A Formal Study of a Visual Language for the Visualization of Document Type Definition

Jean-Yves Vion-Dury, Emmanuel Pietriga

A Formal Study of a Visual Language for the Visualization of Document Type Definition

Jean-Yves Vion-Dury Emmanuel Pietriga*

Xerox Research Centre Europe
6 Chemin de Maupertuis 38240 Meylan - France
{jean-yves.vion-dury, emmanuel.pietriga}@xrce.xerox.com

Abstract

This formal study proposes a transformational approach to the definition of general purpose visual languages based on hierarchical structures, addressing more specifically DTD visualization as its application area. We show that such visual languages can be constructed through progressive refinement of a syntax based on nested/juxtaposed rectangles. Several transformation stages, which can all be formally characterized, produce a high quality visual representation which expresses the fundamental properties of the original structure. Moreover, this approach opens some perspectives in proving visual properties through standard mathematical tools such as inductive proofs, thus establishing some practical links between visual language theory and classical language theory.

1. Introduction

Document Type Definitions (DTDs) are used to constrain the structure of XML documents (trees). This paper describes formally a visual language for the graphical representation of DTDs, which is used in VXT, a visual programming language specialized in the specification of XML transformations [7].

Section 2 presents the basic graphical object model we consider, and a visual syntax $\mathcal{V}$ adapted to the representation of hierarchical structures, such as structured documents (SGML, XML, HTML) and DTDs. The required properties of this syntax are identified and captured in a pivotal abstract syntax $\mathcal{AS}$ which simplifies formal treatment by abstracting over some spatial issues while still having a visual semantics. The latter is defined through a translation function from the language of $\mathcal{AS}$ into the language generated by $\mathcal{V}$. Section 3 proposes a formal definition of Document Type Definitions and a first visual semantics by the means of a translation that transforms any valid DTD into a sentence of $\mathcal{AS}$. Section 4 describes how such representations can be simplified and made more effective through an additional transformation based on a term rewriting system. This one reduces the number of graphical objects while preserving all the information. A subtle pretty-printing function is then presented in section 5, which enforces the perception of structural information thanks to spatial analogies. The related work section analyses other solutions for the representation of DTDs and more generally tree-based structures, showing how our work differentiates from them. The conclusion synthesizes the paper and discusses global results.

2. Visual Syntax

2.1. Representation model

graphical object model. We first define Gl as an abstract graphical type which does not have any representation. Objects of type Gl have five attributes, namely x, y which represent the coordinates of the object's center, z for the depth of the object (this information is used to determine the order in which to paint objects) and w, h for the width and height. Renderable objects can be of type Sh for shapes or Tx for text strings. Both are subtypes of Gl, the first one defining a shape to draw, the second one a text value. Sh also has a subtype Re for a particular kind of shape : rectangles. Type Sh also defines two additional attributes for color and border style (solid or dashed). For objects of type Sh (which are not of type Re), attributes w and h are computed by respectively projecting them on horizontal and vertical axes. Graphical objects being rendered in an infinite virtual space [7], they can have negative coordinates and their width and height is not limited.

bounding box function. We then define function BB,

*in partnership with INRIA Rhone-Alpes (Projet OPERA), 655 Av de l'Europe, 38330 Monbonnot Saint-Martin - France

which returns the smallest rectangle bounding two objects :

$$BB : Gl \times Gl \rightarrow Re$$

and which is defined as follows :

$$BB(A_1, A_2) = R$$

$with$

$$R.x = (minX + maxX)/2$$
$$R.y = (minY + maxY)/2$$
$$R.w = maxX - minX$$
$$R.h = maxY - minY$$

$where$

$$maxX = max(A_1.x + A_1.w/2, A_2.x + A_2.w/2)$$
$$maxY = max(A_1.y + A_1.h/2, A_2.y + A_2.h/2)$$
$$minX = min(A_1.x - A_1.w/2, A_2.x - A_2.w/2)$$
$$minY = min(A_1.y - A_1.h/2, A_2.y - A_2.h/2)$$

spatial relations. Using this relation, we introduce the following relational constraints, all defined by a 3-tuple $(r\ i\ j)$ where r is a name for the relation, i and j being objects of type Gl or one of its subtypes :

Relation	Arg. types	Definition
$CenteredOn$	$Gl \times Gl$	$(i.x = j.x) \wedge (i.y = j.y)$
$LeftOf$	$Gl \times Gl$	$(i.x < j.x) \wedge (i.y = j.y)$
$Above$	$Gl \times Gl$	$(i.y > j.y) \wedge (i.x = j.x)$
$Over$	$Gl \times Gl$	$(i.z > j.z)$
$Intersects$	$Gl \times Gl$	$(BB(i,j).w < BB(i,i).w + BB(j,j).w) \wedge (BB(i,j).h < BB(i,i).h + BB(j,j).h)$
$Contains$	$Gl \times Gl$	$BB(i,j) = BB(i,i)$

2.2. Syntax definition

We can now introduce our visual language, defined by a grammar which is based on a slightly extended version [1] of the Relational Grammar formalism described by Wittenburg et al [12].

Definition 1 *The grammar is a 5-tuple $\mathcal{V}$ =($\mathcal{N}$,Σ,S,$\mathcal{R}$,P) where :*

- *$\mathcal{N}$ is the following set of nonterminals : $\mathcal{N} = \{N_a, N_b, N_c\}$.*
- *Σ is a set of graphical terminal symbols of type Sh, Re or Tx, disjoint from $\mathcal{N}$.*
- *S is the start symbol in $\mathcal{N}$: $S = N_a$.*
- *$\mathcal{R}$ is the set of relation symbols defined above : $\mathcal{R} = \{Contains, CenteredOn, Above, LeftOf, Intersect\}$.*
- *P is the following set of productions of the form $A \rightarrow \alpha/\beta/F$, where $A \in \mathcal{N}$, $\alpha \in (\mathcal{N}|\Sigma)^+$, β is a set of relational constraints of the form $(r\ i\ j)$ where $r \in \mathcal{R}$ and i, j are integers referencing a member of α (the left-hand-side of a rule is conventionally referenced as 0, the one or more right-hand-side elements are referenced $1..n$ in the order in which they appear in the definition. F is a set of assignement statements of the form $(a\ 0) = (a\ i)$, i referencing a member of α (see [12]):*

$N_a \rightarrow Gl\ Sh$ 1.0
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_a \rightarrow Gl\ Tx$ 1.1
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_a \rightarrow Gl\ Re$ 1.2
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_a \rightarrow Gl\ Re\ N_a$ 1.3
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(CenteredOn\ 3\ 2)\ (Contains\ 2\ 3)\ (Over\ 3\ 2)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_a \rightarrow Gl\ N_b$ 1.4
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_a \rightarrow Gl\ N_a\ N_c$ 1.5
$(Contains\ 1\ BB(2,3))\ (Over\ 2\ 1)\ (Above\ 2\ 3)$
$(CenteredOn\ BB(2,3)\ 1)\ (Over\ 3\ 1)$
$(not\ Intersect\ 2\ 3)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_b \rightarrow Gl\ N_a\ N_b$ 1.6
$(Contains\ 1\ BB(2,3))\ (not\ Intersect\ 2\ 3)$
$(CenteredOn\ BB(2,3)\ 1)\ (Over\ 3\ 1)$
$(LeftOf\ 2\ 3)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_b \rightarrow Gl\ N_a$ 1.7
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_c \rightarrow Gl\ N_a\ N_c$ 1.8
$(Contains\ 1\ BB(2,3))\ (Over\ 2\ 1)$
$(CenteredOn\ BB(2,3)\ 1)\ (Over\ 3\ 1)$
$(Above\ 2\ 3)\ (not\ Intersect\ 2\ 3)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

$N_c \rightarrow Gl\ N_a$ 1.9
$(CenteredOn\ 2\ 1)\ (Contains\ 1\ 2)\ (Over\ 2\ 1)$
$(x\ 0) = (x\ 1), (y\ 0) = (y\ 1), ((z+1)\ 0) = (z\ 1),$
$(w\ 0) = (w\ 1), (h\ 0) = (h\ 1)$

2.3. Visual properties of the syntax

The language defined by this grammar is a superset of the sentences that can be generated from the translation of DTDs. Basically, it allows the nesting at an arbitrary level of graphical objects, including text strings, provided that objects containing other objects are of type Re (i.e rectangles). When a rectangle contains a set of objects, elements of this set can be laid out either horizontally or vertically but cannot overlap even partially, and the set is always centered with respect to the parent rectangle, whose size is defined so as to fully contain its children.

Figure 1 shows two examples of visual sentences allowed by the grammar : all objects and sets of objects are centered with respect to their parent and are fully contained

[1] Some relational constraints reference bounding boxes defined on objects belonging to α. This liberty, with respect to the formalism, is of no consequence since we are not interested in the properties associated with the fomalism : we rely on it only as a descriptive tool and do not intend to use it for parsing.

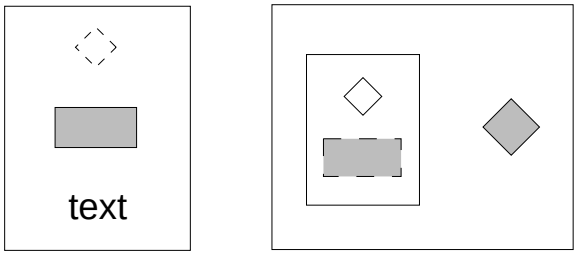


Figure 1. Examples of correct visual sentences

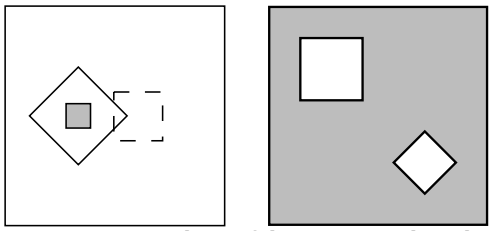


Figure 2. Examples of incorrect visual sentences

within them. Figure 2 illustrates two sentences that do not belong to the language. The left-hand one is incorrect because shapes other than rectangles are not allowed to contain objects and because two sibling shapes partially overlap. In the right-hand one, components of the outmost rectangle are not aligned vertically or horizontally, thus making the sentence incorrect.

Example 1 *the full derivation path of the leftmost sentence in Figure 1* [2] [3] .

$$
\begin{aligned}
&(\{N_a\}, \emptyset)\\
\xrightarrow{(1.3)}\ &(\{Gl_1, Re_2, N_a\} \models \mathcal{R}_1 \cup \{(Contains\ Gl_1\ Re_2),\\
&(CenteredOn\ Re_2\ Gl_1), (CenteredOn\ N_a\ Re_2),\\
&(Over\ N_a\ Re_2), (Contains\ Re_2\ N_a), (Over\ Re_2\ Gl_1)\})\\
\xrightarrow{(1.5)}\ &(\{Gl_1, Re_2, Gl_3, N_a, N_c\} \models \mathcal{R}_2 \cup \{(Above\ N_a\ N_c),\\
&(Contains\ Gl_3\ BB(N_a, N_c)), (Over\ N_a\ Gl_3),\\
&(not\ Intersect\ N_a\ N_c), (Over\ N_a\ Gl_3),\\
&(CenteredOn\ BB(N_a, N_c)\ Gl_3)\})\\
\xrightarrow{(1.0)}\ &(\{Gl_1, Re_2, Gl_3, Gl_4, Sh_5, N_c\} \models\\
&\mathcal{R}_3 \cup \{(Over\ Sh_5\ Gl_4), (CenteredOn\ Sh_5\ Gl_4),\\
&(Contains\ Gl_4\ Sh_5)\})\\
\xrightarrow{(1.8)}\ &(\{Gl_1, Re_2, Gl_3, Gl_4, Sh_5, Gl_6, N_a, N_c\} \models\\
&\mathcal{R}_4 \cup \{(Over\ N_a\ Gl_6), (Contains\ Gl_6\ BB(N_a, N_c)),\\
&(Over\ N_c\ Gl_6), (CenteredOn\ BB(N_a, N_c)\ Gl_6),\\
&(Above\ N_a\ N_c), (not\ Intersect\ N_a\ N_c)\})\\
\xrightarrow{(1.2)}\ &(\{Gl_1, Re_2, Gl_3, Gl_4, Sh_5, Gl_6, Gl_7, Sh_8, N_c\} \models\\
&\mathcal{R}_5 \cup \{(CenteredOn\ Sh_8\ Gl_7), (Contains\ Gl_7\ Sh_8),\\
&(Over\ Sh_8\ Gl_7)\})\\
\xrightarrow{(1.9)}\ &(\{Gl_1, Re_2, Gl_3, Gl_4, Sh_5, Gl_6, Gl_7, Sh_8,\\
&Gl_9, N_a\} \models \mathcal{R}_6 \cup \{(CenteredOn\ N_a\ Gl_9),\\
&(Contains\ Gl_9\ N_a), (Over\ N_a\ Gl-9)\})\\
\xrightarrow{(1.1)}\ &(\{Gl_1, Re_2, Gl_3, Gl_4, Sh_5, Gl_6, Gl_7, Sh_8,\\
&Gl_9, Gl_{10}, Tx_{11}\} \models \mathcal{R}_7 \cup \{(CenteredOn\ Tx_{11}\ Gl_{10}),\\
&(Contains\ Gl_{10}\ Tx_{11}), (Over\ Tx_{11}\ Gl_{10})\})
\end{aligned}
$$

[2] $\mathcal{R}_i$ represents the union of $\mathcal{R}_{i-1}$ with the set of relations specified in the previous derivation step.

[3] We note $S \models R$ the couple (S, R) such that elements in S satisfy all relations in R.

2.4. Abstract syntax

We now introduce a formal abstraction which will ease the subsequent formal treatments and transformations (with underlying visual soundness).

definition. We propose to capture the previous visual properties into an abstract syntax which is not purely visual but rather structural [4]. The tree grammar formalism [9] is a natural extension of CFG grammars allowing the definition of languages as sets of trees instead of sets of strings. This formalism, while powerful in capturing tree-like structures, is also clear and simple enough to understand and supports inductive proof approaches naturally.

Definition 2 (Abstract Syntax $\mathcal{AS}$)

$$
\begin{array}{lcll}
G & \rightarrow & \mathbf{Box}(G, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c}) & \textit{container object} \quad 2.10\\
G & \rightarrow & \mathbf{VA}(G, G, \mathbf{d}) & \textit{Ver. alignment} \quad 2.11\\
G & \rightarrow & \mathbf{HA}(G, G, \mathbf{d}) & \textit{Hor. alignment} \quad 2.12\\
G & \rightarrow & \mathbf{Shp}(\mathbf{f}, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c}) & \textit{terminal shape} \quad 2.13\\
G & \rightarrow & \mathbf{Txt}(\mathbf{v}) & \textit{terminal text} \quad 2.14
\end{array}
$$

where **f** is a meta-variable for various shapes such as (**triangle**, **square**, $\cdots$), **s** denotes style (**dashed**,**solid**), **w**,**h**, **d** are positive numerical variables for width, height and spacing) and **c** represents color. Meta-variable **v** designates the text value. We note $\mathcal{L}(\mathcal{AS})$ the (tree) language generated by $\mathcal{AS}$.

Example 2 (an element of $\mathcal{L}(\mathcal{AS})$) *(captures the structure of the leftmost sentence in Fig. 1)*

$$
\begin{array}{l}
\mathbf{Box}(\\
\quad \mathbf{VA}(\\
\quad\quad \mathbf{Shp}(\mathbf{f}_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2),\\
\quad\quad \mathbf{VA}(\mathbf{Txt}(\mathbf{v}_4), \mathbf{Shp}(\mathbf{f}_5, \mathbf{s}_5, \mathbf{w}_5, \mathbf{h}_5, \mathbf{c}_5), \mathbf{d}_3)\\
\quad , \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0)
\end{array}
$$

We now demonstrate the membership of the previous tree by providing a full derivation path in $\mathcal{AS}$.

Example 3 (The derivation path of Example 2) *the previous tree belongs to $\mathcal{L}(\mathcal{AS})$, as shown by the following derivation path*

$$
\begin{aligned}
G &\rightarrow \mathbf{Box}(G, \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.10)\\
&\rightarrow \mathbf{Box}(\mathbf{VA}(G, G, \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.11)\\
&\rightarrow \mathbf{Box}(\mathbf{VA}(\mathbf{Shp}(\mathbf{f}_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2),\\
&\qquad G, \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.13)\\
&\rightarrow \mathbf{Box}(\mathbf{VA}(\mathbf{Shp}(\mathbf{f}_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2),\\
&\qquad \mathbf{VA}(G, G, \mathbf{d}_3)\\
&\qquad , \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.11)\\
&\rightarrow \mathbf{Box}(\mathbf{VA}(\mathbf{Shp}(\mathbf{f}_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2),\\
&\qquad \mathbf{VA}(\mathbf{Txt}(\mathbf{v}_4), G, \mathbf{d}_3)\\
&\qquad , \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.14)\\
&\rightarrow \mathbf{Box}(\mathbf{VA}(\mathbf{Shp}(\mathbf{f}_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2),\\
&\qquad \mathbf{VA}(\mathbf{Txt}(\mathbf{v}_4), \mathbf{Shp}(\mathbf{f}_5, \mathbf{s}_5, \mathbf{w}_5, \mathbf{h}_5, \mathbf{c}_5), \mathbf{d}_3)\\
&\qquad , \mathbf{d}_1), \mathbf{s}_0, \mathbf{w}_0, \mathbf{h}_0, \mathbf{c}_0) && (2.13)
\end{aligned}
$$

visual semantics. The visual semantics of this abstract syntax is defined by a translation function which produces as output a set of indexed visual objects compatible with the representation model introduced in section 2.

The function accepts items of $\mathcal{L}(\mathcal{AS})$, and returns a set of indexed graphical objects Γ. It computes the value of visual attributes associated to each object through an ordered sequence of a basic linear algebra instruction set. Elements of Γ are indexed sets of graphical objects noted $\gamma = \{O_1, \cdots, O_n\}$ where the meta-variable O ranges over the syntactic categories $\{Gl, Re, Tx\}$. The function also propagates a numerical index i which corresponds to the depth of the current node in the source tree. We use the notation $\gamma[G_k.attr := \cdots]$ for the set computed from γ by applying the operation $O_k.attr := \cdots$ on every element O_k of γ. For instance, if $\gamma = \{Re_1, Re_2\}$ then $\gamma[O_k.x := O_k.x + 10]$ will compute a new set $\gamma' = \{Re_2, Re_3\}$ where all x coordinates are shifted by 10. We consider also $\gamma[O_k.attr := \cdots, O_k.attr := \cdots]$ as a shorthand for $(\gamma[O_k.attr := \cdots])[O_k.attr := \cdots]$. Finally, $BB(\gamma)$ denotes the bounding box obtained from all objects in γ.

Definition 3 ($\mathcal{T}$, Visual translation of $\mathcal{L}(\mathcal{AS})$)

$$\mathcal{T}[\![\ ]\!] : \mathcal{L}(\mathcal{AS}) \times N \rightarrow \Gamma$$

$$\mathcal{T}[\![\textbf{Txt}(\mathbf{v})]\!]_i = \{Tx_i, Gl_i\} \qquad 3.15$$
$$\text{such that} \left[\begin{array}{ll} Tx_i.z = i & Tx_i.x = 0 \\ Tx_i.y = 0 & Tx_i.value = \mathbf{v} \\ Gl_i = BB(Tx_i, Tx_i) & \end{array}\right.$$

$$\mathcal{T}[\![\textbf{Shp}(\mathbf{f}, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c})]\!]_i = \{Sh_i, Gl_i\} \qquad 3.16$$
$$\text{such that} \left[\begin{array}{lll} Sh_i.x = \mathbf{0}, & Sh_i.y = \mathbf{0}, & Sh_i.w = \mathbf{w}, \\ Sh_i.h = \mathbf{h}, & Sh_i.shape = \mathbf{f}, & Sh_i.border = \mathbf{s}, \\ Sh_i.z = i & Sh_i.color = \mathbf{c} & Sh_i.value = \mathbf{v} \end{array}\right.$$
$$\text{and } Gl_i = BB(Sh_i, Sh_i)$$

$$\mathcal{T}[\![\textbf{Box}(G_1, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c})]\!]_i = \gamma_1 \cup \{Re_i, Gl_i\} \qquad 3.17$$
$$\text{with} \left[\begin{array}{ll} \mathcal{T}[\![G_1]\!]_j = \gamma_1, & B_1 = BB(\gamma_1), \\ Re_i.x = B_1.x, & Re_i.y = B_1.y, \\ Re_i.w = \max(\mathbf{w}, B_1.w + \mathbf{2}), & \\ Re_i.h = \max(\mathbf{h}, B_1.h + \mathbf{2}) & \\ Re_i.border = \mathbf{s}, & Re_i.color = \mathbf{c} \\ Re_i.z = i, & Gl_i = BB(Re_i, Re_i) \end{array}\right.$$

$$\mathcal{T}[\![\textbf{HA}(G_1, G_2, \mathbf{d})]\!]_i = \gamma_1 \cup \gamma_2' \cup \{Gl_i\} \qquad 3.18$$
$$\text{with} \left[\begin{array}{lll} \mathcal{T}[\![G_1]\!]_j = \gamma_1, & B_1 = BB(\gamma_1), & \\ \mathcal{T}[\![G_2]\!]_j = \gamma_2, & B_2 = BB(\gamma_2), & Gl_i = BB(B_1, \gamma_2') \end{array}\right.$$
$$\text{and } \gamma_2' = \gamma_2[G_k.y := B_1.y,\ G_k.x := B_1.w + B_2.w + \mathbf{d}]$$

$$\mathcal{T}[\![\textbf{VA}(G_1, G_2, \mathbf{d})]\!]_i = \gamma_1 \cup \gamma_2' \cup \{Gl_i\} \qquad 3.19$$
$$\text{with} \left[\begin{array}{lll} \mathcal{T}[\![G_1]\!]_j = \gamma_1, & B_1 = BB(\gamma_1), & \\ \mathcal{T}[\![G_2]\!]_j = \gamma_2, & B_2 = BB(\gamma_2), & Gl_i = BB(B_1, \gamma_2') \end{array}\right.$$
$$\text{and } \gamma_2' = \gamma_2[G_k.x := B_1.x,\ G_k.x := B_1.h + B_2.h + \mathbf{d}]$$

The following property states that the visual translation produces visual sentences which belong to the language generated by the visual syntax $\mathcal{V}$.

Property 1 (membership of $\mathcal{T}[\![\mathcal{AS}]\!]$.) *for all tree t belonging to $\mathcal{L}(\mathcal{AS})$, $\mathcal{T}[\![t]\!]_0$ belongs to $\mathcal{L}(\mathcal{V})$*

Proof 1 (by induction on the length of derivation paths)
Sketch: the induction hypothesis $\mathcal{H}_{i \geq 1}$ is

for all derivation	$G \xrightarrow{i} t$
there exist $j \geq 1$	$N_1 \models \emptyset \xrightarrow{j} \gamma \models \mathcal{R}$
and a mapping	$\phi : \gamma \rightarrow \mathcal{T}[\![t]\!]_0$
such that	$\mathcal{T}[\![t]\!]_0 \models \mathcal{R}_\phi$,
where	$\mathcal{R}_\phi$ *denotes the transposition of relations in* $\mathcal{R}$ *through* ϕ

We then demonstrate that $\forall i \geq 1,\ \mathcal{H}_1 \wedge \cdots \wedge \mathcal{H}_i \implies \mathcal{H}_{i+1}$. *We just produce one of the 3 cases (***Box***):*

3. Visual representation of DTDs

The interested reader will found the precise specification of DTDs in [1]. For the clarity of our presentation, we use hereafter a simplified version which does not handle attributes even if our implementation does represent them [7].

3.1. Document Type Definitions

Definition 4 (Document Type Definition) *A DTD D_l is a set of rules indexed by a unique name, noted $D_{name_i} = \{name_1 \rightarrow N_1, \cdots, name_n \rightarrow N_n\}$ in which the rule named $name_i$ is considered as the root. The structure N of each rule is defined through the following (abstract) syntax $\mathcal{RH}$:*

$$R \quad \rightarrow \quad R? \mid R^* \mid R^+ \mid (R_1, \ldots, R_n) \mid (R_1 | \ldots | R_n) \mid \# \mid \textit{EMPTY} \mid \textit{ANY} \mid \textit{name}$$

Definition 5 (Valid DTD) *A DTD is valid if all rule names are defined and if no pathological recursive rules are defined, i.e. rules that imply infinite derivation paths. We call $\mathcal{DTD}$ the set of all valid DTDs.*

The XML DTD described here after

```
<!ELEMENT  mail      (sender, subject,textbody)>
<!ELEMENT  sender    (#PCDATA)>
<!ELEMENT  subject   (#PCDATA)>
<!ELEMENT  textbody  (p)+>
<!ELEMENT  p         (#PCDATA | cite)*>
<!ELEMENT  cite      (#PCDATA)>
```

is thus noted

$$\begin{array}{lll} \mathrm{D_{mail}} & = & \{\text{mail}\rightarrow (\text{sender}, \text{subject}, \text{textbody}), \\ & & \text{sender}\rightarrow \#, \\ & & \text{subject}\rightarrow \#, \\ & & \text{textbody}\rightarrow \text{p}+, \\ & & \text{p}\rightarrow (\#|\text{cite})*, \\ & & \text{cite}\rightarrow \# \\ & & \} \end{array}$$

In order to simplify notations we consider DTDs as mappings, and thus for instance, the functional notation $D_{\text{mail}}(p)$ corresponds to $(\# \mid \text{cite})*$.

3.2. Translation Semantics

informal description. We propose a recursive function which computes a visual representation of any legal DTD. Possible recursion and cross-references in rule definitions are handled thanks to contextual information. Each element is processed normally the first time it is encountered in the tree, subsequent references to this element being symbolized by a special graphical object with the element's name as a decoration but no content.

The function is defined over DTDs, sets of labels (rule names) Ψ and $\mathcal{L}(\mathcal{AS})$. Elements of Ψ are sets of labels noted $\psi = \{l_1, \ldots, l_n\}$, memorizing which rules l_i have already been processed (to avoid endless processing of recursive rule definition and to allow the factorization of graphical object translation).

The full signature of the main function is formally defined by:

$$\mathcal{VF}[\![.]\!] : \mathcal{L}(\mathcal{RH}) \times DTD \times \Psi \rightarrow \mathcal{L}(\mathcal{AS}) \times \Psi$$

and the translation of a DTD D_ℓ, noted $\mathcal{VF}[\![D_\ell]\!]$ is recursively defined.

Definition 6 (translation function $\mathcal{VF}$)

$$\mathcal{VF}(D_\ell) = t \text{ with } t, \psi = \mathcal{VF}[\![D_\ell(\ell)]\!]_{D_\ell, \emptyset}$$

$$\mathcal{VF}[\![\#]\!]_{D_\ell,\psi} = \textbf{Shp}(\textbf{lozenge}, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{yellow}), \psi \quad 6.20$$

$$\mathcal{VF}[\![ANY]\!]_{D_\ell,\psi} = \textbf{Box}(\textbf{Txt}(ANY), \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{white}), \psi \quad 6.21$$

$$\mathcal{VF}[\![EMPTY]\!]_{D_\ell,\psi} = \textbf{Shp}(\textbf{rectangle}, \textbf{solid}, \textbf{0}, \textbf{0}, \textbf{void}), \psi \quad 6.22$$

$$\mathcal{VF}[\![(R_1, \cdots, R_n)]\!]_{D_\ell,\psi} = \textbf{Box}(t, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{blue}), \psi_n \quad 6.23$$
$$\textit{with} \begin{cases} t = \textbf{HA}(t_1, t_2, \textbf{2}) \\ t_1, \psi_1 = \mathcal{VF}[\![R_1]\!]_{D_\ell,\psi} \\ \vdots \\ t_i, \psi_i = \textbf{HA}(t'_i, t_{i+1}, \textbf{2}), \\ \textit{and } t'_i, \psi_i = \mathcal{VF}[\![R_i]\!]_{D_\ell,\psi_{i-1}} \\ \vdots \\ t_n = t'_n,\ t'_n, \psi_n = \mathcal{VF}[\![R_n]\!]_{D_\ell,\psi_{n-1}} \end{cases}$$

$$\mathcal{VF}[\![(R_1|\cdots|R_n)]\!]_{D_\ell,\psi} = \textbf{VA}(t, t_2, \textbf{2}), \psi_2 \quad 6.24$$
$$\textit{with} \begin{cases} t = \textbf{Box}(t_1, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{green}) \\ t_1, \psi_1 = \mathcal{VF}[\![R_1]\!]_{D_\ell,\psi} \\ t_2, \psi_2 = \mathcal{VF}[\![(R_2|\cdots|R_n)]\!]_{D_\ell,\psi_1} \end{cases}$$

$$\mathcal{VF}[\![(R)]\!]_{D_\ell,\psi} = \mathcal{VF}[\![R]\!]_{D_\ell,\psi} \quad 6.25$$

$$\mathcal{VF}[\![R?]\!]_{D_\ell,\psi} = \textbf{HA}(t, \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, \textbf{10}, \textbf{10}, \textbf{white}), \textbf{2}), \psi_1$$
$$\textit{with } t = \textbf{Box}(t_1, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{white}) \quad 6.26$$
$$\textit{and } t_1, \psi_1 = \mathcal{VF}[\![R]\!]_{D_\ell,\psi}$$

$$\mathcal{VF}[\![R^*]\!]_{D_\ell,\psi} = \textbf{HA}(t, \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, \textbf{10}, \textbf{10}, \textbf{c}), \textbf{2}), \psi_1$$
$$\textit{with } t = \textbf{Box}(t_1, \textbf{dashed}, \textbf{10}, \textbf{10}, \textbf{white}) \quad 6.27$$
$$\textit{and } t_1, \psi_1 = \mathcal{VF}[\![R]\!]_{D_\ell,\psi}$$

$$\mathcal{VF}[\![R^+]\!]_{D_\ell,\psi} = \textbf{HA}(t, \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, \textbf{10}, \textbf{10}, \textbf{white}), \textbf{2}), \psi_1$$
$$\textit{with } t = \textbf{Box}(t_1, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{white}) \quad 6.28$$
$$\textit{and } t_1, \psi_1 = \mathcal{VF}[\![R]\!]_{D_\ell,\psi}$$

$$\mathcal{VF}[\![name]\!]_{D_\ell,\psi} = \begin{cases} \textbf{VA}(\textbf{Txt}(name), t_1, \textbf{2}), \psi & \textit{if } name \in \psi \\ \textbf{VA}(\textbf{Txt}(name), t_2, \textbf{2}), \psi_1 & \textit{otherwise} \end{cases} \quad 6.29$$
$$\textit{with } t_1 = \textbf{Shp}(\textbf{rectangle}, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{grey})$$
$$\textit{and } t_2 = \textbf{Box}(t, \textbf{solid}, \textbf{10}, \textbf{10}, \textbf{white})$$
$$\textit{and } t, \psi_1 = \mathcal{VF}[\![D_\ell(name)]\!]_{D_\ell,\psi \cup \{name\}}$$

Property 2 $\forall d_m \in \mathcal{DTD},\ \mathcal{VF}[\![d_m]\!] \in \mathcal{L}(\mathcal{AS})$

Proof 2 *We first prove that the function computes a result for every finite DTD (the difficulty is for recursive DTDs), and then we demonstrate that the result belongs to $\mathcal{L}(\mathcal{AS})$.*

Note that the terms generated by $\mathcal{VF}$ are more specific than terms of $\mathcal{AS}$. For instance, no term having the form $\textbf{VA}(\textbf{VA}(G_1, G_2), G_3)$ is produced. We propose to capture these specific structural properties through a tree grammar $\mathcal{DS}$ which is a refinement of $\mathcal{AS}$.

Definition 7 (A more specific abstract syntax $\mathcal{DS}$)

$$\begin{array}{lcll} D & \rightarrow & \textbf{Box}(\textbf{Txt}(\textbf{v}), \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}) & 7.30 \\ D & \rightarrow & \textbf{Box}(H, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}) & 7.31 \\ H & \rightarrow & \textbf{HA}(D, H, \textbf{d}) \mid \textbf{HA}(D, D, \textbf{d}) & 7.32 \\ D & \rightarrow & \textbf{VA}(\textbf{Txt}(\textbf{v}), \textbf{Shp}(\textbf{f}, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}), \textbf{d}) & 7.33 \\ D & \rightarrow & \textbf{VA}(\textbf{Txt}(\textbf{v}), \textbf{Box}(D, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}), \textbf{d}) & 7.34 \\ D & \rightarrow & \textbf{VA}(\textbf{Box}(D, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}), \textbf{d}, V) & 7.35 \\ V & \rightarrow & \textbf{VA}(\textbf{Box}(D, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}), \textbf{d}, V) \mid D & 7.36 \\ D & \rightarrow & \textbf{HA}(\textbf{Box}(D, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c}), \textbf{Shp}(\textbf{f}, \textbf{s}, \textbf{w}, \textbf{h}, \textbf{c})) & 7.37 \end{array}$$

We now prove that $\mathcal{DS}$ is a specialization of $\mathcal{AS}$

Property 3 $\mathcal{L}(\mathcal{DS}) \subseteq \mathcal{L}(\mathcal{AS})$

Proof 3 *We prove for all term t that*

$$D \xrightarrow{*} t \implies G \xrightarrow{*} t$$

(by induction on the length of derivation paths)

And last, we state that $\mathcal{T}$ produces terms of $\mathcal{L}(\mathcal{DS})$

Property 4 *for all valid DTD D_ℓ, $\mathcal{T}[\![D_\ell]\!] \in \mathcal{L}(\mathcal{DS})$*

Proof 4 *By structural induction on terms returned by $\mathcal{T}$*

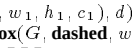
Example 4 (Translations of the mail DTD D_{mail}) *See Figure 3 for $\mathcal{VF}(D_{mail})$ and Figure 4 for $\mathcal{T}[\![\mathcal{VF}(D_{mail})]\!]$.*

4. Simplification

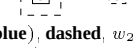
overview. A term rewriting system [3], named $\mathcal{O}$, is used to simplify the visual sentences obtained after translation. It consists of a set of unordered orthogonal rewriting rules, which can for instance be applied using a depth-first, innermost-first strategy. Basically, the simplification erases graphical objects representing cardinality and transfers this information to their children by updating attributes such as the border style (solid or dashed).

Definition 8 *Rewriting system $\mathcal{O}$ over $\mathcal{L}(\mathcal{AS})$*

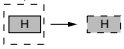
$$\textbf{Box}(\textbf{VA}(\textbf{Txt}(v), \textbf{Shp}(\textbf{rectangle}, \textbf{solid}, w_1, h_1, c_1), d), \textbf{dashed}, w_2, h_2, c_2) \rightarrow \textbf{VA}(\textbf{Txt}(v), \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, w_1, h_1, c_1), d) \quad 8.38$$



$$\textbf{Box}(\textbf{VA}(\textbf{Txt}(v), \textbf{Box}(G, \textbf{solid}, w_1, h_1, c_1), d), \textbf{dashed}, w_2, h_2, c_2) \rightarrow \textbf{VA}(\textbf{Txt}(v), \textbf{Box}(G, \textbf{dashed}, w_1, h_1, c_1), d) \quad 8.39$$



$$\textbf{Box}(\textbf{Box}(H, \textbf{solid}, w_1, h_1, \textbf{blue}), \textbf{dashed}, w_2, h_2, c_2) \rightarrow \textbf{Box}(H, \textbf{dashed}, w_1, h_1, \textbf{blue}) \quad 8.40$$



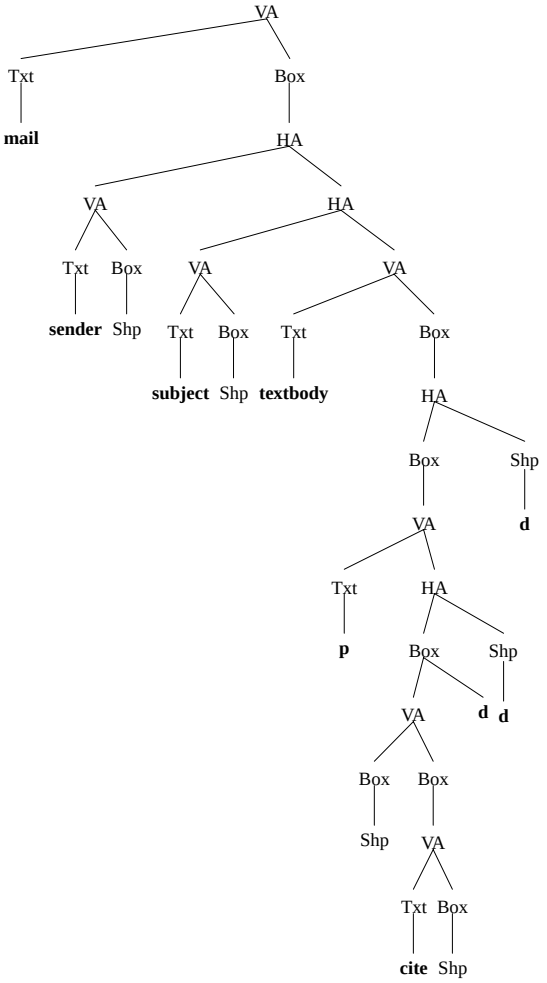


Figure 3. the translation $\mathcal{VF}(D_{mail})$ (simplified leaves, with d = dashed)

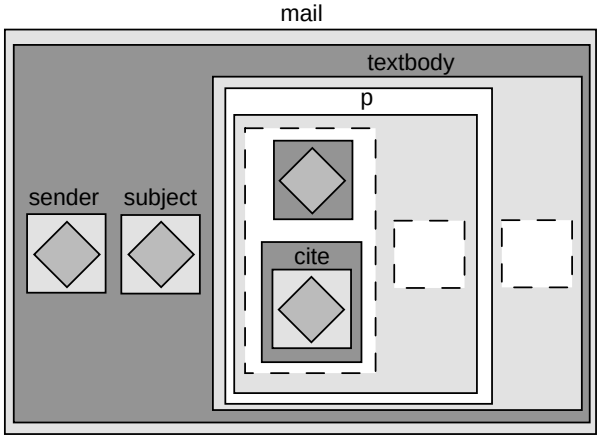
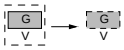
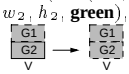


Figure 4. Raw translation of the mail DTD (no optimization, no pretty printing)

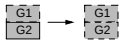
Box(VA(Box(G, solid, w_1, h_1, green), V, d), dashed, w_2, h_2, c_2) → VA(Box(G, dashed, w_1, h_1, green), V, d) 8.41



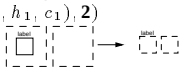
VA(Box(G_1, dashed, w_1, h_1, green), VA(Box(G_2, solid, w_2, h_2, green), V, d_2), d_1) → VA(Box(G_1, dashed, w_1, h_1, green), VA(Box(G_2, dashed, w_2, h_2, green), V, d_2), d_1) 8.42



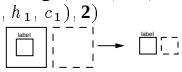
VA(Box(G_1, dashed, w_1, h_1, green), Box(G_2, solid, w_2, h_2, green), d) → VA(Box(G_1, dashed, w_1, h_1, green), Box(G_2, dashed, w_2, h_2, green), d) 8.43



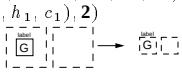
HA(Box(VA(Txt(v), Shp(rectangle, solid, w_1, h_1, c_1), d_1), dashed, w_2, h_2, c_2), Shp(rectangle, dashed, w_2, h_2, c_2), 2) → HA(VA(Txt(v), Shp(rectangle, dashed, w_1, h_1, c_1), d_1), Shp(rectangle, dashed, h_1, h_1, c_1), 2) 8.44



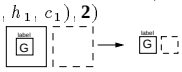
HA(Box(VA(Txt(v), Shp(rectangle, solid, w_1, h_1, c_1), d_1), solid, w_2, h_2, c_2), Shp(rectangle, dashed, w_2, h_2, c_2), 2) → HA(VA(Txt(v), Shp(rectangle, solid, w_1, h_1, c_1), d_1), Shp(rectangle, dashed, h_1, h_1, c_1), 2) 8.45



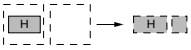
HA(Box(VA(Txt(v), Box(G, solid, w_1, h_1, c_1), d_1), dashed, w_2, h_2, c_2), Shp(rectangle, dashed, w_3, h_3, c_3), 2) → HA(VA(Txt(v), Box(G, dashed, w_1, h_1, c_1), d_1), Shp(rectangle, dashed, h_1, h_1, c_1), 2) 8.46



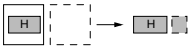
HA(Box(VA(Txt(v), Box(G, solid, w_1, h_1, c_1), d_1), solid, w_2, h_2, c_2), Shp(rectangle, dashed, w_3, h_3, c_3), 2) → HA(VA(Txt(v), Box(G, solid, w_1, h_1, c_1), d_1), Shp(rectangle, dashed, h_1, h_1, c_1), 2) 8.47



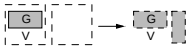
HA(Box(Box(H, solid, w_1, h_1, blue), dashed, w_2, h_2, c_2), Shp(rectangle, dashed, w_3, h_3, c_3), 2) → HA(Box(H, dashed, w_1, h_1, blue), Shp(rectangle, dashed, h_1, h_1, blue), 2) 8.48



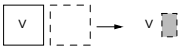
HA(Box(Box(H, solid, w_1, h_1, blue), solid, w_2, h_2, c_2), Shp(rectangle, dashed, w_3, h_3, c_3), 2) → HA(Box(H, solid, w_1, h_1, blue), Shp(rectangle, dashed, h_1, h_1, blue), 2) 8.49



HA(Box(VA(Box(G, solid, w_1, h_1, green), V, d_1), dashed, w_2, h_2, c_2), Shp(rectangle, dashed, w_3, h_3, c_3), 2) → HA(VA(Box(G, dashed, w_1, h_1, green), V, d_1), Shp(rectangle, dashed, w_1, h_1, green), 2) 8.50



HA(Box(V, solid, w_1, h_1, c_1), Shp(rectangle, dashed, w_2, h_2, c_2), 2) → HA(V, Shp(rectangle, dashed, w_2, h_2, green), 2) 8.51



Property 5 (closure over $\mathcal{DS}$) *for all trees t belonging to $\mathcal{L}(\mathcal{DS})$, $\mathcal{O}(t)$ belongs to $\mathcal{L}(\mathcal{DS})$*

Proof 5 *We prove the termination over $\mathcal{L}(\mathcal{AS})$ through a measure of the complexity of any tree $t \in \mathcal{L}(\mathcal{AS})$. We then demonstrate for each rule that if the left-hand side belongs to $\mathcal{L}(\mathcal{DS})$, then the right-hand side belongs to $\mathcal{L}(\mathcal{DS})$*

Example 5 *The optimized mail DTD (see Figure 5)*

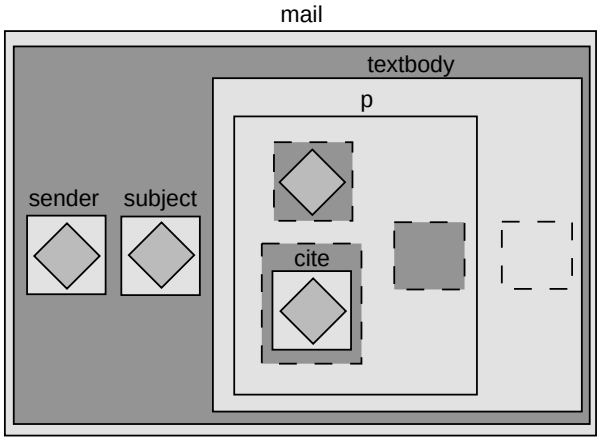


Figure 5. mail DTD, (simplification, but no pretty printing)

5. Pretty-Printing

overview. The principle of the pretty-printing we propose is (i) to establish visual analogies between nodes having the same depth in the tree structure, (ii) to make explicit node types with respect to the DTD semantics (e.g sequences are blue, alternations green). The first point is addressed through a fine tuning of block sizes that avoids pathological cases (e.g too thin rectangles). Figure 6 gives an intuition of the result for the mail DTD. Note that our representation space is based on a camera metaphor, which allows the user to zoom on any part of the observed sentence ([7, 10, 11]).

Definition 9 *Pretty-Printing over simplified terms*

$$\mathcal{P}[\![.]\!] : \mathcal{L}(\mathcal{AS}) \times Int \times Int \rightarrow \mathcal{L}(\mathcal{AS}) \times Int \times Int$$
$$\mathcal{P}[\![\textbf{Shp}(f, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c})]\!]_{W,H} = \textbf{Shp}(f, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c}), \max(W, w), \max(H, h)$$

$$\mathcal{P}[\![\textbf{Box}(G, \mathbf{s}, \mathbf{w}, \mathbf{h}, \mathbf{c})]\!]_{W,H} = \textbf{Box}(G', \mathbf{s}, W', H', \mathbf{c}), W', H'$$
with $G', w', h' = \mathcal{P}[\![G]\!]_{W/1.2, H/1.2}$
and $H' = \max(h' * 1.2, H), \quad W' = \max(w' * 1.2, W, H')$

$$\mathcal{P}[\![\textbf{VA}(\textbf{Box}(G_1, \mathbf{s}_1, \mathbf{w}_1, \mathbf{h}_1, \mathbf{c}_1), \textbf{Box}(G_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2), \mathbf{d})]\!]_{W,H} =$$
$$\textbf{VA}(G''_1, G'_2, \mathbf{0}), w, h$$
with $G'_1, w'_1, h'_1 = \mathcal{P}[\![\textbf{Box}(G_1, \mathbf{s}_1, \mathbf{w}_1, \mathbf{h}_1, \mathbf{c}_1)]\!]_{W,H}$
and $G'_2, w'_2, h'_2 = \mathcal{P}[\![\textbf{Box}(G_2, \mathbf{s}_2, \mathbf{w}_2, \mathbf{h}_2, \mathbf{c}_2)]\!]_{w'_1, h'_1}$
and $G''_1, w, h = \mathcal{P}[\![G'_1]\!]_{w'_2, h'_2}$

$$\mathcal{P}[\![\textbf{VA}(\textbf{Box}(G_1, \mathbf{s}_1, \mathbf{w}_1, \mathbf{h}_1, \mathbf{c}_1), \textbf{VA}(G_2, G_3, \mathbf{d}'), \mathbf{d})]\!]_{W,H} =$$
$$\textbf{VA}(G''_1, G_4, \mathbf{0}), w, h$$
with $G'_1, w'_1, h'_1 = \mathcal{P}[\![\textbf{Box}(G_1, \mathbf{s}_1, \mathbf{w}_1, \mathbf{h}_1, \textbf{green})]\!]_{W,H}$
and $G_4, w_4, h_4 = \mathcal{P}[\![\textbf{VA}(G_2, G_3, \mathbf{d}')]\!]_{w'_1, h'_1}$
and $G''_1, w, h = \mathcal{P}[\![G'_1]\!]_{w_4, h_4}$

$$\mathcal{P}[\![\textbf{VA}(\textbf{Txt}(v), \textbf{Box}(G_1, \mathbf{s}_1, \mathbf{w}_1, \mathbf{h}_1, \mathbf{c}_1), \mathbf{d})]\!]_{W,H} =$$
$$\textbf{VA}(\textbf{Txt}(v), \textbf{Box}(G'_1, \mathbf{s}_1, \mathbf{w}, \mathbf{h}, \mathbf{c}_1), h * 0.1), w, h$$
with $G'_1, w', h' = \mathcal{P}[\![G_1]\!]_{W,H}$
and $w = \max(BB(\textbf{Txt}(\mathbf{v})).w, w' * 1.2)$
and $h = \max(BB(\textbf{Txt}(\mathbf{v})).h, h' * 1.2)$

$$\mathcal{P}[\![\textbf{HA}(G_1, G_2, \mathbf{d})]\!]_{W,H} = \textbf{HA}(G''_1, G'_2, \mathbf{d}), w, h$$
with $G'_1, w'_1, h'_1 = \mathcal{P}[\![G_1]\!]_{W,H}$
and $G'_2, w'_2, h'_2 = \mathcal{P}[\![G_2]\!]_{w'_1, h'_1}$
and $G''_1, w, h = \mathcal{P}[\![G'_1]\!]_{w'_2, h'_2}$

$$\mathcal{P}[\![\textbf{HA}(G_1, \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, \mathbf{w}, \mathbf{h}, \mathbf{c}), \mathbf{d})]\!]_{W,H} =$$
$$\textbf{HA}(G'_1, \textbf{Shp}(\textbf{rectangle}, \textbf{dashed}, w', h'_1, c'), \mathbf{d}), w, h$$
with $G'_1, w'_1, h'_1 = \mathcal{P}[\![G_1]\!]_{W,H}$
and

$$\begin{cases} \text{if } G'_1 = \textbf{VA}(\textbf{Box}(G, s, w_1, h_1, c_1), G, d_1) \\ \quad w' = \min(h_1, w_1), c' = c_1 \\ \text{if } G'_1 = \textbf{Box}(G, s, w_1, h_1, c_1) \\ \quad w' = h'_1, c' = c_1 \\ \text{if } G'_1 = \textbf{VA}(\textbf{Txt}(v), \textbf{Box}(G, s, w_1, h_1, c_1), d_1) \\ \quad w' = h'_1, c' = c_1 \\ \text{otherwise} \\ \quad w' = h'_1, c' = c \end{cases}$$

Property 6 *The pretty-printing of all term in $\mathcal{L}(\mathcal{DS})$ is a term in $\mathcal{L}(\mathcal{DS})$*

Proof 6 *By induction on derivation paths $D \stackrel{i}{\rightarrow} t$*

implementation. The representation obtained thanks to the pretty-printing function is not completely satisfactory. The layout algorithm implemented in VXT [7] is based on another function, too complex to be formally described in this paper. This function further enhances the user's perception of the structure, by assigning the same height to all nodes at the same absolute depth in the DTD tree, without taking into account the choice and sequence nodes, considered as constructors rather than true nodes of the DTD. Choice and sequence nodes are simply assigned a width and a height slightly bigger than their content. The result of this enhanced pretty-printing function is illustrated in Figure 6.

6. Related works

The representation of tree structures as nested graphical elements has been explored by Schneiderman with *treemaps* [5], which represents all nodes as rectangles, the children being nested inside their parent. Treemaps have been used to visualize large hierarchies, such as file systems and other complex tree structures.

Several visual representations of DTDs have been proposed, both in industrial products and research prototypes, but none of these proposals seems to have been specified or studied using formal methods. Near and Far [6], an XML modeling and authoring tool, represents DTDs graphically as an editable node-link diagram. Elements, text nodes and cardinality information are all mapped to different graphical nodes, while sequence and choice are respectively mapped to straight and curved edges, making the structure representation complex and hard to understand, especially when sequences or choices directly contain other sequences and choices. Furthermore, the tree-layout method, combined with very basic navigation and zooming capabilities, makes in our opinion this representation less intuitive and less scalable than the one proposed here.

XML-GL [2], a visual language for querying and restructuring XML data, proposes another visual representation

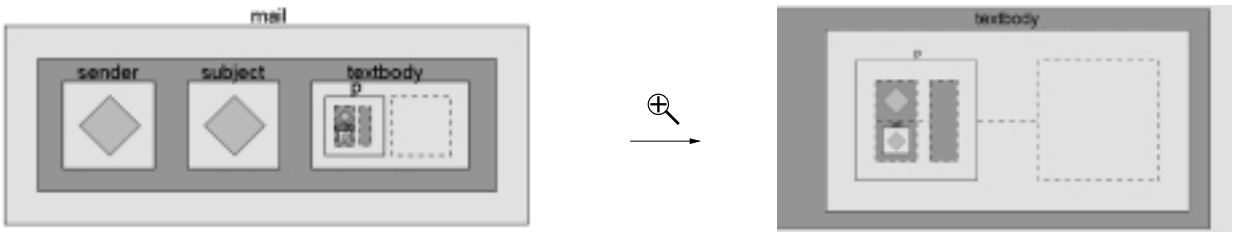


Figure 6. The final pretty-printed mail DTD

of DTDs, still based on node-link diagrams. The graphical data model, called XML-GDM, represents cardinality information as textual attributes on edges representing parent-child relations, and choice by a 'xor' edge crossing all parent-child edges part of the disjunction. XML-GDM supports both DTDs and XML instances. However, the distance between a DTD and associated instances seems bigger in XML-GL than in our representation, and XML-GDM graphs seem less scalable and legible for complex DTDs.

7. Conclusion

Martin Erwig underlined the importance of visual abstract syntaxes [4], and remarked that few papers deal with abstract syntaxes. We add that most of them rely on simplified graph-based syntactic formalisms [8]. In that respect, this paper shows that (i) tree-based visual abstract syntaxes can deal with a general class of visual language syntaxes through well-know proof technics, and (ii) that transformational approaches can help in finding the right abstraction level. The following diagram sums up the transformation architecture developed in this paper. The pivot abstract syntax $\mathcal{AS}$ and its associated translation semantics $\mathcal{VF}$ is quite general and can be used in other studies. We demonstrated through the simplification transformation $\mathcal{O}$ and a pretty-printing transformation $\mathcal{P}$ that the proposed approach produces realistic outputs while enabling formal treatments.

$$\begin{array}{ccccccc} \mathcal{DTD} & \xrightarrow{\mathcal{T}} & \mathcal{L}(\mathcal{DS}) & \xrightarrow{\subseteq} & \mathcal{L}(\mathcal{AS}) & \xrightarrow{\mathcal{VF}} & \mathcal{L}(\mathcal{V}) \\ & & \mathcal{O}\downarrow & & & & \\ & & \mathcal{L}(\mathcal{DS}) & \xrightarrow{\subseteq} & \mathcal{L}(\mathcal{AS}) & \xrightarrow{\mathcal{VF}} & \mathcal{L}(\mathcal{V}) \\ & & \mathcal{P}\downarrow & & & & \\ & & \mathcal{L}(\mathcal{DS}) & \xrightarrow{\subseteq} & \mathcal{L}(\mathcal{AS}) & \xrightarrow{\mathcal{VF}} & \mathcal{L}(\mathcal{V}) \end{array}$$

This work could be extended by proposing *visual soundness properties*, establishing relations between structural properties of DTDs and their effective perception by observers of the visual translation. For instance, showing that all element names of a DTD are visible, and that the parent-child relationship is actually translated by a nesting relation. Another soundness property could rely on operational view of DTDs, showing that if a (document) tree conforms to a DTD, its visual translation (to be defined through the pivot syntax $\mathcal{AS}$) conforms in some sense to the visual translation of its DTD. This should reveal that this essential property is captured by the visual language.

References

[1] T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. eXtensible Markup Language (XML) 1.0 (second edition), October 2000. http://www.w3.org/TR/2000/REC-xml-20001006.

[2] S. Ceri, S. Comai, E. Damiani, P. Fraternali, S. Paraboschi, and L. Tanca. XML-GL: a graphical language for querying and restructuring XML documents. *8th WWW conf.*, 1999.

[3] N. Dershowitz and J.-P. Jouannaud. *Handbook of Theoretical computer Science - Rewrite Systems*, pages 243–320. Elsevier Science Publishers B.V., j. van leeuwen edition, 1990.

[4] M. Erwig. Abstract syntax and semantics of visual languages. *Journal of Visual Languages and Computing*, 1998.

[5] B. Johnson and B. Shneiderman. Treemaps: a space-filling approach to the visualization of hierarchical information structures. *Proceedings of the 2nd International IEEE Visualization Conference, San Diego*, pages 284–291, 1991.

[6] OpenText. Near & far designer, March 2001. http://www.opentext.com/near_and_far/.

[7] E. Pietriga and J.-Y. Vion-Dury. VXT: Visual XML Transformer. *Submitted to the IEEE Symposium on Visual/Multimedia Approaches to Programming and Software Engineering (Human Centric Computing Languages and Environments)*, 2001.

[8] J. Reckers and A. Schürr. Defining and parsing visual languages with layered graph grammars. *Journal of Visual Languages and Computing - Academic Press*, 8:27–55, 1996.

[9] J. W. Thatcher. *Tree Automata: An Informal Survey*, chapter 4, pages 143–172. Prentice-Hall, 1973.

[10] J.-Y. Vion-Dury and F. Pacull. A structured workspace for a visual configuration language. In *Proceedings of VL'97*, Capri, Italy, 1997. IEEE symposium on Visual Languages.

[11] J.-Y. Vion-Dury and M. Santana. Virtual images: Interactive visualization of distributed object-oriented systems. In *OOPSLA'94*, Portland,Oregon, USA, October 1994.

[12] K. Wittenburg and L. Weitzman. Relational grammars: Theory and practice in a visual language interface for process modeling. *AVI '96 International Workshop on Theory of Visual Languages*, May 1996.