Towards the NCL Raw Profile

Guilherme Lima
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 3503
gflima@telemidia.puc-rio.br

Luiz Fernando Gomes Soares
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 4330
lfgs@inf.puc-rio.br

Carlos de Salles Soares Neto
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 3503
csalles@telemidia.puc-rio.br

Marcio Ferreira Moreno
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 3503
marcio@telemidia.puc-rio.br

Romualdo Rezende Costa
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 3503
romualdo@telemidia.puc-rio.br

Marcelo Ferreira Moreno
Lab. TeleMídia - DI – PUC-Rio
Rua Marquês de São Vicente 225
222453-900 Rio de Janeiro, RJ
+55-21-3527-1500 Ext: 3503
moreno@telemidia.puc-rio.br

ABSTRACT

In this paper, we describe the first steps towards the definition of the new NCL (Nested Context Language) 3.1 profile, called Raw profile, aiming at a more efficient and more reliable implementation of the Ginga-NCL presentation Engine.

The new profile also allows the definition of a new transfer syntax notation, which will smooth the progress of integrating other programming languages to the Ginga-NCL environment. This will help the liaison between NCL and Ginga-NCL ITU-T Recommendation with other declarative hypermedia language standards.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Constructs and Features – *control structures.*

General Terms

Standardization, Languages.

Keywords

Nested Context Language – NCL, Ginga-NCL presentation environment, NCL EDTV profile, NCL Raw profile.

WebMedia 2010, October 5–8, 2010, Belo Horizonte, Minas Gerais, Brazil.

1. INTRODUCTION

NCL (Nested Context Language) digital TV applications that are in conformance with ITU-T H.761 Recommendation for IPTV services [6, 7] and in conformance with the Nipo-Brazilian ISDB-T_B Standard [1, 5] shall follow the NCL Enhanced Digital TV profile (EDTV) [9]. NCL 3.1 EDTV profile is based on a conceptual data model, called NCM – Nested Context Model [8], which favors the authoring process. NCM is a conceptual model with several reuse features. NCL extends these features with other syntactic sugar[1] add-ons in order to easy even more the authoring process.

On the other hand, the NCL player (called Ginga-NCL) should use a data model as close as possible to the presentation engine's execution platform in order to make a better use of the scarce resources of usual TV receivers to achieve an efficient and reliable implementation.

The different goals of these two data models make them semantically distant. As a consequence, the process to convert one to another, in the course of a DTV (digital TV) application presentation, can be very complex and prone to errors.

To solve this problem, an alternative is to sacrifice the authoring process, offering a lower-level authoring language. However this is against the DTV principles, mainly concerning its declarative environment, in which DTV application authors are the main focus. A second alternative, adopted in the current event-oriented reference implementation of Ginga-NCL, is to sacrifice the

[1] In the context of this paper, a syntactic sugar refers to a syntax within NCL that is designed to allow things to be expressed more clearly, more concisely, or in an alternative style that someone may prefer, while alternative ways of expressing them exist.

presentation engine conception, obliging it to perform using a data model moved away from its designing goals. This alternative, although possible, especially in high performance receivers, can lead to implementations more prone to efficiency and reliability problems.

Thus, a third alternative must be pursued. The solution can come from dividing the conversion from the authoring data model to the presentation data model into two steps. In other words, the solution can be the introduction of an intermediate conceptual data model between the two target ones. The problem is then moved to correctly choose this new data model to have both a translation process from the authoring data model to this intermediate data model, and another translation process, as a second step, towards the presentation data model simple enough to enable reliable converters and efficient and reliable presentation engine implementations.

Following this third alternative, this paper proposes the use of a new NCL profile, called NCL Raw profile, as the intermediate step to support the final conversion to Ginga-NCL internal structures used in a DTV application scheduling. The new profile, designed without almost all syntactic sugar present in the NCL EDTV profile, will allow for using a very simple NCM derived data model, since several of its entities will be removed.

The NCL Raw profile has been designed to accomplish the requirement of having reliable converters for EDTV NCL 3.1 applications (or for the old EDTV NCL 3.0 version) to allow reliable and efficient Ginga-NCL implementations. The profile also takes into account the new features being thought for the NCL 3D profile currently under development. It must be stressed that the profile goal is not the authoring process but to act as an intermediate language within a translation process; although it can be used as an intermediate language or even be the resultant language of authoring tools, as discussed in Section 2.

The remainder of this paper is organized as follows. Section 2 states some additional advantages of using the NCL raw profile as a liaison with other declarative languages and as a transfer syntax notation. Section 3 discusses the syntactic sugars removed from the NCL 3.1 EDTV profile [10] in order to compose the Raw profile. Section 4 briefly presents the two-step converters. Finally, Section 5 is reserved for the conclusions.

2. THE NCL RAW PROFILE AS A TRANSFER SYNTAX NOTATION

Currently, the transfer syntax notation (from the server side to the client side) for DTV applications, both for ITU-T IPTV services and for ISDB-T_B applications, is the same authoring syntax notation that must follow the NCL EDTV profile, in its full capabilities. This means that the two conversion steps towards the Ginga-NCL presentation data model must be done in the client (receiver) side. Figure 1 shows the Ginga-NCL architecture modules [11], including the Converters.

The definition of the NCL Raw profile allows for its use as the basis of a new transfer syntax. In this case, the first step in the conversion process can be done in the server side, and the second one in the receiver side. This approach has several advantages.

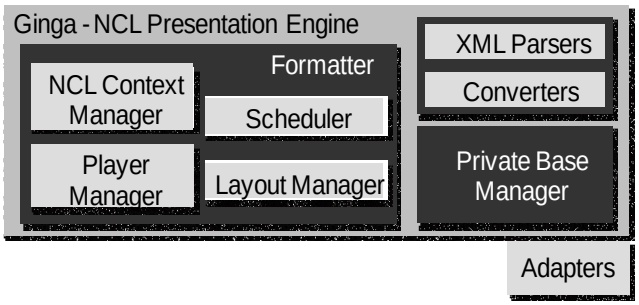

Figure 1. Ginga-NCL architecture.

First, it allows for a simpler interpretation procedure at the client side. Since the client (receiver device) is usually a platform with limited resources, this can be a good advantage.

Second, as the Raw profile is not tailored for authoring, it is less structured, with practically no reuse features and no syntactic sugar. Applications written following the Raw profile are much more difficult to be understood and, thus, to be cloned.

Third and the main one, the Raw profile can be the intermediate notation of converters from other declarative languages. Thus the authoring phase can use languages other than NCL, more tailored to user's flavor, without imposing any additional load to the receiver. Therefore, NCL Raw profile can act as the liaison transfer syntax among several declarative languages developed for hypermedia applications, such as SMIL [2], SVG [12], LASER [4], etc. This is a very interesting point to be worked out in near future.

It is very important to stress closing this section that in order to not leave a legacy, an application following the Raw profile should be able to run in receivers prepared to receive transfer syntax notation based on the EDTV profile. This principle guided the profile design.

3. THE PROFILE SCHEMA

As aforementioned, all syntactic sugars of NCL 3.1 EDTV profile were removed in defining the NCL 3.1 Raw profile. Table 1 presents all NCL 3.1 modules used in the Raw profile. The table shows the attributes and contents that come from the NCL Raw DTV profile, besides those defined in the modules themselves. Element attributes that are required are underlined. In the table, the following symbols are used: (?) optional (zero or one occurrence), (|) or, (*) zero or more occurrences, (+) one or more occurrences. The child element order is not specified in the tables.

The NCL Raw profile schemas can be downloaded from:
http://www.ncl.org.br/NCL3.1/RawProfile

Table 1. Modules used in the Raw profile.

Extended Module	Element	Attributes	Content
Structure	ncl	*id, title, xmlns*	(head?, body?)
	head		(meta*, metadata*)
	body	*id*	(port | property| media | context | link | meta | metadata)*
Media	media	*id, src, refer, type instance*	(area | property)*
Context	context	*id*	(port| property| media| context | link | meta | metadata)*
MediaContentAnchor	area	*id, coords, begin, end, text, position, first, last, label, clip*	empty
CompositeNodeInterface	port	*id, component, interface*	empty
PropertyAnchor	property	*name, value, externable*	empty
Linking	link	*id*	link definition
Meta information	meta	*name, content*	empty
	metadata	*empty*	RDF tree

3.1 NCL Properties

Different from the EDTV profile, media and composite object's properties must be defined using <property> elements. In the Raw profile the Layout, Descriptor, and DescriptorControl modules were removed.

Since all properties must be defined in <property> elements, the *externable* attribute has the important function of defining if the defined interface is able to be used in relationships with other objects or not. In other words, the attribute defines the encapsulation of an NCL object and thus its visibility to external entities.

The TransitionBase and BasicTransition modules were also removed from the EDTV profile to define the Raw profile. Transitions are assumed to be media-object's exhibition properties to be defined in <property> elements. Appendix A presents the reserved words for all predefined properties.

3.2 Reuse and Import Features

The Raw profile does not support any importing facility. Reuse can only be defined referring to entities defined in the same NCL document. This means that the *refer* attribute can only have an id value defined in the same NCL application.

Moreover, syntactic reuse is not allowed. Reuse is only allowed for presentation objects. This means that the *instance* attribute can only receive the "instSame" and "gradSame" values.

3.3 Context Elements

The <context> elements are not essential in a transfer syntax notation, since it does not have a relevant role regarding presentation scheduling tasks performed by Ginga-NCL. However, this element was not removed from the EDTV profile since it supports the application structuring. Structured notations can be useful at the client side if live editing of NCL applications is allowed and if relationship actions may be applied to a set of objects.

The <body> element was kept in the Raw profile only by syntactic reasons to represent the most external context. However, it must be treated as any <context> element.

3.4 Link Elements

NCL 3.1 introduced a new syntax to define <link> elements [10] without referring to <causalConnector> elements. This new syntax is not supported in the NCL 3.0 EDTV profile.

In the Raw profile, the EDTV Connector Facility was removed, as well as all child elements of a <link> element. In the Raw profile the <link> element does not have child elements and may only have the *id* attribute. The <link> element's content is defined as in the NCL 3.1 EDTV profile [10].

Different from the EDTV profile, object alternatives are not chosen from a <switch> element in the Raw profile. Alternatives must be selected during navigation, testing <link> element's causal conditions based on the <property> elements defined in the NCL settings node [9]. Therefore, in the Raw profile the TestRule, TestRuleUse, ContentControl, and SwitcInterface modules were removed.

4. CONVERTERS

The current Ginga-NCL implementation also uses an object-oriented event-driven data model that is nearly a one-to-one translation of the objects of the EDTV-based authoring model. The conversion from one model to the other is done in only one step. As mentioned in Section 1, the adopted approach sacrifices the presentation engine efficiency and reliability, obliging it to perform using a data model moved away from its design goals.

In order to allow some desired features (for example, to start an application from whichever moment in time), other data structures are used, replicating some of the functionalities already provided by the presentation data model. The replications are needed to benefit performance efficiency (in most cases to minimize delays) at a cost of a more complex and thus possibly a less reliable implementation.

All schedule procedures of Ginga-NCL can be based in a Hypermedia Temporal Graph (HTG) [3] from where all other data structures can be derived. The conceptual data model provided by HTG is much more efficient than the current data model regarding

scheduling procedures, and much simpler, and thus with an implementation much more reliable. However, the HTG data model has a much bigger semantic distance from the authoring NCM model than in the current Ginga-NCL presentation model. Therefore, the use of HTG requires more complex converters. Because converters execute at document compile time and because conversions can be carried out in two steps, the use of HTG as the basis of the presentation data model seems to be very promising, besides other advantages already discussed in previous sections.

4.1.1 From the NCL EDTV profile to the Raw profile

The converter between the NCL EDTV profile to the NCL Raw profile can run in the server or client sides. An implementation for the server side will be available in C/C++ open-source code (www.ncl.org.br/tools.html) in the beginning of the second half of 2010.

The embedding of this converter in the client side Ginga-NCL implementation is required, since Ginga-NCL must be able to run NCL applications based on the EDTV profile. The next version of the Ginga-NCL reference implementation is already in course and shall embed this converter.

Listing 1 shows a possible translation of an EDTV profile's <switch> to Raw profile. In the example, the <switch> element is converted into a <context>. The original <media> elements are copied to this <context> and associated <bindRule> elements become <link> elemens that check "settings" properties. It is easy to see that this translation preserves <switch> element semantics.

```
<!-- EDTV profile code -->
<simpleRule id="rA" var="X" comparator="lt" value="1"/>
<simpleRule id="rB" var="X" comparator= "eq" value="1"/>
…
<switch id="S">
  <bindRule rule="rA" constituent="mA"/>
  <media id="mA" src="mA.png"/>
  <bindRule rule="rB" constituent="mB"/>
  <media id="mB" src="mB.png"/>
</switch>

<!-- resulting Raw profile code -->
<context id="S">
  <media id="mA" src="mA.png"/>
  <link id="rA">onBegin S and X lt 1 then start mA end</link>
  <media id="mB" src="mB.png"/>
  <link id="rB">onBegin S and X eq 1 then start mB end</link>
</context>
```

Listing 1. Example of EDTV switch simulation in Raw profile.

4.1.2 From the Raw profile to HTG and scheduling plans

The current implementation of Ginga-NCL is able to convert from the NCL 3.0 EDTV profile to the HTG data structure. This converter shall be modified to accept the new NCL 3.1 <link> element and shall be simplified by removing all NCL 3.1 EDTV elements and attributes that are not used in the NCL Raw profile.

As aforementioned although HTG is included in the current Ginga-NCL implementation, and although it is a better approach to control the required scheduling plans [3], all scheduling performed in the current implementation is performed based on the object-oriented event-driven data model, conceived to benefit the conversion process: A decision at least open to discussion.

The next version of the Ginga-NCL reference implementation already in course shall use HTG in its full advantages to control the presentation. From HTG, all scheduling plans (content loading, content prefetching, media-player loading, QoS negotiation, and presentation plans) shall be created.

4.1.3 From other languages to the Raw profile

If the first conversion step is performed in the server side, Ginga-NCL will be able to run applications developed in other programming languages, once converters for these languages are conceived.

As a proof of concept a converter was developed for the SMIL Tiny profile [2].

5. CONCLUSIONS

The design of the new NCL 3.1 Raw profile is the initial step towards an efficient (in terms of delay, processing, and memory use) and reliable next version of the reference Ginga-NCL implementation.

This work is also the initial step towards the NCL 4.0 version, which will include support for 3D objects, among other features.

The Raw profile is also a step ahead in the liaison between ITU-T and other standardization groups, in especial the W3C (World Wide Consortium), aiming at the integration of NCL, SMIL and SVG technologies, and ISO (International Organization for Standardization), aiming at the integration of NCL and MPEG-4 LASER technologies.

The work in the TeleMídia Lab, where NCL and Ginga-NCL were conceived, on the new presentation engine version is evolving with high priority and we expect to run a complete solution at the beginning of 2011.

6. ACKNOWLEDGMENTS

The authors would like to thank the TeleMídia Lab team who provided a thoughtful discussion of this work, tracked down and fixed problems in the initial new implementation of Ginga-NCL. The authors also thank CNPq, CAPES, MCT and CETIC/RNP for their support.

7. REFERENCES

[1] ABNT NBR Associação Brasileira de Normas Técnicas. *Digital Terrestrial Television Standard 06: Data Codification and Transmission Specifications for Digital Broadcasting, Part 2 – GINGA-NCL: XML Application Language for Application Coding* (São Paulo, SP, Brazil, November, 2007). http://www.abnt.org.br/imagens/Normalizacao_TV_Digital/ABNTNBR15606-2_2007Ing_2008.pdf

[2] Bulterman, Dick C.A., Rutledge, Lloyd W. SMIL 3.0 - Flexible Multimedia for Web, Mobile Devices and Daisy. *Talking Books*. 2nd ed. Springer, 2009. ISBN: 978-3-540-78546-0

[3] Costa R.M.R, Moreno M.F., Soares L.F.G. Intermedia Synchronization Management in DTV Systems. *Proceedings of ACM Symposium on Document Engineering* (Sao Paulo, Brazil, 2008). DocEng 2008, pp. 289-297. ISBN: 978-1-60558-081-4.

[4] Dufourd, J.-C.; Avaro, O.; Concolato, C. An MPEG standard for rich media services. *IEEE Multimedia Journal*. Volume 12, Issue 4, Oct.-Dec. 2005 Page(s): 60 – 68.

[5] ITU-R Recommendation BT-1699. *Harmonization of declarative content format for interactive TV applications*. Geneva, 2009.

[6] ITU-T Recommendation H.760. *Overview of Multimedia Application Frameworks for IPTV*. Geneva, April, 2009.

[7] ITU-T Recommendation H.761. *Nested Context Language (NCL) and Ginga-NCL for IPTV Services*. Geneva, April, 2009.

[8] Soares L.F.G., Rodrigues R.F. Nested Context Model 3.0 Part 1 – NCM Core. *Technical Report. Informatics Department of PUC-Rio*, MCC 18/05. Rio de Janeiro, May, 2005. ISSN 0103-9741.

[9] Soares L.F.G., Rodrigues R.F. Nested Context Language 3.0 Part 8 – NCL Digital TV Profiles. *Technical Report. Informatics Department of PUC-Rio*, MCC 35/06. Rio de Janeiro, October, 2006. ISSN 0103-9741 http://www.ncl.org.br/documentos/NCL3.0-DTV.pdf.

[10] Soares, L.F.G.; Lima, G.F.; Soares Neto, C.S. NCL 3.1 EDTV Profile. Submitted to the *II Workshop on Interactive Digital TV*. Belo Horizonte, MG, Brazil. October 2010.

[11] Soares, L.F.G.; Moreno, M.F.; Soares Neto, C.S.; Moreno, M.F. Ginga-NCL: Declarative Middleware for Multimedia IPTV Services. *IEEE Communications Magazine*. Vol. 48, No. 6, pp. 74-81. June, 2010. ISSN: 0163-6804.

[12] W3C World-Wide Web Consortium. *Scalable Vector Graphics – SVG 1.1 Specification*, W3C Recommendation. 2003. http://www/w3/org/TR/SVG11

APPENDIX A: Predefined Properties for Media Objects

Property name	Meaning
top, left, bottom, right, width, height	Screen positions
location	Screen positions
size	Screen size
bounds	Screen positions
baseDeviceRegion	A region on the screen
deviceClass	Class of secondary devices
plan	Graphical plan
explicitDur	Explicit duration of a media content
background	Background color
visible	Visibility control
transparency	Transparency level
rgbChromakey	RGB color for chromakey
fit	Way to fulfill a region
scroll	Scroll enabling
style	Reference to a style sheet
soundLevel, trebleLevel, bassLevel	Sound control
balanceLevel	Sound control
zIndex	Superposition index
fontColor	Font color
fontAlign	Font Align
fontFamily	Font Family
fontStyle	Font Style
fontSize	Font Size
fontVariant	Font Variant
fontWeight	Font Weight
player	Player identifier for media exhibition content
reusePlayer	Player life cycle control
playerLife	Player life cycle control
moveLeft, moveRight, moveUp, moveDown, focusIndex	Key navigation control properties
focusBorderColor;	Border color for media object in focus
selBorderColor	Border color for media object in selected
focusBorderWidth	Border width for media object in focus
focusBorderTransparency	Border transparency for object in focus

Property name	Meaning
focusSrc, focusSelSrc	Content to be display for object in focus
freeze	Control of the presentation end of a continuous content
transInType	Type of the transition-in
transInSubtype	Subtype of the transition-in
transInDur	Transition-in duration
transInStartProgress	Amount of progress through the transition at which to begin execution
transInEndProgress	Amount of progress through the transition at which to end execution
transInDirection	The direction the transition will run
transInFadeColor	Fade color for the transition-in
transInHorRepeat	Specifies how many times to perform the transition pattern along the horizontal axis
transInVertRepeat	Specifies how many times to perform the transition pattern along the vertical axis
transInBorderWidth	Specifies the border width along a wipe edge
transInBorderColor	Specifies the border color along a wipe edge
transOutType	Type of the transition-out
TransOutSubtype	Subtype of the transition-out
transOutDur	Transition-out duration
transOutStartProgress	Amount of progress through the transition at which to begin execution
transOutEndProgress	Amount of progress through the transition at which to end execution
transOutDirection	The direction the transition will run
transOutFadeColor	Fade color for the transition-out
transOutHorRepeat	Specifies how many times to perform the transition pattern along the horizontal axis
transOutVertRepeat	Specifies how many times to perform the transition pattern along the vertical axis
transOutBorderWidth	Specifies the border width along a wipe edge
transBorderColor	Specifies the border color along a wipe edge