tasks :

- delimiting the thematic scope, either by exploring a thematic ontology, or by querying the archives;
- gathering relevant pieces of information related to the thematic scope.
- organizing the discourse intentional structure, narration and argumentation according to the author's thesis.

A preliminary draft document emerges, having the form of a graph of thematic intentions linked by discursive relations, with references to collection of equivalent pieces of information, such as archive small segments, which could be used as support for each of the elicited intentions. At this point the author starts taking into account genre constraints to plan the actual discourse by choosing and organizing the elements which will constitute the final document.

It is important to note that such a design process cannot be straightforward, but, as discussed by Hayes and Flower concerning the process of writing [Hayes 1981], is an iterative, incremental loop based on a *construct*, *examine* and *improve* cycle. Some aspects of the genre are intrinsic to the narrative and

[2] From a real scale example in SAPHIR project at INA.

argumentative structures, others depend on the video montage. In the paper, we focus on the last point.

2.2.2 Genre-Related Constraint Classes

We now analyse the classes of constraints which the author needs to take into account when producing a genre compliant multimedia document. To simplify the exposition we focus on a single point of this genre, which rules should follow a *topic introduction* within TV news; constraint classes found more generally are similar.

Structural constraints. A major characteristic of the TV news genre is that any new subject is *introduced* by the *TV news anchor*[3]. The video channel and the audio channel are used in synchrony to start the announcement of the topic for a few seconds, and then the video channel presents several brief suggestive sequences while the audio proceeds with the presentation. Once the introduction is over, the development may take one of several forms, including small reportages, each being introduced. However, the rule for introducing sub-topics differs slightly as shown in figure 1. While the sound channel is used to introduce the subtopic, typically by a journalist rather than by the news anchor, the video channel presents, as an overview, a sequence of short samples related to the subtopic. Optionally a textual dubbing announces the subtopic.

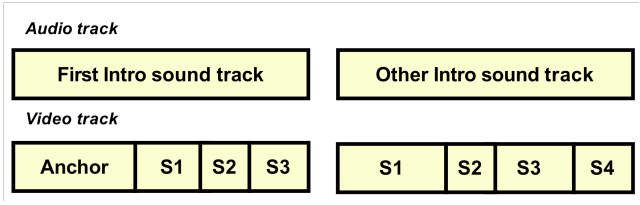
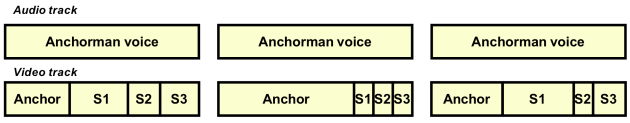


Figure 1: Distinct introduction structures.

Temporal constraints. The duration of an introduction, the ratio between the time the presenter is displayed and the duration of the visual samples in the main introduction as well as in introduction of subtopics have strong relationships and bounds, as shown in figure 2. Suggestive samples must be neither too long nor too short, the amount of time during which the presenter is visible is usually between 30% and 45% of the whole introduction time, the remainder being samples. The duration of samples must be well balanced. Samples must be organized jointly in terms of rhetoric, visual properties and duration.



(a) Proper timing, (b) Samples too short, (c) Improper balance.

Figure 2: Timings in an introduction.

Rhetorical and visual effects. As a general rule, it is important to maintain visual diversity in the design of a video sequence. For example, it is not suitable to concatenate two close-up sequences of the same individual, especially from opposite sides; alternating close-up and wide-angle views better keeps the attention of the viewer. Similarly, for rhetorical reasons, consistency between an image and its semantic use is desirable. An introduction would not start with a picture of an individual leaving, especially if filmed from the rear, unless this has a strong contextual meaning (such as a minister being dismissed.) One would rather choose an image of someone facing the camera or moving towards it; an image depicting departure would be suitable within a conclusion. General rules, which concern other genres in addition to the TV news genre, are responsible for the selection of the segments actually used from among the collections of semantically equivalent archive pieces.

Flexibility and Alternatives. Genre compliance is typically a problem of fuzzy optimisation. For a given genre, several alternatives, sometimes quite different in terms of structure, may exist. No constraint can be strict. Rather, a genre appears as a fuzzy set of suggestions of how to make the document better. Typically, a genre expert makes statements of the form "option A is usually better, but if A is not possible, then B would be sufficient". Thus backtracking may be needed to try alternative strategies when the most common one fails. Flexibility is necessary to permit a document to emerge, even when no solution satisfies all the rules. Therefore rigid, model driven approaches are inadequate to address genre conformance issues.

Other rules. The actual rules which characterise the genre TV news in its full generality are far more complex and numerous. For reasons of clarity, we have chosen to focus on these few, related to *introduction* in this example, to exhibit the problems and describe the proposed solution.

2.2.3 Towards Genre Driven Multimedia Production

Human expertise is required to address genre driven issues, since it appears to be extremely difficult to describe these kinds of constraints in a generic manner. Currently, only a few *ad hoc* solutions are in use to help genre compliance, most often hardwired either in PHP or Java programs or with tricky XSLT code. Tackling the complexity and the generality of genre compliance requires other specification paradigms.

Consider what aid an author would want in order to produce a genre compliant document. She would expect that genre adaptation would be handled in an analogous fashion to layout. Since layout is automatically computed from the author's expression of a logical document by applying a style-sheet driven transformation, genre adaptation should result from the transformation of an intended document into a genre compliant document.

In a computer-aided environment, producing a genre compliant document typically involves a two stage iterative process. An author creates and expresses her intended document. This source document itself includes some genre aspects, related to narration, argumentation and so on, but does not take much account, if any, of other genre specific constraints. An automated process then transforms the document to produce the genre compliant document according to a specification expressed in terms of transformation rules. The author takes advantage of a feedback loop in which she observes the document produced and improves upon it, but does so by editing the intended document rather than the target.

The next sections present and discuss an original approach we have implemented to assist in this form of genre driven multimedia production.

[3] See: http://museum.tv/archives/etv/A/htmlA/anchor/anchor.htm

3. SOLUTION DESIGN RATIONALE

Despite the considerable volume of published work on genre, this notion still lacks a usable characterization as an independent entity. Furthermore genre is an evolving notion, insofar as new genres constantly appear, particularly involving new media. Recent trends in genre theory [Duff 2000] suggest, rather, a constructive approach whereby a genre is characterized as a *function* mapping a *form* onto *content*. Such a constructive approach appears particularly appropriate in the domain of document production, and is explicitly used by [Falkovych & Nack 2005] to emphasize the generic notion of document genre as an approach to multimedia document production. Our work follows this trend, and expresses the Duff function in terms of document transformation.

3.1 Specifying the Elaboration Process

As observed in section 2, design is a lengthy process of eliciting and resolving constraints [Waern 1986]. In [Schön 1983], Schön discusses how in this process the designer observes her emerging design while she shapes it. She is involved in a reflexive practice. Consequently, the structure of the resulting product differs from the structure of the elaboration process and is known only when the process ends.

Our goal is the production of genre compliant documents, not simply their description. While a description satisfactorily characterises the form of a document instance, it contains no indication of how such a document may be produced. Consider, for instance, a result expressed as a SMIL specification of the temporal constraints between the samples to be displayed in an `introduction`. This result might specify that a video-segment should be clipped while playing, but SMIL does not have the means to express a choice of video-segments from an archive, in term of their semantic and rhetorical roles, or features such as duration or visual attributes. By way of contrast, an approach which models the construction process by describing the human producer's expertise as she builds up a genre compliant document will be far more expressive in this regard.

Accordingly, we have chosen a paradigm close to the work strategy of a human author. This approach has also made it easier for us to capture human expertise. In overall terms, an author typically has a set of candidate segments to which she assigns a ranking, and then selects segments to maximise the overall ranking while satisfying the constraints. In more detail, when one observes the behaviour of an expert at work, it becomes apparent that approach is an incremental one, based on an overall strategy of *construct*, *examine* and *improve.*

- The author starts with an *intended document*, that is a draft document, which both provides the overall infra-structure of the target document, and also contains abundant candidate data, from among which the elements of the target document are to be chosen;
- She then iteratively transforms the document, making it increasingly compliant, until she determines that it cannot be further improved.

The process is not monotonic but may involve backtracking when locally appropriate choices lead to failure elsewhere. In accordance with this observation, our work uses an iterative transformation process which progressively improves the target document, rather than producing a document by, say, a single XSLT-style source-to-target transformation. We elaborate on and further justify this approach in the subsequent two subsections.

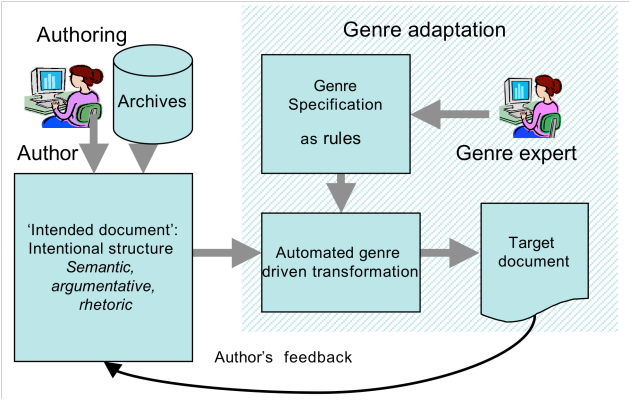


Figure 3: Incremental genre driven edition process.

3.2 Incremental Improvement Strategy

Most successful human processes rely on an incremental improvement strategy. Observing what is wrong with and proposing how to incrementally improve upon something which already exists, is often far easier than attempting to produce a complete, optimal solution from scratch. The maxim *first produce something and then improve upon it* embodies a well known, rigorous scientific guideline to address complex system issues, practiced in many domains ranging from art to industry. Furthermore, we note that human expertise in creative tasks relies heavily on the feedback cycle as introduced in the action theory [Norman 1986], in which new action is driven from the evaluation of the current state; an accomplished designer will quickly sketch an initial design or prototype, but will then iteratively observe how to improve upon it. Accordingly, rather than describing a genre compliant document, we have chosen to characterize which properties seem unacceptable for a document of the intended genre and to propose corresponding corrective actions to change a document to be closer to the chosen genre.

3.3 The Starting Point

Any incremental improvement approach pre-supposes an initial object, an initial document in our case. Genre conformance should be regarded as a transformation which preserves the *intention* of communication of the author, already embodied in a first draft intended document. In this approach, as detailed in [Gaillard 2007] and [Nanard 2005], and as recalled in section 2, the authoring activity starts by creating an intended document, which contains data which may or may not be kept in the target document and whose initial structure may evolve through the incremental transformation process. Thus, genre conformance in our approach may be viewed as post-processing the intended document.

4. TECHNICAL ASPECTS OF THE SOLUTION

4.1 Solution Genesis

Our initial experiments in producing genre compliant documents made use of XSLT transformations. This first study indicated some major drawbacks to XSLT, notably its lack of flexibility when it is necessary to express complex computations on attributes. Even more problematic was the necessity to conceive

the transformation as a whole, which becomes difficult when transformations are complex or are fuzzily specified, as are genre constraints. The need to be able to break the transformation into more pieces is reinforced by the incremental aspect of design. As a consequence we have studied the usability and the effectiveness of approaches other than XSLT.

We observed a close similarity between processing genre compliant documents and the translation process between natural languages. Both processes must preserve the author's intention, must comply with contextual rules which exist on the target document, and both processes are handled as document transformation. Based on this observation we realised that general tools which are relevant for natural language processing could be applied to the process of producing documents complying with a given genre.

We recall that two distinct approaches are used for language processing, namely

- the *productive* approach, which aims at formally defining a language as, for example, the set of words derived from its axiom by production rules;
- the *interpretative* approach, which aims at building a function over sets of words which exhibits an interpretation of their structure, so that an existing word can be transformed into an actual structure.

As our discussion in section 3 suggests, of these two, the interpretative approach seems particularly well suited to our needs. An interpretative approach makes possible the local recognition of complex situations within a document. Indeed, it is for this precise reason that the interpretative approach is well suited to natural language translation since such an approach may supersede a grammar in identifying specific constructs which need specific translation. In our case, the interpretative approach through incremental transformation promotes the emergence of structure complying with a genre.

As a consequence, our work makes use of a particular interpretative system known as SYGMART [Chauché 1984]. SYGMART relies on Markov algorithms [Markov 1962], which are formal re-writing systems. Initially designed for natural language processing, SYGMART operates internally by transforming strings which represent trees. SYGMART tries to exhibit how the elements of sentences, even invalid ones, are related. Both the general tree transformation formalism of SYGMART, and its features, which we discuss in what follows, encouraged us to consider how to apply techniques which have been successfully used in Natural Language Processing to the problem of genre compliance.

4.2 SYGMART: a General Transformation Engine

SYGMART comprises a set of engines dedicated to the analysis and translation of natural language texts. The central component of SYGMART is the TELESI engine, which iteratively transforms one decorated tree into another, according to a general rule-based tree transformation specification given in the SYGMART-TELESI language. Two other minor pre- and post-processing rule-driven engines are used as input/output processors. The first of these breaks down the input text into lexemes, and converts it into the initial decorated tree on which TELESI operates. The second converts the final internal decorated tree into an output text. SYGMART has no predefined entry or output language since it can process any language, natural or otherwise. One can use any kind of rules according to the needs of the particular application.

The tree transformation performed by the TELESI engine iterates as long as the evolving structure permits transformation rules to fire. The final result is obtained when the tree has reached a steady state in which no rule fires. Thus, the source and result documents simply represent the initial and the terminal states of the tree.

The SYGMART engine operates on decorated trees whose nodes refer to attribute blocks, which are built on classical primitive types including *integer*, *Boolean*, *float*; *string*, *enumeration*, *reference*. Attributes are directly evaluated, handled, computed and reassigned whenever appropriate during the transformation process, as determined by the rules. The document content itself, represented for instance as lexemes, is consistently handled within attributes blocks and thus is neither in the tree, nor the tree itself as a DOM tree would be. Thus, in the implementation, each node of the internal SYGMART tree contains only one reference to a computable attribute structure, enabling extremely fast and powerful processing.

SYGMART defines a transformation by means of sets of *rules*. Each rule defines a matching *condition* (the *trigger*) and a replacement *action*. Since the attributes decorate the tree but are not part of it, the condition and the action parts of a rule are described using two parts: one related to the tree and one related to the attributes (for examples of rules, see section 4.4.2).

The trigger defines a tree *pattern* and characterizes the attribute values which are evaluated respectively on the full tree and in the attribute blocks, to determine if the rule fires. If the rule fires, the action part defines the new tree pattern which replaces the tree pattern defined in the trigger, and specifies how to compute attributes.

As a consequence, a rule specifies

- how a tree pattern evolves by the transformation regardless of any attribute;
- how node attributes are evaluated in the trigger and computed when the transformation fires, regardless of their location in the blocks. It is worth mentioning that a given attribute block may be referenced by several distinct nodes or aliases.

SYGMART rule sets are structured into *steps* and *grammars*. In contrast to programming languages, there are no explicit control structures in SYGMART. The transformation process is implicitly controlled by clusters of ordered rules, called grammars. At any given time, only the rules of one particular grammar are considered. The end of the grammar is reached when none of its rules fires. A rule may explicitly call one or several grammars on selected sub-trees. A set of grammars within a step is organized as a graph, with possible cycles, called a *grammar network*. Each grammar has a *nextgrammar conditional list,* which defines which action is to be performed, according to the current state, when the end of that grammar is reached.

One may specify conditions to either

- select a new rule cluster to activate (explicit links to) other grammars,
- end processing of the grammar (or return from the current call),

- backtrack to the state the system was in before the activation of this ill-finishing cluster and proceed further in selection within the next grammar list which led to the ill-finishing grammar.

A step ends when none of its grammars is active. Unlike grammars, steps are always sequentially ordered, and their sequence can be easily driven as a runtime parameter.

Full details of SYGMART are beyond the scope of this paper, but may be found in [Chauché 2001].

4.3 Using SYGMART for Genre Driven Production

Many of the features of SYGMART are well suited for handling complex and fuzzy transformation processes. In particular, the ability to organize the specification of the rule sets in terms of steps and grammars greatly improves readability and reduces the complexity of designing a transformation by permitting it to be described it aspect by aspect. This is possible due to the iterative process operating within the document tree. These features make it possible to specify separately specific aspects of genre such as narration, argumentation, final form, etc. Furthermore backtracking is a valuable feature to address fuzzy aspects of genre. Thus we use SYGMART as the internal engine for genre driven document production.

However, since most of our data are described in XML, both input and output (for which we use SMIL[4]), we had first to adapt SYGMART to operate on XML documents. In order to do so, we developed in Java a work environment known as TTE (*Tree Transformation Environment*) and an associated language known as TTL (*Tree transformation language*). TTE and TTL alleviate some details of the syntax of the SYGMART-TELESI language by providing an XML-like form for SYGMART-TELESI rules. One can find a detailed description of TTL in [Nanard 2006].

In the remainder of this section we discuss some specific issues relating to the use of the SYGMART system in producing genre compliant documents, without emphasizing TTL syntax details. These issues are all closely related to the document engineering aspects of our approach, and include such engineering-related issues as the separation of form and content, specification, re-use (of both design and content), and modularity (again of both design and content.

4.3.1 Reusing Genre Constraints

As mentioned earlier, one of our major objectives is to make use of the expertise of authors in order to automatically adapt an intended document to a given genre. This is analogous to the ability to change the appearance of a document simply by running another XSL-Stylesheet to a given XML source. Genre related transformations must be designed to characterise a genre in a general enough manner to be reusable on different intended documents.

As illustrated in figure 3, we distinguish two kinds of users:

- The genre *design expert*, who is an individual capable of creating reusable TTL steps and grammars, which transform a document to make it comply with some aspects of a given genre. She specifies the required grammar networks, including backtracking when needed. The genre design expert tunes and makes available collections of grammar steps, each of which addresses a given aspect of genre compliance.
- The document *author* is more concerned with the document contents. She focuses on aspects such as content selection, narrative and argumentative structures, of course, designed with the target genre in mind. She is no more concerned by other details of genre than a textual document author would be by such non-content issues as font sizes, orphan and widows lines, or floating images. In both cases issues going beyond content are expected to be handled automatically.

Typically, a genre design expert delivers the means to run a reusable genre transformation on the various instances of documents of a given document series, while the document author delivers the initial intended document[5] instances of the series.

4.3.2 Attribute Ontology.

To support the form of reuse just described, a shared vocabulary between the genre specification, the document authoring, and the archive indexing is essential. Such sharing is required to match the intended document description, expressed as properties and structure of contents, speech acts, author's intentions, discourse architecture, and so forth, with the attributes in the rules specifying a genre. Attributes are not specific to a given source document, with an *ad hoc* TTL transformation to be applied to it, but are shared by all instances of a given series and the corresponding TTL genre specifications.

Significant work on vocabulary definition has therefore been undertaken, which has led to a precise *ontology* upon which attributes names are defined. The work on this ontology was performed as part of the SAPHIR project, and for our purposes, we restrict ourselves to noting three major parts of this ontology:

- domain related concepts, which are used for indexing content semantics;
- document discourse related concepts, such as narration, discourse, arguments, speech acts;
- media expression related concepts such as duration, view, motion, image quality, interdependency between sound and image tracks.

Attributes handled in the TTL rules are built with respect to this ontology. The majority of these attributes concern the target document space, rather than the domain space.

Domain and content oriented vocabulary are used mainly in the earlier stages of document preparation for indexing archive data, and for searching and selecting relevant documents from among indexed archives, and are not usually of direct concern to the genre compliance transformation process.

Document related and media related concepts are those of greatest relevance to our description. Media related indexing [Chamming's 2003] is necessary to help evaluate the actual contextual usability of a given media item whose content has been retrieved as relevant in terms of domain concepts. Most of the genre rules are concerned with this aspect; see for example the rule in section 4.4.3.

[4] To mount videos from fragments.

[5] In the sense of section 2.

4.3.3 Intended Document Specification

An intended document structure expresses the author's intentionality, and involves more than simple descriptions of containment. The term *intended document* denotes an organized set of information which specifies, in terms of speech acts, the narrative and argumentative structure the author wishes for her target document. Additionally, within the intended document the author proposes various content items as possible elements to be used in the speech acts. As indicated earlier, in our approach the author of an instance of a document in a series elaborates only an intended (source) document which represents her intended communication.

In the example, as discussed below, when the author identifies a part of the document as an *introduction*, she is doing more than giving a name to a part or specifying a logical structure class to map a presentation rule. Rather, she is specifying the role of this part within the narrative structure; she invokes it as a speech act, and thereby implicitly selects the organization rules which must be respected in order to comply with the desired genre. Similarly, in the intended document the author will propose a number of different video-segments to fill a single role, thereby indicating that the author considers these segments equivalent for her purpose -- it is for the system to choose which combination of author specified content-items is the best fit to comply with all the constraints relevant to the genre in question. Consequently, individual content items of the intended document may or may not be present in the final document when the iterative process of transforming the source document to one which complies with the genre constraints is complete.

In technical terms, the intended document is described as a XML document, most of whose elements mainly express speech acts, discourse construction, and reference candidate media items. The first steps of TTL, which are transparent to the user, transform this XML file, regarded considered by SYGMART as a simple sequence of characters, into an initial internal SYGMART tree representing the source DOM tree.

4.3.4 TTL Grammar Steps

Production of the final document is performed as a TTL transformation. This transformation applies a set of genre-adapting grammar rules, provided by the genre domain expert described earlier. In order to address incremental description and reuse issues, the transformation specification is organized as a collection of *steps*, each of which addresses some specific aspect of the genre compliance. These steps mimic the work an expert would perform when adapting an intended document to the desired genre.

Each step is organized as a network of grammars which perform the corrective transformations relevant to a specific genre-related issue in the document. This modularity enables the genre expert to reduce the complexity of her task by splitting it into smaller parts, and to elaborate libraries of off-the-shelf reusable steps representing strategy variants for dealing with a given genre aspect. Additionally, one may specify and implement transformations at a higher level of abstraction, by appropriately combining steps.

4.4 The TV News Example

We now discuss the implementation of our example. As indicated in section 2.2, the object is to automatically build news digests on given topics by reusing richly indexed segments available in the audiovisual archives. We assume that the author has made available an intended document, as described in section 4.3.3; exact details of this preliminary stage are given in [Gaillard 2007]. To simplify the presentation, we limit ourselves to a description of the adaptation of the speech act *introduction* within the document, and we concentrate on the principles behind the approach rather than giving full details of the grammars used.

4.4.1 Intended Document Data

In the intended document, the author describes the planned narration, and references the possible elements which could be used in the target document. She suggests starting the *digest* with the speech act *introduction*, and provides several possible sound clips which she judges to be appropriate as an introduction, together with a set of short illustrative video-sequences. A general ranking is already elaborated for each of these elements, either automatically by the search engine which delivered them, or as explicitly adjusted by the author. Furthermore, metadata from the rich indexing information of each archive segment, particularly the segment's visual characteristics, are preserved within the intended document. The speech act *introduction* is also used elsewhere, for instance to introduce sub-topics of the general discourse.

4.4.2 Genre Compliance

A genre expert has made available a reusable set of TTL steps which transform the intended document source into a target document compliant with the *TV news* genre described in section 2.2. To illustrate the variety of possible treatments, in the following we point out some typical roles of TTL grammars involved in the process, rather than describing individually each grammar.

Converting structural properties into attributes. A first set of TTL grammars is concerned with pre-processing the structure so as to elaborate SYGMART attributes from the XML source file in order to boost computation. For instance, the TV news genre implies that the first introduction has an organization different from other introductions. To manage this, let us consider a Boolean attribute *first_intro* (with the obvious meaning). Manifestly, this attribute cannot be present in the intended source file, since the distinction depends exclusively upon the genre, but equally this attribute is extremely useful for later computation. Such transformation of a structural property into an attribute, computed once, improves both the readability and the efficiency of other rules, since subsequent matching can be expressed in terms of and triggered on this attribute, rather than on the structural property. Many similar structure-to-attributes conversions are processed as preliminary steps.

Computing new attributes. A second use of TTL grammars is to make use of simple attributes to store complex properties which will be used later in the transformation. For instance, the choice of samples to be presented during the introduction relies upon a rank sorting of candidate segments. Once an audio-track introduction is selected from among the candidates, the introduction duration is determined. Only then does it become possible to calculate admissible duration ranges for samples, and hence a contextual ranking for each sample segment which may appear in the introduction can be computed. Computing such a ranking may be complex, and may depend upon other data, such as duration, interest, visual parameters, and so forth. So, evaluating it in context, but once for all, improves treatments. We note that this

type of computation is easily expressed in TTL, whereas it would be extremely difficult to use *xsl:variables* to do so.

Specification as sets of corrective actions. The most frequent role of steps is to mimic the process which adapts a document. In our example, an author would choose the set of samples so that its elements be the best possible ones according to their interest, but organized so that media related constraints be satisfied too. The grammar organization and rules specification is directly a formalization of this process: one grammar computes contextual ranking, one sorts elements, one deals with constraints of adjacency. We have limited the example to introductions. Practically the expert writes steps or grammars which concern the various speech acts, and discourse structural elements which are relevant for the concerned genre. One may consider a rule as the recognition of a property which is unacceptable in the target document and achieve a corrective action: for instance, if two samples are close-up views, it is suitable to add a wide-angle between them, even if this sample is less ranked than the two others, as shown in the next subsection.

4.4.3 Using Grammars and Rules

We focus now on the TTL programming style. This is not a extensive list of possible roles and tricks. Each problem raises original solutions, and it is far out of this paper's scope to examine them.

Working on selected sub-trees. We note two important features of TTL which facilitate structural computing [Nürnberg 2003]: aliases and calls.

First, aliases make it possible to work more efficiently on structures. A new node resulting from a transformation references an attribute block, which is either totally new, or is a clone of an existing block, or may even be an existing block itself. This makes it possible to build new sub trees, the nodes of which refer to the same attributes blocks as other nodes (see figure 4).

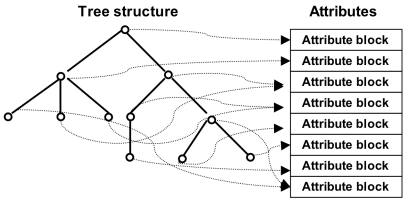


Figure 4: Separation of attribute blocks from tree structure.

Thus, these nodes behave like aliases. For instance, one may build several sub-trees containing aliases of segment descriptions, and then sort them according to various criteria.

```
<grammaire iterative="yes"
name="doSortAccordingToInterest">
<!-- This single-rule grammar recursively sorts a
sub-tree in which two nodes seg1 and seg2 are
considered. It is assumed that each of seg1 and
seg2 describes a segment. If the interest of the
seg2 is greater than that of seg1, then the two
segments are permuted.-->
<rule name="sortingSegments">
  <where>..seg1..seg2..</where>
  <when> seg1: element = "segment";
         seg2: element = "segment"
  </when>
  <if>interest(seg2) &gt; interest(seg1)</if>
  <becomes>seg2,seg1</becomes>
</rule> <nextgrammar name="%STOP"/> </grammaire>
```

A second TTL feature supporting operating on selected sub-trees, is the *grammar call* which is more classical but nonetheless effective. We make extensive use of this feature in organizing the transformation process. In order to select segments used as samples, a first grammar typically builds a set of candidate segments. This grammar then calls a second grammar, which takes this sub-tree as parameter and sorts it according to a local ranking. The grammar then calls a third grammar to permute these segments to satisfy some other constraint, such as (in our example) the rule which states that *no pair of adjacent segments may be close-ups of the same individual*. The following rule shows how to express such a constraint as a corrective action:

```
<!--This rule works on a sub-tree samples, which
is presumed to be already sorted by ranking. Among
the nodes of samples we consider three, locally
named close1, close2, wide. close1 and close2 are
specified as adjacent. If close1, close2 and its
siblings up to wide are close-up views, while wide
is a wide-view, the tree is reorganized so that it
contains all items present in samples before
close1, then in sequence close1, wide, close2 and
its siblings-->
<rule name="noConsecutiveCloseupViews">
<where>samples(..close1*close2.. wide..)</where>
  <when> close1:view="closeup";
  close2:view="closeup";
  close2..:view="closeup";
  wide:view="wide";</when>
  <becomes>
sample(samples[..close1],
close1, wide,  close2, samples[close2..])
  </becomes>
</rule>
<!—we recall that this rule operates as long as
its condition matches-->
```

Once segments are ordered and comply with adjacency constraints, a grammar determines their begin and end. At end, a grammar deletes from this sub-tree the elements which start beyond the allotted duration for samples. Working on a sub-tree of aliases rather than on the initial items preserves them for possible backtracking needs.

Backtracking. A notable feature of TTL-SYGMART is its backtracking capability. This feature is extremely useful for supporting the fuzziness of genre constraints which often contain statements such as "xxx would be the best but, if not possible, then yyy would also be acceptable". In our example, once an ordered set of potential samples is elaborated, we need to choose which segments are actually to be kept in the final document. The total sample duration time must respect certain constraints; for instance it must allow enough time, but not too much, to have the presenter on screen at the beginning of the introduction, as illustrated on figure 5.

If the currently elaborated sequence is not compatible with this constraint, we must select and examine an alternative. At this point the grammar triggers backtracking and the next grammar in a conditional list is executed to explore another strategy. In our example, one such strategy might be to simply toggle the visual constraints on the first segment of the sample. Starting with a wide-angle view rather than a close-up is usually less pleasing, but, if the number of samples is odd, this changes the entire selected set and thereby deliver a new total duration. Backtracking being a general feature, several strategies can be checked in this

fashion. Backtracking may go back further and, for instance, check the impact of using, say, a completely new anchorman.

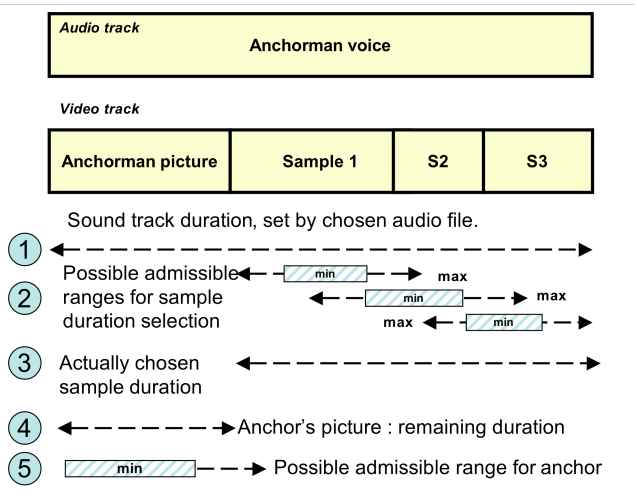


Figure 5: Typical dependencies between duration constraints.

We note that the strategy implemented is not to search extensively for a formal optimum. Rather, the "best" solution consists in accepting the first compatible solution which is enunciated as being the preferred one in the genre description according to the criteria value provided by the author in the intended source document.

5. DISCUSSION AND CONCLUSION

We have described an approach towards providing reusable and non-specific specifications for certain aspects of genre. It should be made clear that in addition to the generality and the actual usability of the mechanisms described, significant work to address the complex problem of genre compliance in its whole generality would still be necessary. The TTE/TLL tools have been in use for over two years, and the experiments demonstrate that tasks which would usually require human expertise can be successfully automated for strongly typed genres. In addition to the current example, we have also used this technique to produce websites presenting news.

Programming specific applications, in Java say, to handle the current example might appear simpler, but leads to intricate *ad hoc* solutions. Conversely, a transformation-based solution appears promising. The transformation of documents based on their structure has long played a central role in a variety of applications of digital documents, and there is a considerable literature on this topic. Early work on the use of such transformation techniques in order to produce target documents of a variety of types appears, for example, in [Akpotsui 1992] and [Kuikka 1995], which discuss grammar based transformations, and in [Chiba 1995] and [Furuta 1988] which presents tree-based transformations. In [Amaneddine 2003] a taxonomy of transformation methods is presented. More recently, document transformation work has centered on XML technologies and some typical work on XML document transformation is reported in Leinonen 2003]. Work on using the XSL language as a vehicle for transformation appears in [Maneth 2004]. It should be noted that the work cited thus far is exclusively concerned with transformation of the *logical* structure of a document, as expressed in XML or some similar notation. More recent work on document transformation, including our own, takes a broader view of document structure. In [Lumley 2005] a unified multi-structured document view is discussed, while in [Bouayad 2000] the relationship between logical (text) and rhetorical structure is discussed. Some early work on Genre driven document production appears in [Bateman 1995] and [Vaughan 1998].

Our choice of SYGMART, a scientifically sound natural language processing tool, rather than XSL, comes from the need for a strong formalism which is sufficiently general to handle in a unified manner the diversity of constraints which appear in genres. Markov transformations[6] are known as one of the three fundamental programming paradigms (recursive functions, Turing machine, and Markov transforms). Transformation based approaches are well suited to deal with document engineering.

As mentioned earlier, XSLT provided our first test bed, but we discovered that XSLT exhibited too many limitations. In an XSLT transformation, a generated element can no longer be changed unless a new transformation is run with the generated file as input data. In TTL, on the other hand, several rules may fire at different times on the same sub-tree which may thus evolve in an incremental fashion. Source and result are simply different states of the same tree being iteratively transformed. XSLT and TTL also differ in their behaviour in respect of non-matching items. XSLT expresses one-shot *production*, and any source document part which matches no pattern is lost. TTL expresses iterative *change*, and any part of the tree which matches no pattern simply remains unchanged. This distinction has important consequence in terms of design. Designing an XSL template requires mastery of all the necessary knowledge to produce the correct result in a single transformation. Conversely, a TTL transformation can readily be decomposed into simpler parts, which are easier to conceive, to describe and to reason about, without any need to consider the detail of these rules. Sets of rules which concern particular situations can be designed separately. This approach represents a profound difference of design paradigm; the TTL approach provides smaller, reusable parts to specify genre in terms of transformations.

Attributes in TTL are not limited to those of the source document and new attributes can be created at will as needed. Attribute computability gives a great flexibility and power of expression to the language. Such expressive power is lacking when using XSL, which is hampered by the source DOM structure. Transformation rules are often specified with meaningful attribute names, closer to the author's concepts than to the source syntax. This ability improves the reusability of rules., A preliminary transformation step can, if necessary, elaborate such reusable internal attributes from the external attributes present in a specific source document.

Finally we note that the backtracking feature is extremely helpful in dealing with the fuzziness of genre specifications. This important aspect of SYGMART has no counterpart in XSL.

The approach described in this paper provides a generic technical solution for handling genre constraints in order to produce genre compliant multimedia documents. Exhibiting rules specifying particular genres remains an open problem, one that is more concerned with knowledge elicitation than with technical aspects.

[6] See http:en.wikipedia.org/wiki/Markov_algorithm

6. REFERENCES

[1] Amaneddine, N., Badr, Y., A Taxonomy of Transformation Methods for Structured Documents. In*: The Second ACS/IEEE International Conference on Computer Systems and Applications* (AICCSA), Tunis- Tunisie, 14/07/03-18/07/03, IEEE, Juillet 2003.

[2] Akpotsui, E. , Quint, V., and Roisin, C., Type Modelling for Document Transformation in Structured Editing Systems, *Mathematical and Computer Programming*, Special Issue devoted to the 1992 Workshop on Principles documents.

[3] Bateman, J. A. and Teich, E., Selective information presentation in an integrated publication system: an application of genre-driven text generation. *Information Processing and Management,* 31, 5, 753-768, 1995.

[4] Bouayad-Agha, N., Power, R., and Scott, D., Can text structure be incompatible with rhetorical structure, *Proceedings of the first international conference on Natural language generation*, 2000, Mitzpe Ramon, Israel.

[5] Chamming's, L. *Pour une description sémiotique de l'audiovisuel*. Actes du colloque international "sémiologie 2003". Paris, Sorbonne, Novembre 2003.

[6] Chandler, D., An introduction to genre theory. 1997, Available online at (accessed 24 February 2006): www.aber.ac.uk/media/Documents/intgenre/intgenre.html

[7] Chauché, J., Un outil multidimensionnel de l'analyse du discours. In *Proceedings Coling84*, 10th International Conference on Computational LINGuistics, Stanford University, California, 1984, 1984, 11–15.

[8] Chauché, J., Sygmart Reference Manual, 2001, see: www.lirmm.fr/~chauche/REFERENCESYG/docsyg.html

[9] Chiba, K. and Kyojima, M., Document transformation based on syntax-directed tree translation, *Electronic Publishing: Origination, Dissemination, and Design*, 8(1), 15–29, 1995.

[10] Crowston, K., Williams, M., Reproduced and emergent genres of communication on the Worl-Wide-Web, in *Proc. 30 Hawai¨Conference on System Sciences* ((HICSS97), vol. VI, 30-39, 1997.

[11] Duff, D., *Modern Genre Theory*, London: Longman, 2000.

[12] Falkovych, K.I., Nack, F.-M., Composing discourse based on genre semantics, *Report INS-E0516*, December 2005 (CWI).

[13] Fowler, A., Genre, In Erik Barnouw (Ed.): *International Encyclopedia of Communications*, Vol. 2. New York: Oxford University Press, 215-7, 1989.

[14] Furuta, R. and Stotts, P. D., Specifying structured document transformations, in *Proceedings Electronic Publishing'88*, 109–120, Cambridge University Press, 1988.

[15] Gaillard, L., Nanard, J., Bachimont, B., Chamming's, L., Intentions based authoring process from audiovisual resources, Research Report RR07008 LIRMM, 2007.

[16] Geurst, J., Bocconi, S., van Ossenbruggen, J., and Hardman, L., Towards ontology-driven discourse: from semantic graphs to multimedia presentations. In Second International Semantic Web Conference (ISWC2003), 2003, 597–612.

[17] Hayes, J.R. and Flower, L.S., Identifying the organization of the writing process, in L.W. Gregg, E.R. Steinberg (Eds), *Cognitive process in writing*, Lawrence Erlbaum Associates, 1980, 3–30.

[18] Iksal, S. Garlatti, S. Adaptive special reports for on-line news papers. In: *Proceedings of the Workshop on Electronic Publishing, associated to the conf. Adaptive Hypermedia* (AH'2002), Universidad de Malaga, 2002, 27–30.

[19] Kuikka, E. and Penttonen. M., Transformation of structured documents. *Electronic Publishing: Origination, Dissemination, and Design*, 8(4), 319-341, December 1995.

[20] Leinonen, P., Automating XML document structure transformations, document querying and transformation, *Proceedings of the 2003 ACM symposium on Document Engineering*, Grenoble, France, 2003.

[21] Lumley, J., Gimson, R., and Rees, O., A Framework for Structure, Layout and Function in Documents, Digital Media Systems Laboratory , HP Laboratories Bristol, Tech. Report HPL-2005-95(R.1), December 19, 2005.

[22] Maneth, S., and Neven, F.*, Structured Document Transformations Based on XSL*, LNCS, 1949/2000, 2004.

[23] Markov, A., *The theory of algorithms*, Tr Mat. Inst. Steklow XLII, 375 pp. Translation: Office of Technical Services, U.S. Department of Commerce, Washington, D.C., 1962.

[24] Nanard, M., Nanard, J., Betaille, H., Beyond logical structure, in *Proceedings Symposium MetaInformatics*, 2005. (ACM Conference Proceedings Series).

[25] Nanard, M., Nanard, J., Chauché, J., and King, P.R., A Structural Computing Approach to the Production of Multimedia Document Series, *New Review of Multimedia and Hypermedia*, Vol. 12, n°2, 2006, 165-190.

[26] Norman, D. A., Cognitive Engineering, in S. Draper, Donald A. Norman, *User centered system desig*n, Lawrence Erlbaum Associates, Publishers, 1986.

[27] Nürnberg, P. J., Wiil, U, K., and Hicks, D., L., A Grand Unified Theory for Structural Computing, MetaInformatics 2003, LNCS vol. 3002, Springer Verlag, 2004.

[28] Price, S.L., Semantic Components: A new model for enhancing retrieval of domain specific information, Department of Computer Science Portland State University.

[29] Saphir project, see: http://www.ina.fr/recherche/projets/encours/saphir.fr.html

[30] Schön D., *The reflective practitioner,* Basic Books, 1983.

[31] Vaughan, M. and Dillon, A. The Role of genre in shaping our understanding of digital documents. In Preston, Cecilia M., Eds. *Proceedings American Society for Information Science* 35, 559-566, Pittsburgh, PA, 1998.

[32] Wærn Y., *Cognitive aspects of computer supported tasks,* Wiley, 1986.

[33] Whiting, M.A., Cowley, W., Cramer, N., Gibson, A., Hohimer, R., Scott, R. and Tratz, S. Enabling Massive Scale Document Transformation for the Semantic Web: the Universal Parsing Agent TM, *Proceedings ACM Symposium on Document Engineering*, Bristol 2005.

[34] W3C, XML, XSL, see: http://www.w3.org