Scientific Research Publishing

Journal of Computer and Communications, 2017, 5, 69-82
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

A Stable and Consistent Document Model Suitable for Asynchronous Cooperative Edition

Maurice Tchoupé Tchendji[1], Rodrigue D. Djeumen[2], Marcellin T. Atemkeng[3]

[1]Department of Mathematics and Computer Science, Faculty of Sciences, University of Dschang, Dschang, Cameroon
[2]Department of Mathematics and Computer Science, Faculty of Sciences, University of Douala, Douala, Cameroon
[3]Department of Physics and Electronics, Rhodes University, Grahamstown, South Africa
Email: ttchoupe@yahoo.fr, maurice.tchoupe@univ-dschang.org, djeumenr@yahoo.fr, m.atemkeng@gmail.com

How to cite this paper: Tchoupé, M.T., Rodrigue, D.D. and Atemkeng, M.T. (2017) A Stable and Consistent Document Model Suitable for Asynchronous Cooperative Edition. *Journal of Computer and Communications*, **5**, 69-82.
https://doi.org/10.4236/jcc.2017.58006

Received: May 10, 2017
Accepted: June 27, 2017
Published: June 30, 2017

Abstract

Complex structured documents can be intentionally represented as a tree structure decorated with attributes. Ignoring attributes (these are related to semantic aspects that can be treated separately from purely structural aspects which interest us here), in the context of a cooperative edition, legal structures are characterized by a document model (an abstract grammar) and each intentional representation can be manipulated independently and eventually asynchronously by several co-authors through various editing tools that operate on its "*partial replicas*". For unsynchronized edition of a partial replica, considered co-author must have a syntactic document local model that constraints him to ensure minimum consistency of local representation that handles with respect to the global model. This consistency is synonymous with the existence of one or more (global) intentional representations towards the global model, assuming the current local representation as her/their partial replica. The purpose of this paper is to present the *grammatical structures* which are grammars that permit not only to specify a (global) model for documents published in a cooperative manner, but also to derive automatically via a so call *projection operation*, consistent (local) models for each co-authors involved in the cooperative edition. We also show some properties that meet these grammatical structures.

Keywords

Structured Documents, Documents Models, Grammars, Cooperative Edition, Structured Edition, Projections, Views, Partial Replicas

1. Introduction

With the rise of XML technologies and Web services, structured documents have become important tools for the publication and exchange of information between most often heterogeneous and remote applications. The ever-increasing

power of communication networks in terms of throughput and security as well as efficiency is concern, has revolutionized the way of such documents are edited. Indeed, to the classical model of an author editing his document locally and autonomously, was added the (asynchronous) cooperative editing in which, several authors located on geographically distant sites, coordinate to edit asynchronously the same structured document (**Figure 1**).

Cooperative structured editing is a research field related to *computer-supported cooperative work—CSCW* [1], which Baecker, *et al.* in [2] defined as a set of activities performed on computers and coordinated by a group of collaborative entities. Structured cooperative publishing is a hierarchically organized group publishing work, that operates according to a schedule involving deadlines and task sharing (coordination). When it is asynchronous, each of the participating co-authors in the edition has on its site, a replica of the structured document (intentionally represented as an abstract tree) on which he acts. It is generally preferable for safety reasons[1], efficiency[2], … that this copy is only a *partial replica* of the global document, *i.e.* consisting only of parts of the document containing relevant information related to the considered co-author. In this case, in order to minimize the inconsistencies that can be introduced in the partial replica when locally edited, and to ensure that at the end of edition (or at specific times), the different contributions will be structurally merged [3] [4], each co-author must have on his publishing local site a local document model (a grammar) which is consistent with the global model. Intuitively, a local document model is consistent with respect to the global model, when any partial document *t'* that is conform to him is the partial replica of at least one document *t* conform to the global model.

The central issue addressed in this paper can be simply presented by means of an example of unsynchronized cooperative structured editing process (**Figure 1**). In fact, one can easily imagine an editing process in which several authors work together to produce a pluri-disciplinary book and such that, according to its own field of expertise, everyone contribute to more or less disjointed parts of the same document.

It may be interesting for these authors to specify previously (may be together) the overall hierarchical structure of the document via a grammatical model; we call thereafter *global model* of the document. From it are derive for each of the co-authors a dedicated (local) model called thereafter *local model*. This local model can be regarded as a "*view*" on the global model and obtained by means of a *projection* operation performed on it, which retains on the global model only syntactic categories with a demonstrated interest for the considered author.

For example, **Figure 1** present an overview of the cooperative edition distributed on three sites. Site 1 is dedicated to the edition and the merging of

[1]For a given co-author, some parts of the document may contain sensitive information. It is preferable that he is not even informed of the presence of this information in the document. As we shall see later, the projection operation will resolve this confidentiality problem.

[2]Handled documents pass through the network. They will circulate all the more quickly as their size is reduced. For this reason, the replica of the document to be sent to a co-author must contain only the parts which are of obvious interest to him: it's a partial replica. Here too, the projection operation will solve this concern.

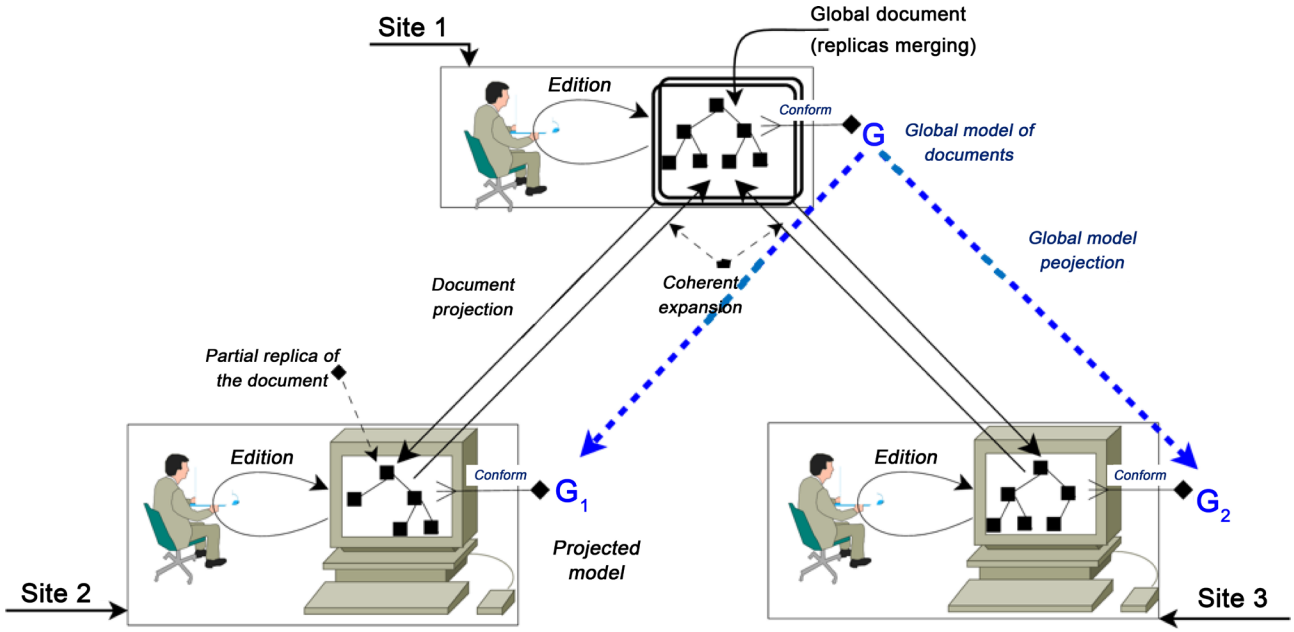

Figure 1. The desynchronized cooperative editing of partial replicas of a structured document.

the (global) document according to the (global) document model G hosted on him. On G, two projections are made to obtain G_1 and G_2, the local models hosted by site 2 and 3 an used for syntactically constrain the desynchronized edition of the partial replicas of the global document on the sites 2 (resp. site 3). Note that, documents published on these sites can be saved (serialized) then restored by parsing. The overall document is subsequently obtained from the site 1 by performing a *consistent expansion*[3] of the various documents published on sites 2 and 3.

The purpose of this paper is to propose a generic document model allowing to specify syntactically both the global model and derived local models, which are consistent with the global model.

In order to do this, we propose the *grammatical structures* (a subset of the extended context free grammars) as well as a *projection* operation which allows to derive from a grammatical structure (global model) and a set of syntactic categories relevant to a given co-author, a local grammatical structure dedicated to him.

Organization of the manuscript: Section 2 presents some concepts and definitions used thereafter. Section 3 presents the *grammatical structures*, the *projection algorithm* on grammatical structures and some features of this model. Section 4 is devoted to the conclusion.

2. Preliminaries

2.1. Extended Context Free Grammars, Documents and Compliances

It is usual to represent the abstract structure of a document by a tree (derivation tree) and its model by an Extended Context Free grammars (ECFG)[4]. In an ECFG, the right member of each production is a regular expression as opposed

[3]The problem of re-synchronization—*consistent expansion*—a posteriori is presented and resolved in [4] where we can also find many basic definitions reused here.

to the sequence of terminal and non-terminal that constitute the right hand side of productions in classical context free grammar. More formally, an extended context free grammar $\mathbb{G}=(S,\mathcal{P})$ is given by a finite set of syntactic categories S, a finite set of production rules $\mathcal{P}$ written as $s\to\mathcal{P}_s$ such that, $s\in S$ and $\mathcal{P}_s$ is a regular expression defined on S.

The dependency graph $D_{\mathbb{G}}$ of grammar $\mathbb{G}$ is a graph whose set of node tags is included in S and, for all rules $s\to\mathcal{P}_s$ in $\mathcal{P}$, there is an arrow from s to b, for all b in a word belonging to the language denoted by $\mathcal{P}_s$ and termed $\mathscr{L}(\mathcal{P}_s)$. An ECFG is said to be *non recursive* if and only if $D_{\mathbb{G}}$ is *acyclic*, and *recursive* if not.

A document t conforms to a grammar $\mathbb{G}$ and we write $t\therefore\mathbb{G}$, if it is a derivation tree of this grammar: it's the case if for any t node n labeled $s\in S$ and with children nodes $n_1,\cdots,n_m$ labeled respectively $s_1,\cdots,s_m$, $s_i\in S$, the word $s_1\cdots s_m\in\mathscr{L}(\mathcal{P}_s)$.

2.2. View, Projection, Partial Replica and Consistency

The derivation tree giving a (global) representation of a structured document published cooperatively, makes visible all the grammar's grammatical symbols. As mentioned in Section 1 above, a coauthor handling such a document using a structured dedicated editor of his area of expertise, do not necessarily have access to all of these grammatical symbols; only a subset of them correspond to syntactic categories perceptible as such by this tool: hence the notion of "*view*" [4]. A view $\mathcal{V}$, is a subset of grammatical symbols ($\mathcal{V}\subseteq S$). Intuitively, they are symbols associated with visible syntactic categories in the considered representation (derivation tree).

Each *view* $\mathcal{V}$ is associated with a *projection* operation noted $\pi_{\mathcal{V}}(t)$, on derivation trees t which erases nodes labeled by invisible symbols while retaining the subtree structure. *Partial replication* is the result of the projection of a document (derivation tree) with respect to a given view. For example, in the **Figure 2** from the global document t in the center, and views $V_1=\{A,B\}$ and $V_2=\{A,C\}$ on the alphabet $S=\{A,B,C\}$, we have on the left the partial replica $t_{V_1}=\pi_{V_1}(t)$, and on the right the partial replica $t_{V_2}=\pi_{V_2}(t)$.

The edition type considered in this paper is asynchronous. On a site i hosting a document model $\mathbb{G}_{\mathcal{V}_i}$ on which a partial replica t_{V_i} is updated with $t_{V_i}\therefore\mathbb{G}_{\mathcal{V}_i}$, we will say that t_{V_i} is *consistent* vis-a-vis a global model $\mathbb{G}$, and we write $t_{\mathcal{V}_i}\trianglelefteq\mathbb{G}$ if and only if a document $t\therefore\mathbb{G}$ exists and $t_{\mathcal{V}_i}=\pi_{\mathcal{V}_i}(t)$. Also, a local model $\mathbb{G}_{\mathcal{V}_i}$ is consistent vis-a-vis a global model $\mathbb{G}$ if and only if $\forall t_i\therefore\mathbb{G}_{\mathcal{V}_i},\exists t\therefore\mathbb{G}$ such that $t_i=\pi_{\mathcal{V}_i}(t)$.

2.3. Some Definitions and Notations

Let $\mathbb{G}=(S,\mathcal{P})$ be an extended context free grammar, $X_i,X_j\in S$, $\alpha\in S^*$, $\mathcal{V}\subseteq S$ a view, t a derivation tree for $\mathbb{G}$ $(t\therefore\mathbb{G}),D_{\mathbb{G}}$ the dependency graph of $\mathbb{G}$ and $p\in\mathcal{P}$ a production rule.

[4]The DTD (Document Type Definition) for example are special cases of extended context free grammars verifying property of *one-unambiguity* [5].

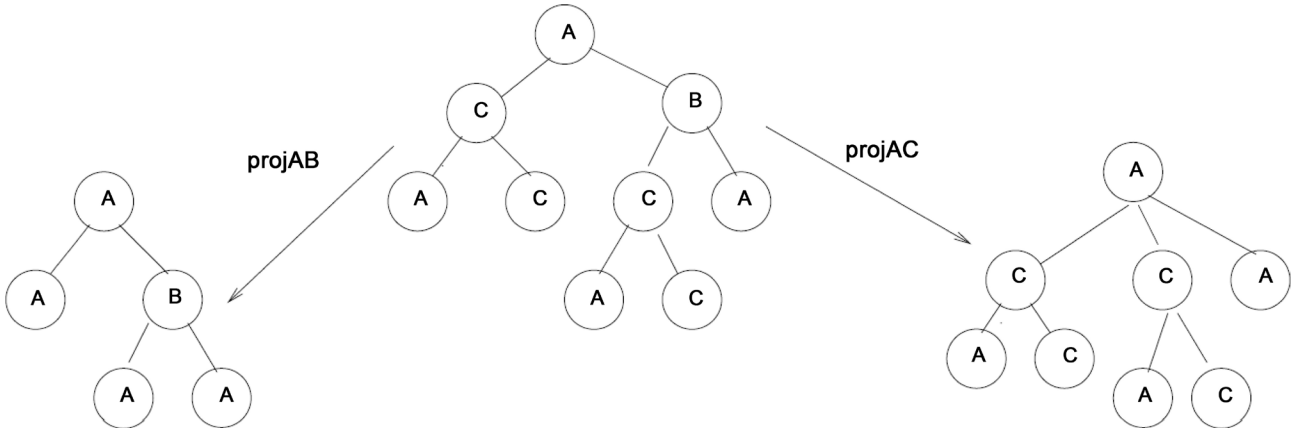

Figure 2. One document (center) and two partial replicas obtained by projections.

$\mathbb{G}$ is said to be finite type if and only if $D_{\mathbb{G}}$ is non recursive. $\mathbb{G}$ is said to be finite type with respect to $\mathcal{V}$ if the restriction of dependency graph $D_{\mathbb{G}}$ on symbols which belongs to $\mathcal{V}$ is not recursive.

We note $nt(t) \subseteq S$ the t's set nodes labels, and $root(t)$ the t's root node label.

The notation " $p@X \to \alpha$ " means that "p has the form $X \to \alpha$ ". We introduce function $lhs(p)$ (resp. $rhs(p)$) which returns the symbol (resp. the symbols) in left hand side (resp. right hand side) of his argument p, a production rule. For example, if $p@X_0 \to X_1 \cdots X_n$, $lhs(p) = X_0$ (resp. $rhs(p) = \{X_1, \cdots, X_n\}$). Also, $p[X_1/\alpha_1, \cdots, X_n/\alpha_n]$ means the substitution in the right hand side of p of all occurrences of each symbols $X_i \in S$ by the corresponding α_i. For example, with $p: X_0 \to aX_1bX_1cX_5$, $\alpha_1 = X_2X_3$, $p[X_1/\alpha_1] = \{X_0 \to aX_2X_3bX_2X_3cX_5\}$.

$\mathscr{L}(\mathbb{G}, A_i)$ is the language generated by grammar $\mathbb{G}$ from symbol $A_i \in S$.

3. A Document Model Stable by Projection Operation, for Cooperative Asynchronous Edition

In this section, we present *grammatical structures* which are a particular form of non-recursive extended context free grammars (ECFG). Indeed, to make the projection (defined below, Section 3.2) possible, it is not permitted to have in this model, recursive grammar symbols[5]. The *grammatical structures* will then be models for documents of bounded depths (consequence of the non-recursivity of the symbols) but of unbounded widths. Moreover, they will allow to specify in a homogeneous way both the global model for the global document and the local models for its various partial replicas.

3.1. Defining (Abstract) Grammatical Structures

A grammatical structure $\mathbb{G} = (S, \mathcal{P})$ is given as:

- a set S of non recursive grammatical symbols, and
- a set $\mathcal{P}$ of production rules. Each rule in $\mathcal{P}$ has one of the two forms:
  - $p@A_0 \to A_1 \cdots A_n$ (classical form of context free grammars rules),

[5]As in [6], we are just interested by non recursive models. This is not an aberration because, from statistical point of view, non recursive DTDs are more frequent than recursive ones.

– $q@A \to B^*$ (*i.e.* A is build up by a list of B)

We recall that an equivalent ECFG can be evidently be derived from a grammatical structure.

3.2. Projection of a Grammatical Structure

Let $\mathbb{G} = (S, \mathcal{P})$ be a grammatical structure, $\mathcal{V} \subseteq S$ a view; let also $\overline{\mathcal{V}} = S \setminus \mathcal{V}$ be the complementary of $\mathcal{V}$ in S. The view $\mathcal{V}$ projection on $\mathbb{G}$, termed $\pi_{\mathcal{V}}(\mathbb{G})$ is the grammatical structure $\mathbb{G}_{\mathcal{V}} = (S_{\mathcal{V}}, \mathcal{P}_{\mathcal{V}})$ where:

- $\mathcal{P}_{\mathcal{V}}$ is obtained from $\mathcal{P}$ by successive rewriting of symbols in $\overline{\mathcal{V}}$ in terms of those in $\mathcal{V}$, then, by substituting properly the result (of this rewriting) in the subset of rules $\mathcal{P}$ having symbols in $\mathcal{V}$ on the left hand side: $\forall p \in \mathcal{P}_{\mathcal{V}}, p@X_0 \to X_1 \cdots X_n$ or $p@X_0 \to X_1^*, X_i \in S_{\mathcal{V}}, 0 \le i \le n$.
- $S_{\mathcal{V}} = \mathcal{V} \cup S^{new}$: syntactic categories of the projected grammar contains symbols of the view with enventually new symbols introduce for *structuring* purpose belonging to set S^{new}. As the process of obtaining the production rules of the projected model proceed by successive rewriting of symbols which did not belong to the view, it can occur during the rewriting process of some symbols that, new symbols being added for format purpose (or decomposition) in order to bring some rules back to the form of the production rules adopted for the grammatical structures[6] (cf. Section 3.1).

The algorithm for deriving $\mathcal{P}_{\mathcal{V}}$ and $S_{\mathcal{V}}$ proceeds in two steps:

Step 1: consider the subset $Prod_{\overline{\nu}} \subseteq \mathcal{P}$ of $\mathbb{G}$'s rules which left hand side does not belongs to the view $\left(Prod_{\overline{\nu}} = \left\{p \in \mathcal{P}, lhs(p) \in \overline{\mathcal{V}}\right\}\right)$ and transform them by successive rewriting to rules like $X' \to \beta$, an acceptable rule of grammatical structure, with $X' = lhs(p)$, and β containing only $S_{\mathcal{V}}$ symbols. Hence $P_{\overline{\nu}}$ set is given as:

$$\begin{aligned} P_{\overline{\nu}} = \Big\{ & A \to \mu_1 \mid \cdots \mid \mu_n, A \in \overline{\mathcal{V}} \Big\} \text{ with : } \{\mu_1, \cdots, \mu_n\} = \Big\{ u = v_1 w_1 v_2 \cdots v_k w_k v_{(k+1)}, \\ & \exists p@A \to v_1 A_1 v_2 \cdots v_k A_k v_{(k+1)}, v_i \in \mathcal{V}^*, A_i \in \overline{\mathcal{V}}, w_i \in \mathscr{L}(\mathbb{G}_{\overline{\nu}}, A_i) \Big\}. \end{aligned} \quad (1)$$

Indeed, $P_{\overline{\nu}}$ can be considered as production rules of a concrete context free grammar $\mathbb{G}_{\overline{\nu}}$ with $\overline{\nu} = \overline{\mathcal{V}} \cup S^{new}$ as non terminal symbols and $\mathcal{V}$ as terminal symbols; then $\mathbb{G}_{\overline{\nu}} = \left(\overline{\mathcal{V}} \cup S^{new}, \mathcal{V}, P_{\overline{\nu}}\right)$.

From Equation (1), one easily deduces that $P_{\overline{\nu}}$ is in fact the union of the rewriting of the productions of $\mathbb{G}$ having a symbol belonging to $\overline{\mathcal{V}}$ in her left hand side. Thus, for every symbol X_i' belonging to $\overline{\mathcal{V}}$, if we note $P_{\overline{\nu}}(X_i')$ the set obtained by rewriting rules of $\mathbb{G}$ having $X_i' \in \overline{\mathcal{V}}$ as left hand side, we have $P_{\overline{\nu}} = \cup P_{\overline{\nu}}(X_i')$ with $X_i' \in \overline{\nu}$. Recall that, symbols in $\mathcal{V}$ are considered as terminal symbols when rewriting.

Algorithm 1 describes the construction process of $P_{\overline{\nu}}(X_i'), X_i' \in \overline{\nu}$. Let's emphasis that, for effective construction of $P_{\overline{\nu}} = \cup P_{\overline{\nu}}(X_i'), X_i' \in \overline{\nu}$, the different

[6]For example a form of rule like $p : X_0 \to X_1^* X_2$ can be obtained after successive rewriting of a rule; this is not an acceptable form of rule. So a new restructuring symbol X^1 is created and rule p is decompose in two new rules as follow $p_1 : X_0 \to X^1 X_2$ and $p_2 : X^1 \to X_1^*$.

Input : $\mathcal{P}$: set of production rules of the grammatical structure $\mathbb{G}$;
$\mathcal{V}$: view;
$X'_i \in \bar{\mathrm{v}}$: a symbol not belonging to the view;
Output: $P_{\bar{\mathrm{v}}}(X'_i)$: rules obtained by view $\mathcal{V}$ projection on rules of $\mathcal{P}$ with X'_i in left hand side.

1 $P_{\bar{\mathrm{v}}}(X'_i) \leftarrow \Phi$; /* initialization to empty set */
2 **for All** $p \in \mathcal{P}$; $p@X'_i \rightarrow \beta$ **do**
3 **if** $p@X'_i \rightarrow X_1^*$ **then**
4 **if** $X_1 \in \mathcal{V}$ **then**
5 $P_{\bar{\mathrm{v}}}(X'_i) = P_{\bar{\mathrm{v}}}(X'_i) \cup \{p\}$; /* p is in the sought format */
6 **else** /* $X_1 \notin \mathcal{V}$: replace properly X_1 in p by α */
7 $P_{\bar{\mathrm{v}}}(X'_i) = P_{\bar{\mathrm{v}}}(X'_i) \cup \{p[X_1/\alpha], \alpha \in rhs(p'), p' \in P_{\bar{\mathrm{v}}}(X_1)\}$;
8 **endif**
9 **else** /* $p@X'_i \rightarrow X_1 \cdots X_n$ */
10 we let $rhs(p) = \beta_0 X'_1 \beta_1 X'_2 \ldots \beta_{m-1} X'_m \beta_m$ with $\beta_i \in \mathcal{V}^*$, $X'_i \in \overline{\mathcal{V}}$, $0 \le i \le m$
/* replace properly the X'_i by corresponding α'_i */
11 $P_{\bar{\mathrm{v}}}(X'_i) = P_{\bar{\mathrm{v}}}(X'_i) \cup \{p[X'_1/\alpha'_1 \ldots X'_m/\alpha'_m],\ \alpha'_i \in rhs(p'),\ p' \in P_{\bar{\mathrm{v}}}(X'_i)\}$;
12 **endif**
13 **endfor**

Algorithm 1. Construction of $P_{\bar{v}}(X'_i) = \{p : X'_i \rightarrow \beta, \forall X_j \in \beta, X_j \in \mathcal{V}\}$.

sets $P_{\bar{v}}(X'_i), X'_i \in \bar{v}$ should be built according to the topological sorting of the X_i dependency graph: a symbol is evaluated after evaluation of symbols from which it depends.

Step 2: Consider the subset $Prod_v \subseteq \mathcal{P}$ of $\mathbb{G}$'s rules, with view symbols in left hand side ($Prod_v = \{p \in \mathcal{P}, lhs(p) \in \mathcal{V}\}$ [7]); for every rule in this set, replace all occurrences of $\overline{\mathcal{V}}$ elements in right hand side, by their right hand side counterpart in $P_{\bar{v}}$, this by all means; we finally obtain the set $\mathcal{P}_{\mathcal{V}}$ of production rules of the projected grammatical structure.

$$\mathcal{P}_{\mathcal{V}} = \{A \rightarrow \mu_1 | \cdots | \mu_n,\ A \in \mathcal{V}\} \text{ with : } \{\mu_1, \cdots, \mu_n\} = \{u = v_1 w_1 v_2 \cdots v_k w_k v_{(k+1)},$$
$$\exists p@A \rightarrow v_1 A_1 v_2 \cdots v_k A_k v_{(k+1)}, v_i \in \mathcal{V}^*, A_i \in \overline{\mathcal{V}}, w_i \in \mathscr{L}(\mathbb{G}_{\bar{v}}, A_i)\}. \quad (2)$$

As for $P_{\bar{v}}$ (Equation (1)), we deduce from Equation (2) that $P_{\bar{v}}$ is the reunion of the sets obtained by rewriting the productions of $\mathbb{G}$ having symbols X_i belonging to $\mathcal{V}$ in their left hand side, by using $P_{\bar{v}}$; that sets is denoted $\mathcal{P}_v(X_i)$. Thus, $\mathcal{P}_{\mathcal{V}} = \cup \mathcal{P}_v(X_i)$ with $X_i \in \mathcal{V}$. The construction of $\mathcal{P}_{\mathcal{V}}(X_i), X_i \in \mathcal{V}$ is described in **Algorithm 2** below.

Algorithm 3 purpose is to construct $p[X'_1/\alpha'_1, \cdots, X'_m/\alpha'_m]$. It explicitly presents when *restructuring symbols* are created (line 5) and when they are explicitly used (line 5 and line 8) in generated productions rules.

3.3. Grammatical Structures Properties

Let $\mathbb{G}$ be a grammatical structure, and $\mathcal{V}$ a view; $\mathbb{G}$ satisfies properties below:

Property 1: $\pi_{\mathcal{V}}(\mathbb{G})$ is a grammatical structure (stability property); this property is guaranteed by **Algorithm 3**.

[7] Note that $Prod_{\bar{v}} \cup Prod_v = \mathcal{P}$.

Input : $\mathcal{P}$: set of production rules of the grammatical structure $\mathbb{G}$;
$\mathcal{V}$: view;
$X_i \in \mathcal{V}$: A symbol belonging to the view;
Output: $\mathcal{P}_{\mathcal{V}}(X_i)$: rules obtained by view $\mathcal{V}$ projection on rules of $\mathcal{P}$ with X_i in left hand side;

$P_{\mathcal{V}}(X_i) \leftarrow \Phi$;
for All $p \in \mathcal{P}$; $p@X_i \to \beta$ **do**
  if $p@X_i \to X_1^*$ **then**
    if $X_1 \in \mathcal{V}$ **then**
      $P_{\mathcal{V}}(X_i) = P_{\mathcal{V}}(X_i) \cup \{p\}$;
    else /* $X_1 \notin \mathcal{V}$: replace properly X_1 in p by α */
      $P_{\mathcal{V}}(X_i) = P_{\mathcal{V}}(X_i) \cup \{p[X_1/\alpha], \alpha \in rhs(p'),\ p' \in P_{\bar{\mathcal{V}}}(X_1)\}$;
    endif
  else /* $p@X_i \to X_1 \cdots X_n$ */
    we let $rhs(p) = \beta_0 X_1' \beta_1 X_2' \ldots \beta_{m-1} X_m' \beta_m$ with $\beta_i \in \mathcal{V}^*$, $X_i' \in \overline{\mathcal{V}}$, $0 \le i \le m$
    /* replace properly X_i' by corresponding α_i' */
    $P_{\mathcal{V}}(X_i) = P_{\mathcal{V}}(X_i) \cup \{p[X_1'/\alpha_1', \ldots, X_m'/\alpha_m'],\ \alpha_i' \in rhs(p'),\ p' \in P_{\bar{\mathcal{V}}}(X_i')\}$;
  endif
endfor

Algorithm 2. Construction of $\mathcal{P}_{\mathcal{V}} = \left\{p : X \to \alpha, X \in \mathcal{V}, \forall X_i \in \alpha, X_i \in \mathcal{V} \cup S^{new}\right\}$.

Input : p: a production rules of the inputted grammatical structure $\mathbb{G}$;
$\{X_1, \ldots, X_m\} \subseteq \overline{\mathcal{V}}$: set of symbols which does not belong to the view;
Output: $p[X_1'/\alpha_1', \ldots, X_m'/\alpha_m']$: rules obtained by substituting X_i in p by α_i;

if $p@X_i^*$ **then**
  /* indeed, we must have $m = 1$ */
  if $\alpha_1 @ X_k^*$ **then**
    return $\{X_1 \to \alpha_1\}$
  else /* $\alpha_1 @ Y_1 \ldots Y_n$ */
    return $\{X_1 \to X^{new1*},\ X^{new1} \to \alpha_1\}$
  endif
else /* $p@X \to \beta_1 X_1 \alpha_2 X_2 \ldots \alpha_{m-1} X_m \alpha_m$ */
  return $\{X \to \beta_1 X_1' \beta_2 \ldots \beta_{m-1} X_m' \beta_m\} \cup \{X^{new_i} \to \alpha_i,\ \alpha_i @ Y^*\}$ in which $X_i' = \alpha_i$ if $\alpha_i @ Y_1 \ldots Y_n$ or $X_i' = X^{new_i}$ if $\alpha_i @ Y^*$ indeed, the production $X^{new_i} \to \alpha_i$ is also outputted.
endif

Algorithm 3. Construction of $\left\{p\left[X_1'/\alpha_1', \cdots, X_m'/\alpha_m'\right], \alpha_i' \in rhs\left(p'\right), p' \in P_{\bar{\mathcal{V}}}\left(X_i'\right)\right\}$.

Property 2: if $t \therefore \mathbb{G}$ then $\left(\pi_{\mathcal{V}}(t) \therefore \pi_{\mathcal{V}}(\mathbb{G})\right)$.

Property 3: if $t'_{\mathcal{V}}$ is a local update of a replica $t_{\mathcal{V}}$ such that $t'_{\mathcal{V}} \therefore \pi_{\mathcal{V}}(\mathbb{G})$, then $\left(\exists t \therefore \mathbb{G}, t'_{\mathcal{V}} = \pi_{\mathcal{V}}(t)\right)$ (consistency property).

We present below, the proof of the Property 2. The proof of Property 3 can be obtained from the proof of Theorem 3.3 given in [7].

Proof. Let $t \therefore \mathbb{G}$ be such that $\pi_{\mathcal{V}}(t) = t_{\mathcal{V}}$; let's show that $t_{\mathcal{V}} \therefore \pi_{\mathcal{V}}(\mathbb{G})$.

In order to do this, if we consider an internal node n of $t_{\mathcal{V}}$ labeled A_i, with its k children, $n_1, \cdots, n_k$ labeled $A_1, \cdots, A_k$; it suffices to show that the word $A_1 \cdots A_k$ belongs to the language denoted by the grammar $\mathbb{G}_{\mathcal{V}}$, admitting the symbol A_i axiom *i.e.* $A_1 \cdots A_k \in \mathscr{L}\left(\mathbb{G}_{\mathcal{V}}, A_i\right)$.

Note that one can define a partition $\Pi = \mathcal{V} \cup \left(S \setminus \mathcal{V} = \overline{\mathcal{V}}\right)$ of S so that, every tree $t \therefore \mathbb{G}$ (**Figure 3(a)**) can be uniquely partitioned into a finite set of maximal subtrees $t_1, t_2, \cdots, t_n$ (**Figure 3(b)**) such as, for any subtree $t_i, 1 \le i \le n$ of the partition, either $nt(t_i) \subseteq \mathcal{V}$, and the labels of the successor nodes of the

leaf nodes of t_i in t if they exist do not belong to $\mathcal{V}$, or $nt(t_i) \subseteq (S \setminus \mathcal{V})$ and the labels of the successor nodes of the leaf nodes of t_i in t if they exist belong to $\mathcal{V}$. When $nt(t_i) \subseteq \mathcal{V}$, we say that t_i is of type $t^{\mathcal{V}}$ and when $nt(t_i) \subseteq (S \setminus \mathcal{V})$, we say that it is of *type* $t^{\bar{\mathcal{V}}}$.

Considering the decomposition of t into subtrees of *type* $t^{\bar{\mathcal{V}}}$ and $t^{\mathcal{V}}$ as described above (Figure 3), a node of t can be found either in a subtree of type $t^{\mathcal{V}}$ or in a subtree of type $t^{\bar{\mathcal{V}}}$. Moreover, by focusing on a node n of $t_{\mathcal{V}}$ and his children $n_1, \cdots, n_k$, they can either: 1) all belong to the same subtree of type $t^{\mathcal{V}}$ (Figure 4) or, 2) belong to different subtrees of type $t^{\mathcal{V}}$ in t; in this case, n is a leaf in the subtree in which it appears, and the $A_i, i = 1, \cdots, k$ are labels of the root nodes (Figure 5) of other subtrees of type $t^{\mathcal{V}}$ or, 3) n and some of his children are in the same subtree and the other are each one in their own subtree (Figure 6). Three case studies are therefore to be considered.

Case 1: $n, n_1, \cdots, n_k$ belong to the same subtree t_j such that $nt(t_j) \subseteq \mathcal{V}$. In this case, according to the construction algorithm of $P_{\mathbb{G}_{\mathcal{V}}}$, $A_1 \cdots A_k \in \mathscr{L}(\mathbb{G}, A_i)$ and therefore to $\mathscr{L}(\mathbb{G}_{\mathcal{V}}, A_i)$.

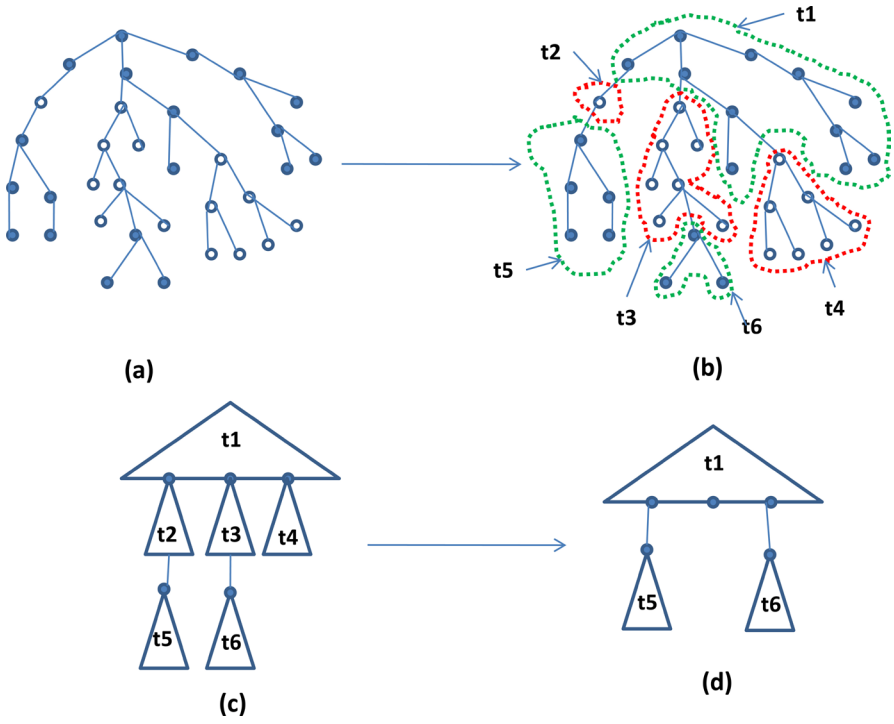

Figure 3. A document (a), its partitioning ((b), (c)) and one of its projections (d).

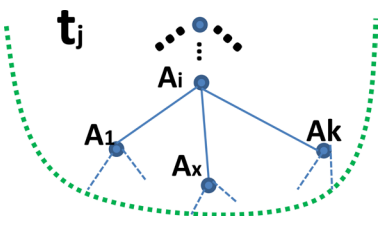

Figure 4. Case where an inner node n and its children $n_1, \cdots, n_k$ belong to the same subtree t_j such that $nt(t_j) \subseteq \mathcal{V}$.

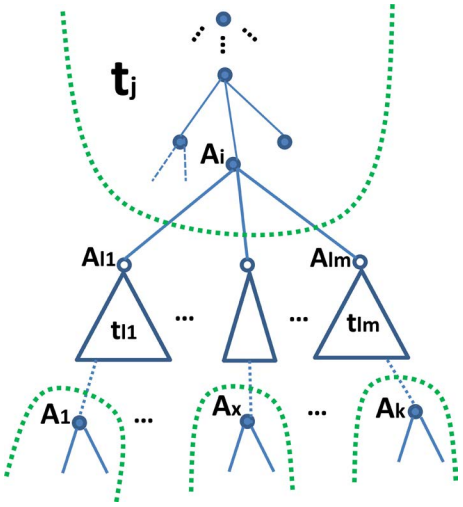

Figure 5. Case of labeled A_i leaf node n of subtree t_j ($nt(t_j) \subseteq \mathcal{V}$), with all its children belonging to subtrees of another type.

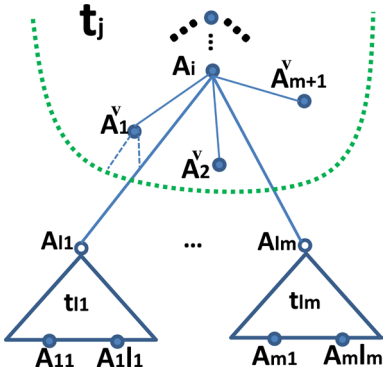

Figure 6. Case of an internal node n labeled A_i of a subtree t_j such that $nt(t_j) \subseteq \mathcal{V}$ with some of its children belonging to subtrees of another type.

Case 2: Node n labeled A_i is a leaf node of a subtree t_j such that $nt(t_j) \subseteq \mathcal{V}$. Let $A_{i_1}, \cdots, A_{i_m}$ be labels of m children of n in t. n has therefore been developed using a $\mathbb{G}_{\mathcal{V}}$'s production rule of one of the two forms $A_i \to B^*$, with $B \notin \mathcal{V}$; or $A_i \to A_1 \cdots A_k$ with $A_j \notin \mathcal{V}$, $j = 1, \cdots, k$. We develop below the second form, the treatment of the first being similar.

There is therefore m sub-terms of t, says $t_{l_1}, \cdots, t_{l_m}$, whose roots are respectively labeled by $A_{i_1}, \cdots, A_{i_m}$ and such that $nt(t_{l_j}) \in \overline{\mathcal{V}}$. According to the $P_{\mathbb{G}_{\overline{\nu}}}$'s building process[8], we can partition the word $A_1 \cdots A_k$ in m sub-words: $A_1 \cdots A_k = \underbrace{A_{11} \cdots A_{1l_1}}\ \underbrace{A_{21} \cdots A_{2l_2}} \cdots \underbrace{A_{j1} \cdots A_{jl_j}} \cdots \underbrace{A_{m1} \cdots A_{ml_m}}$ with

$$\begin{cases} l_1 + \cdots + l_m = k \\ A_{l_j} \Rightarrow^{+}_{\mathbb{G}_{\overline{\nu}}} A_{j1} \cdots A_{jl_j},\ j = 1, \cdots, m \text{ i.e. } A_{j1} \cdots A_{jl_j} \in \mathscr{L}\left(\mathbb{G}_{\overline{\nu}}, A_{l_j}\right) \end{cases}$$

As $A_{i_1} \cdots A_{i_m} \in \mathscr{L}(\mathbb{G}, A_i)$, $A_1 \cdots A_k \in \mathscr{L}(\mathbb{G}_{\nu}, A_i)$ according to the construction process of the productions rules of $\mathbb{G}_{\mathcal{V}}$ (modulo restructuring symbols).

[8]Reminder: the non-terminals of $\overline{\nu}$ are rewritten by the production rules of $P_{\mathbb{G}}$ by considering the symbols of ν as terminals.

Case 3: node n labeled A_i is an internal node of a subtree t_j such that $nt(t_j) \subseteq \mathcal{V}$ and, there is some n's children with labels not in $\mathcal{V}$. As previously, let's termed $A_{l_1} \cdots A_{l_m}$ labels of the m' children of n in t. n has therefore been developed using a production rule of the form $A_i \to A_1 \cdots A_{m'}$ in which at least one non-terminal belong to $\mathcal{V}$ and at least one other belong to $\overline{\mathcal{V}}$ (**Figure. 6**). Let m be the number of non-terminals on the right-hand side of this production belonging to $\overline{\mathcal{V}}$ and named as $A_{l_1} \cdots A_{l_m}$. As before, there are m sub-terms of t, which we call $t_{l_1}, \cdots, t_{l_m}$, having respectively $A_{l_1} \cdots A_{l_m}$ as their root labels nodes and such that $nt(t_{l_i}) \in \overline{\mathcal{V}}$. Similarly, according to the $\mathbb{G}_{\overline{\mathcal{V}}}$'s building process, we can partition the word $A_1 \cdots A_k$ in $2m+1$ sub-words: $A_1 \cdots A_k = A_1^{\nu} \underbrace{A_{11} \cdots A_{1l_1}} A_2^{\nu} \underbrace{A_{21} \cdots A_{2l_2}} \cdots A_m^{\nu} \underbrace{A_{m1} \cdots A_{ml_m}} A_{m+1}^{\nu}$ with

$$\begin{cases} A_i^{\nu} \in \mathcal{V}^*, 1 \le i \le m+1 \\ l_1 + l_2 + \cdots + l_m + \left|A_1^{\nu}\right| + \cdots + \left|A_{m+1}^{\nu}\right| = k \end{cases}$$ and such that

$A_{l_j} \Rightarrow^{+}_{\mathbb{G}_{\overline{\mathcal{V}}}} A_{j1} \cdots A_{jl_j}$, i.e. $A_{j1} \cdots A_{jl_j} \in \mathscr{L}\left(\mathbb{G}_{\overline{\mathcal{V}}}, A_{l_j}\right)$. As $A_{j1} \cdots A_{jl_j} \in \mathscr{L}\left(\mathbb{G}_{\overline{\mathcal{V}}}, A_{l_j}\right)$, $1 \le j \le m$, we have

$$A_i \Rightarrow A_1^{\nu} A_{l_1} A_2^{\nu} A_{l_2} \cdots A_m^{\nu} A_{l_m} A_{m+1}^{\nu} \Rightarrow A_1^{\nu} \underbrace{A_{11} \cdots A_{1l_1}} A_2^{\nu} \underbrace{A_{21} \cdots A_{2l_2}} \cdots A_m^{\nu} \underbrace{A_{m1} \cdots A_{ml_m}} A_{m+1}^{\nu} \in \mathscr{L}\left(\mathbb{G}_{\mathcal{V}}, A_i\right)$$

and then, $A_1 \cdots A_k \in \mathscr{L}\left(\mathbb{G}_{\mathcal{V}}, A_i\right)$. □

3.4. Illustration: Grammatical Structures for the Cooperative Writing of a Small Phone Book

Some of the concepts and algorithms presented in the previous sections are illustrated in this section by considering a simplified case of cooperative writing of a small phone book.

Suppose that two employees of an organization want to cooperate in writing a phone book for their organization. One entry of the book is given by the name (Name), two first names (Fname1 and Fname2), the mails addresses (Emails) and phones numbers (Phones).

A corresponding grammatical structure $\mathbb{G}_{an}$ describing this phone book is given in the **Figure 7**. Let us assume that there are two views:
$V_1 = \{phoneBook, contact, name, fname1, fname2, num\}$ and
$V_2 = \{phoneBook, contact, name, fnames, phones\}$ for each of the respective employees. By applying the **Algorithm 2**, we have in **Figure 8(a)** (resp. **Figure 8(b)**), the grammatical structure $\mathbb{G}_{V_1}$ (resp. $\mathbb{G}_{V_2}$) resulting from view V_1 (resp. view V_2) projection on $\mathbb{G}_{an}$. Note that in $\mathbb{G}_{V_1}$, *phone'* is a structuring symbol.

4. Conclusion

Asynchronous cooperative editing tools generally allows co-authors to edit *complete replicas* of a document and perform a posteriori merging [8] [9] [10] [11] [12] regardless if document is structured or not; It's the case in many tools

```
<phoneBook> -> <contact>*
<contact> -><name><fnames><emails><phones>
<emails> -> <email>*
<name> -> epsilon
<email> -> epsilon
<fnames> -> <fname1> <fname2>
<fname1> -> epsilon
<fname2> -> epsilon
<phones> -> <num>*
<num> -> epsilon
```

Figure 7. A grammatical Structure $\mathbb{G}_{an}$ of a phone book.

```
<phoneBook> -> <contact>*
<contact> -><name><fname1><fname2><phones'>
<name> -> epsilon
<phones'> -> <num>*
<num> -> epsilon
<fname1> -> epsilon
<fname2> -> epsilon
```

(a) Grammatical Structure $\mathbb{G}_{V_1} = \pi_{V_1}(\mathbb{G}_{an})$.

```
<phoneBook> -> <contact>*
<contact> -> <name><fnames><phones>
<name> -> epsilon
<fnames> -> epsilon
<phones> -> epsilon
```

(b) Grammatical Structure $\mathbb{G}_{V_2} = \pi_{V_2}(\mathbb{G}_{an})$.

Figure 8. Two local models resulting from the projection on global model of **Figure 7** according to view V_1 (a) and view V_2 (b).

of version managing like CVs for unstructured documents (textual merge) [13].

In the case of structured editing, all co-authors have the same document model and the merging of complete replicas relies on this model (syntactic merging software) [14] [15]. We were interested in this paper to an innovative case—we did not find any study that was done in this direction—in which the co-authors act on partial replicas of the overall document and each with a local model allowing him to validate locally updates made on its (partial) local replica.

We proposed as a document model in this context, *grammatical structures* allowing both to specify the model for the global document, and local models - for partial replicas—dedicated to each co-author. Furthermore, we have defined a projection operation to automatically derive the local models (grammatical structures) of documents from the global one.

Stability and consistency are some of the major properties enjoyed by grammatical structures. Consistency ensures that, every document validated locally with the local grammatical structure is always the projection of at least one valid document according to the overall grammatical structure: the grammatical structures thus offer to the different co-authors a suitable means of

carrying out local syntactic validations of the asynchronously edited documents, while ensuring consistency.

One can further this study by focusing on bottom-up construction of grammatical structures. The goal is to propose a "*grammatical structures merger*" similar to the "*documents merger*" presented in [4].

References

[1] Grudin, J. (1994) Computer-Supported Cooperative Work: History and Focus. *Computer*, **27**, 19-26. https://doi.org/10.1109/2.291294

[2] Baecker, R.M., Grudin, J., Buxton, W.A.S. and Greenberg, S. (1995) Readings in Human-Computer Interaction: Towards the Year 2000. 2nd Edition, Morgan Kaufmann Publishers, Inc., Burlington.

[3] Mens, T. (2002) A State-of-the-Art Survey on Software Merging. *Journal of IEEE Transactions on Software Engineering*, **28**, 449-462. https://doi.org/10.1109/TSE.2002.1000449

[4] Badouel, E. and Tchoupé, M. (2008) Merging Hierarchically Structured Documents in Workflow Systems, Proceedings of the Ninth Workshop on Coalgebraic Methods in Computer Science (CMCS 2008), Budapest. *Electronic Notes in Theoretical Computer Science*, **203**, 3-24. https://doi.org/10.1016/j.entcs.2008.05.017

[5] Brüggemann-Klein, A. and Wood, D. (1998) One-Unambiguous Regular Languages. *Information and Computation*, **142**, 182-206. https://doi.org/10.1006/inco.1997.2695

[6] Baecker R.M., Cristiana C. and Rosu, D. (2004) On Validation of XML Streams Using Finite State Machines. *Proceedings of the Seventh International Workshop on the Web and Databases*, Paris, 17-18 June 2004, 85-90.

[7] Badouel, E. and Lamine, M. (2014) Opacité des artefacts d'un système workflow. *Revue ARIMA*, **17**, 177-196.

[8] Berlage, T. and Genau, A. (1993) A Framework for Shared Applications with Replicated Architectures. *Proceedings of the Conference User Interface Systems and Technology*, **17**, 249-257.

[9] Balasubramaniam, S. and Pierce, B.C. (1998) What Is a File Synchronizer? *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking* (*MOBICOM*), Dallas, 25-30 October 1998, 98-108. https://doi.org/10.1145/288235.288261

[10] Wilm, J. and Frebel, D. (2015) Real-World Challenges to Collaborative Text Creation. *DChanges*'14 *Proceedings of the 2nd International Workshop on* (*Document*) *Changes*: *Modeling*, *Detection*, *Storage and Visualization*, Fort Collins, 16 September 2014, Article ID: No. 8.

[11] Decouchant, D., Quint, V., Riveill, M. and Vatton, I. (1993) Griffon: A Cooperative, Structured, Distributed Document Editor. Bull-IMAG, Grenoble.

[12] Fish R.S., Kraut, R.E., Leland, M.D.P. and Cohen, M. (1988) Quilt: A Collaborative Tool for Cooperative Writing. *Proceedings of Conference on Office Information Systems*, Palo Alto, 23-25 March 1988, 30-37. https://doi.org/10.1145/45410.45414

[13] Berliner, B. (1990) CVS II: Parallelizing Software Development. *The Advanced Computing Systems Professional and Technical Association* (*USENIX*), Prisma, Inc., Colorado Springs, 22-26.

[14] Buffenbarger, J. (1995) Syntactic Sofware Merging, Software Configuration Man-

agement: Selected Papers SCM-4 and SCM-5. In: Estublier, J., Ed., ACM, New York, 153-172.

[15] Fontaine, R.L. (2002) Merging Xml Files: A New Approach Providing Intelligent Merge of Xml Data Sets. The Pennsylvania State University, Philadelphia.