Automatic Generation of generic Bitstream Syntax Descriptions using gBFlavor

Davy Van Deursen

Supervisor(s): Rik Van de Walle

Abstract— **The heterogeneity of coding formats in today's multimedia environments causes problems for developers of media resource adaptation software because new software has to be written for each coding format. The MPEG-21 framework provides tools assisting in the adaptation of bitstreams in a coding-format agnostic manner. MPEG-21 generic Bitstream Syntax (gBS) Schema is one of these tools and offers the possibility to adapt a bitstream based on its high-level description. This paper proposes a new method for the generation of these high-level descriptions (i.e., generic Bitstream Syntax Descriptions (gBSDs)). The introduced technology, called gBFlavor, is able to automatically generate a coding-format specific parser which produces a gBSD of the given input bitstream. The paper shows that this approach, in comparison with existing technologies, not only leads to a user-friendly creation of gBSDs, but also performs better in terms of execution time.**

Keywords— **Format agnostic adaptation framework, gBS Schema, gBFlavor, MPEG-21**

I. Introduction

The tremendous variety on end-user terminals and network technologies has lead to new views on multimedia access. People want to access multimedia anywhere, anytime, and on any device which is nowadays better known as Universal Multimedia Access. One important tool hereby is the use of scalable coding which implies that a media resource is coded only once (at a high quality) whereupon it can be decoded many times (usually obtaining a lower quality) targeting specific end-user devices and network technologies.

Aside the diversity in devices and network technologies, more and more different coding-formats are developed. This phenomenon makes it hard for developers of media resource adaptation software (e.g., an extractor for a specific scalable coding-format). A solution to this problem is the use of a format-agnostic adaptation engine. The MPEG-21 framework, which tries to realize the 'big picture' in the multimedia production, delivery, and consumption chain, comes with two tools that enable a format-agnostic adaptation process for media resources.

II. Bitstream Syntax Descriptions in MPEG-21

The MPEG-21 framework offers two tools to assist in customizing (scalable) bitstreams in a format-agnostic manner [1]. The first tool is called the Bitstream Syntax Description Language (BSDL) and is built on top of the World Wide Web Consortiums (W3C) XML Schema Language. BSDL is able to describe the structure of a (scalable) bitstream in XML format using its BintoBSD parser resulting in a Bitstream Syntax Description (BSD). This parser needs a Bitstream Syntax Schema (BS Schema), which contains the structure of a certain coding format. The BSDtoBin parser, which takes as input the BS Schema, the customized BSD, and optionally the original bitstream, is able to produce the adapted bitstream.

D. Van Deursen is with Multimedia Lab, Electronic and Information Systems Department, Ghent University - IBBT, Ghent, Belgium. E-mail: davy.vandeursen@ugent.be .

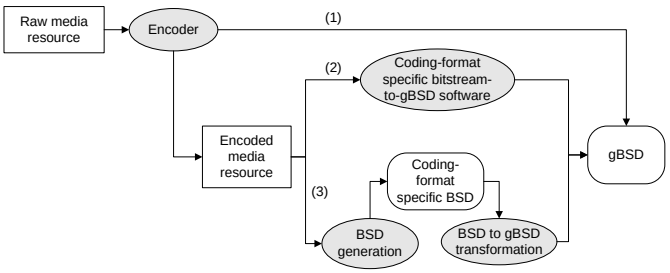

Fig. 1. Possibilities to generate a gBSD: use of dedicated software (option (1) and (2)) or transformation of coding-format specific BSDs (option (3)).

The second tool, called generic Bitstream Syntax Schema (gBS Schema), goes one step further than BSDL. There is only one BS Schema (the gBS Schema) which is used by the gBSDtoBin parser to produce an adapted bitstream. This parser needs a generic Bitstream Syntax Description (gBSD) implying that this description is format-agnostic. However, it is not possible to construct a format-agnostic parser which produces gBSDs (like the BintoBSD parser in BSDL) because of the generic character of gBS Schema.

III. Generation of gBSDs

The generation of gBSDs is not a straightforward task as mentioned in Section II. In Fig. 1, an overview is given of the possibilities to generate a gBSD.

A. Dedicated Software

One possibility to automatically generate gBSDs is to develop dedicated software. For instance, one can extend a specific encoder in a way that it is able to generate the gBSD during the encoding process (option (1) in Fig 1). Another way is to write a parser for a specific coding-format that is able to generate gBSDs (option (2) in Fig 1). An advantage of using dedicated software to generate gBSDs is the performance in execution speed because the software is targeting only one coding-format. The largest disadvantage is the coding-format dependency of the software. Indeed, when a new coding-format has to be supported, new software has to be written. Moreover, generating gBSDs during the encoding process is useless when the media resources are already encoded.

B. Transformation of coding-format specific BSDs

Another possibility to generate gBSDs is transforming coding-format specific BSDs into gBSDs (option (3) in Fig 1).

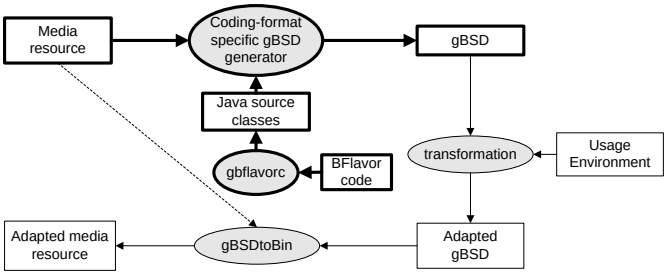

Fig. 2. The gBFlavor architecture.

TABLE I
SIMULATION RESULTS OF THE GBSD GENERATION PROCESS.

bitstream		BSDL + STX			gBFlavor		
name	size (MB)	ET[a] (s)	ES (Mbit/s)	MC (MB)	ET (s)	ES (Mbit/s)	MC (MB)
Crew	9.0	8.0	9.0	1.6	2.4	29.4	0.7
Sailormen	11.6	9.2	10.1	1.6	2.9	32.5	0.7
Driving	6.3	6.7	7.6	1.6	1.8	27.5	0.7
Whale_Show	17.0	11.3	12.0	1.6	3.3	40.6	0.7

[a] ET, ES, and MC denote Execution Time, Execution Speed, and Memory Consumption, respectively.

First, a coding-format specific BSD is generated from a media resource using BSDL or BFlavor [2] for example. Afterwards, the coding-format specific BSD is transformed using common XML transformation technologies like eXtensible Transformation Stylesheets (XSLT), Streaming Transformations for XML (STX), or dedicated software (for example using Simple API for XML (SAX) or Document Object Model (DOM)). Note that the transformation stylesheet is also coding-format specific implying that for every coding-format, a new stylesheet has to be created. The disadvantage of this approach is that firstly two technologies have to be used in order to obtain a gBSD and that secondly this two processes are both coding-format specific.

IV. GBFLAVOR: AUTOMATIC GENERATION OF GBSDS

In order to offer a better solution for the creation of gBSDs, we propose a new technology called gBFlavor (gBS Schema + BFlavor). It is built using the same principles of BFlavor [2] which includes the automatic generation of a coding-format specific parser that is able to produce a BSD for a given bitstream.

A. Architecture

In Fig. 2, the functioning of a gBFlavor-based adaptation framework for media resources is shown. The gBSD generation process is illustrated with bold lines. One has to start with the creation of a BFlavor code for the given coding-format. This BFlavor code contains the high-level syntax of the coding-format in a C++-like manner. Note that this BFlavor code is exactly the same code to use for a BFlavor-based adaptation framework [2]. Given such a coding-format specific BFlavor code, the `gbflavorc` translator (which is built on top of the bflavorc translator) is able to produce Java source classes that are compiled to a coding-format specific parser. This parser takes as input a given media resource and generates its gBSD.

To complete the adaptation chain, the automatically generated gBSD can be adapted according to the given user environment (e.g., dropping some packets). The resulting gBSD is then fed to the MPEG-21's gBSDtoBin parser together with the original media resource in order to generate an adapted media resource.

B. Experimental results

After discussing the global functioning of gBFlavor in the previous subsection, a performance evaluation is given of the gBSD generation process in terms of execution times and memory usage. We used four H.264/AVC encoded bitstreams. Two of them have a resolution of 1280x720 and contain 600 frames (Crew and Sailormen) while the other two have a resolution of 720x480 and contain 450 frames (Driving and Whale_Show). Only solutions pluggable into format-agnostic adaptation frameworks are evaluated implying that the gBSDs are generated using MPEG-21 BSDL followed by an STX transformation and gBFlavor.

In Table I, the performance results are given for MPEG-21 BSDL followed by STX and gBFlavor. It is clear from this table that gBFlavor outperforms the combination of MPEG-21 BSDL and STX in terms of execution time/speed by a factor three. One can also see that both technologies are characterized by a low memory consumption. The BSD generation of MPEG-21 BSDL is the bottleneck for the BSDL-STX combination (the STX transformation takes about 1 second). The parser generated using gBFlavor is very fast because it is coding-format specific, nevertheless it is generated in an automatic way.

V. CONCLUSIONS AND FUTURE WORK

In this overview paper, a new method, called gBFlavor, for the automatic generation of gBSDs was presented. It relies on the automatic generation of a coding-format specific parser which is able to produce a gBSD given a media resource. Using the gBFlavor approach, the generation of gBSDs can be achieved by simply creating a BFlavor code for a given coding-format. This solution is not only more user-friendly than existing methods like dedicated software and the use of MPEG-21 BSDL followed by a STX transformation, but outperforms them in terms of execution time. Future work consists of targeting specific applications enabling the generation of application-specific gBSDs implying an easier adaptation of the gBSDs.

ACKNOWLEDGMENTS

The research activities as described in this paper were funded by Ghent University, the Interdisciplinary Institute for Broadband Technology (IBBT), the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research-Flanders (FWO-Flanders), the Belgian Federal Science Policy Office (BFSPO), and the European Union.

REFERENCES

[1] ISO/IEC JTC 1 *Information Technology Multimedia framework (MPEG-21) Part 7: Digital Item Adaptation*, October, 2004.

[2] W. De Neve, D. Van Deursen, D. De Schrijver, S. Lerouge, K. De Wolf, R. Van de Walle: *BFlavor: a Harmonized Approach to Multimedia Resource Adaptation, inspired by XFlavor and MPEG-21 BSDL*, accepted for publication in EURASIP Signal Processing: Image Communication Journal