JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 37, 295-321 (2021)
DOI: 10.6688/JISE.202103_37(2).0002

Cross-Context Semantic Document Exchange Through A Novel Tabular Document Representation Approach

SHUO YANG[1,+] AND RAN WEI[2]
[1]*School of Computer Science and Cyber Engineering*
Guangzhou University
Guangzhou, 510006 P.R. China
E-mail: yangshuo@gzhu.edu.cn
[2]*Department of Computer Science*
University of California, Irvine
Irvine, CA 92697, USA
E-mail: weir4@uci.edu

Cross-context semantic document exchange is the process of representing, editing, and transmitting a semantic document in one context, and then receiving and interpreting it in another. It is an important research topic in semantic web, e-commerce and artificial intelligence. The current research methods of semantic document exchange generally include standardization, ontology modeling and collaboration templates, each of which has its own limitations. Based on Tabdoc, any semantic document used for interaction has the same syntax, conceptual meaning, and semantic relationship, regardless of context differences. The new method has been implemented in the SDF Tabdoc based on the XML format. By applying SDF Tabdoc, the Tabdoc editor has been designed and implemented to edit any context-independent document templates and document instances. Finally, the effectiveness of the proposed method is proved by case studies and experiments, indicating that any semantic document can be consistently understood between the parties.

Keywords: cross context, semantic document exchange, document inference, semantic interoperability, data integration

1. INTRODUCTION

In the real world, much knowledge is acquired from documents, including technical reports, journals and magazines, to name a few [1]. These documents implicitly contain schema and semantic meaning, which can be viewed as a kind of semantic document. In recent years, automatic semantic document interoperation, including interpretation [2] and employment, has become an important subject under the research topic of information interoperability and compatibility [3, 4] in areas such as industry, business, *etc*.

A semantic document is a document that is represented by humans and computers in a readable and understandable form [5]. But in practice, people often create documents in one environment (or domain), but analyze them in another environment. This process can be called cross-context semantic document interaction (or exchange). Cross-context semantic document exchange and analysis has attracted much attention in semantic web

Received October 1, 2019; revised February 4, 2020; accepted February 23, 2020.
Communicated by Jimson Mathew.
+ Corresponding author.

[6], e-business [7] and artificial intelligence [8, 9, 10]. Specifically, it is the process of representing and editing a semantic document (SD) in one context, and then receiving and interpreting the SD in another context. It can serve as the foundation of business and industrial interactions between service sectors for information system collaboration. For example, semantic document exchange across contexts allows heterogeneous industrial systems to inquire about a product, make a valid offer, negotiate on industrial terms, and sign a contract on behalf of unknown parties involved.

Current research methods on semantic document exchange can be divided into three categories, including standardization [11], ontology modeling [12] and collaboration templates [13]. These methods have their own advantages and disadvantages. Although standardization is effective for the uniform creation of documents, the document types and document elements created using this method are limited to standardized types [14, 15], so this method is more rigid. It is difficult to implement any complex type of documents without any constraints by using standardized methods[1][2][3][4][5][6]. Ontology modeling [16, 17] can be used to design most complex documents when ontologies are carefully designed. The biggest issue with this approach is its characteristic as "domain-wide", which prevents an ontologically modeled semantic document from being understood by other domains or contexts. Collaborative template approach [5, 13, 18, 19, 20] connects heterogeneous domains or contexts well by allowing exchange of heterogeneous semantic documents. However, its drawback is that every document template must be collaboratively created first. Document types that have not been collaboratively created cannot be semantically exchanged between domains or contexts. Furthermore, another drawback is shared by the first two approaches such that the instances filled in a document template cannot guarantee semantic consistency between any document sending party and receiving party; for a polysemous word (*e.g.*, crane) as an instance term, it is difficult to ensure that a sending party and a receiving party share the same understanding. Based on the discussion above, two research problems are summarized as below:

Problem 1 (*Document complexity problem*): Semantic documents are complex because each type of document is a complex semantic phenomenon. Users should be allowed to have autonomy, that is, create any semantic document based on the user's own ideas and requirements. However, because of the flexibility of language grammar and the complexity of conceptual relationships between terms in semantic documents (both are part of the semantic relation, it is not easy to design, create, use and parse complex semantic documents.

Problem 2 (*Cross-context interoperability problem, or cross-context problem*): Semantic documents of different domains have a semantic interoperability problem in three levels: document syntax (*i.e.*, different users can adopt different document grammars),

[1]EDI. http://www.edibasics.com/edi-resources/document-standards/.
[2]eCO. http://eco.commerce.net/.
[3]BizTalk. http://www.BizTalk.org/.
[4]RosettaNet. http://www.rosettanet.org/.
[5]cXML. http://www.cxml.org/.
[6]ebXML. http://www.ebxml.org/.

document relation (*i.e.*, different users can autonomously design a semantic relation in different ways), and document vocabulary (*i.e.*, document users can instantiate the same document template by using differently defined vocabularies). The cross-context problem hinders a semantic document correctly composed by a document writer in one place from being consistently processable and understandable by a document reader in another place [21]. However, many traditional methods (*e.g.*, [5, 13]) only partially consider limited structural relationships (*e.g.*, ancestor-descendent relationships or sibling relationships among XML nodes), which remains insufficient for effective cross-context semantic disambiguation [22].

In order to solve the above two problems, this paper proposes a Tabular Document Representation (TabDoc) method, which represents all heterogeneous semantic documents across contexts in a consistent way, thus making documents in various contexts interoperable at both the syntactic and semantic levels. The central idea of this approach is to adopt a "divide and conquer" strategy by defining complex semantic documents as a three-level structure of vocabulary, relationships and documents. Through this division, complex interoperability problems are divided into independent sub-problems of concept interoperability, relationship interoperability and document interoperability. The main contributions of this paper include:

- It enhances the collaboration between different information systems through a new semantic interoperability approach (Tabdoc approach) by allowing consistent syntactic processing and semantic understanding of exchanged semantic documents.
- It contributes to the intelligence of document understanding and processing by proposing a new semantic interpretation algorithm that provides a foundation to develop intelligent applications.

In the rest of this paper, Section 2 proposes a novel Tabdoc approach to resolving problems of document complexity and cross context. Section 3 implements Tabdoc approach as a Tabdoc Editor. In Section 4, experiments are conducted to evaluate the applicability of Tabdoc Editor. Section 5 compares the approach proposed in this paper with other related ones. Finally, a conclusion is made with summary, contributions and future work.

2. TABULAR DOCUMENT REPRESENTATION (TabDoc)

This section proposes a novel Tabdoc approach to resolving the two problems so as to reduce document complexity and achieve semantic interoperability between exchanged semantic documents across contexts.

2.1 Overview

The Tabdoc method (shown in Fig. 1) is a holistic solution for how to implement cross-context semantic document exchange. It is divided into three levels: vocabulary, relation and document.

The vocabulary level aims to solve the problem of conceptual interoperability by using the general CONEX dictionary (CoDic) [18, 23]. CoDic is a common vocabulary

designed by parties involved in information exchange under the CONEX project [19]. Any dictionary term (often referred to as a symbol) in CoDic is uniquely identified as an internal identifier ($iid \in IID$). This identifier is neutral and independent of any natural language. In CoDic, there are two types of concepts: (1) general concepts for different document systems (DS) in different natural languages, *i.e.*, general concepts of common vocabularies (CV) and (2) local concepts used by various industrial or business groups, *i.e.*, local concepts in local vocabulary (LV), local document templates, and locally instantiated documents. CoDic guarantees that all concepts are accurate and semantically consistent without ambiguous.

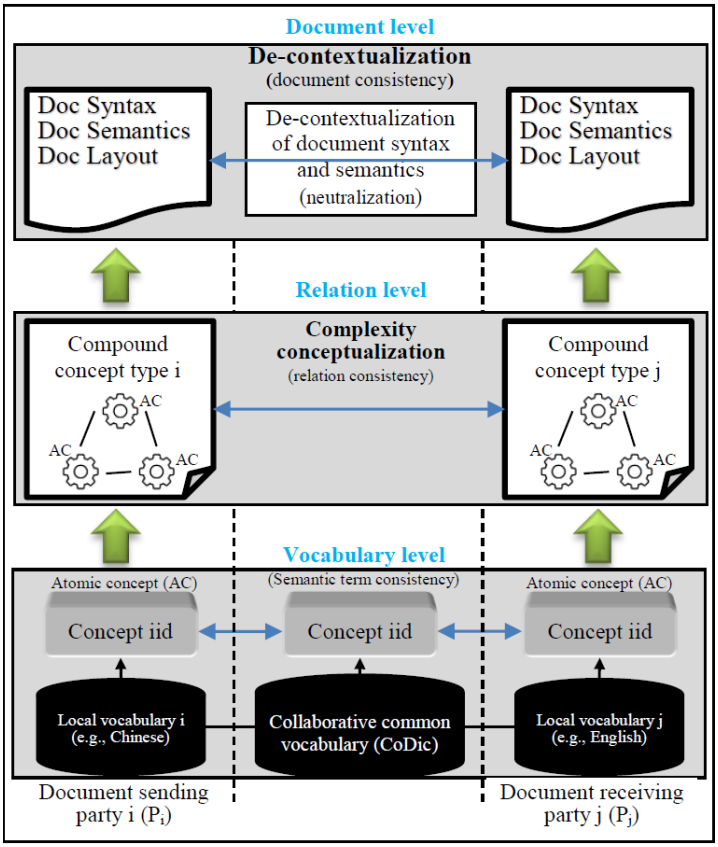


Fig. 1. Tabdoc approach.

The relation (or concept type) level uses a new complexity conceptualization strategy (Section 2.3.1) to resolve the document complexity problem. It views any semantic document as groups of compound concepts (Section 2.3.1), each of which consists of a set of atomic concepts concatenated with semantic relations (Section 2.3.1). Thus, it is necessary to summarize as many compound concept types as possible and illustrate how each of them is represented in semantic documents. The more compound concept types are represented, the more the variety of the document content can be expressed.

The document level achieves the document interoperability among different contexts based on a newly proposed de-contextualization strategy. It includes syntactic de-contextualization and semantic de-contextualization. The reason for designing syntactic de-contextualization is because the grammar rules for document design and representation are context-sensitive and not interoperable between different contexts. Users often use different document generation grammars based on their local culture and customs. However, this prevents the document from being syntactically interoperable between different contexts/domains. To solve this problem, a syntactic de-contextualization for syntactical neutralization is needed to make a document be syntactically interoperable. In addition, the lack of semantic interoperability is obvious for online message passing, because many services have a proprietary interface and were originally designed for standalone applications [2, 3]. To handle this problem, semantic de-contextualization is proposed to enable documents understood unambiguously in heterogeneous contexts.

In summary, the vocabulary level and concept type level provide common concepts and relations, which serves as the foundation to fulfill cross-context semantic understanding. The document type level takes responsibility for cross-context consistent document interoperation by using a novel de-contextualization strategy.

2.2 Syntactic De-contextualization

Syntax is the set of rules, principles, and processes that govern the structure of statements in a given language. To realize syntactic de-contextualization, we design and implement a universal and scalable document representation grammar. It aims to construct general grammatical rules on organization of terms for reading, understanding and editing documents. It needs to conform to two principles. First, the universal document syntax should follow a context-free grammar. Normally, a context-free grammar is a four-tuple consisting of a set of non-terminal variables, terminal variables, production rules and a starting variable. The universal document syntax, acting as a context-free grammar, is the foundation of building a universal document parser, and thus it needs to follow particular rules for easy parsing. Second, the language generated from the universal document syntax should be a collaborative language. This means it allows language users to collaboratively update the language based on their practical intentions.

Tabdoc Grammar, as a universal document syntax, is mainly composed of two types of productions/rules:

$$sign ::= sign(sign(sign[Attlist]^*)^*)|\varepsilon \tag{2.1}$$

$$Attlist ::= attribute-value\ pairs \tag{2.2}$$

where the "sign" in Eq. (2.1) is called a non-terminating symbol or a starting symbol. The symbol ε is indicated as empty. The "Attlist" in Eq. (2.2) is a list of attribute-value pairs as a feature set of any sign element. Tabdoc language is a set of strings that are derived from the Tabdoc syntax by several derived steps. In each step, one or several production/rules are applied to derive the non-terminal. In this article, the Tabdoc language, also known as the Tabdoc schema is designed to create document templates and document instances. The entire process of document creation is described in Eq. (2.3).

$$\begin{aligned} Tabdoc\ Grammar \Rightarrow Tabdoc\ language(\text{i.e.}, &Tabdoc\ schema \\ &in\ XML) \Rightarrow document\ template \Rightarrow document\ instance \end{aligned} \tag{2.3}$$

It needs to be noted that in this paper, Tabdoc schema is implemented in XML schema [24]. However, it can also be implemented by other methods.

2.2.1 Represent a document as a table

Syntactically, a document is about how to organize terms to create a well-formed grammatical relationship pattern. In the Tabdoc Grammar a complex document is in the form of nested tables, described as a set of hierarchically arranged matrices. These nested tables construct a matrix tree, in which the each table in the hierarchy is built in order. The matrix tree (MTree) d_s is defined in Eq. (2.4).

$$d_s = \begin{bmatrix} c_{r_1c_1} & \dots & c_{r_1c_m} \\ \dots & \dots & \dots \\ \dots & c_{r_ic_j} & \dots \\ \dots & \dots & \dots \\ c_{r_nc_1} & \dots & c_{r_nc_m} \end{bmatrix}, c_{r_ic_j} = \begin{bmatrix} c_{r_1c_1} & \dots & c_{r_1c_m} \\ \dots & \dots & \dots \\ \dots & c_{r_ic_j} & \dots \\ \dots & \dots & \dots \\ c_{r_nc_1} & \dots & c_{r_nc_m} \end{bmatrix} \tag{2.4}$$

where any $c_{r_ic_j}$ is a table cell in the ith row and the jth column ($0 < i \leq n$, $0 < j \leq m$). A matrix tree can also be notated in Eq. (2.5).

$$d_s = M^0_{m_0n_0}(1,1).M^1_{m_1n_1}(I_1,J_1)\dots M^k_{m_kn_k}((I^1_k,J^1_k)|(I^2_k,J^2_k)|\dots).(M^{k+1}|\dots) \tag{2.5}$$

where $M^0_{m_0n_0}(1,1)$ is a root cell (*i.e.*, a one cell table) that can contain a table or tables (*e.g.*, separated by a vertical line between embedded sub-tables), and any matrix $M^k_{m_kn_k}(I_k,J_k)$ represents an embedded table in the kth level with dimension $(m_k * n_k)$ in the tree-like hierarchy structure, which contains another sub-table in its cell (I_k,J_k). The location of a cell $C_k(i,j)$ is called the cell identifier (cid), in which the cells (or symbols) of each table in the hierarchy are constructed in a sequential structure. Eq. (2.6) uniquely identifies the cid of a cell:

$$cid = 1.(I,J)_1.(I,J)_2\dots(i,j)_k = 1.(I_1,J_1).(I_2,J_2)\dots(i_k,j_k). \tag{2.6}$$

In order to access the cells in the ith row and the jth column in the k-level matrix, it is only necessary to traverse those cells in the previous (k–1) levels matrixes that have the nested matrixes. (i_k,j_k) denotes the position of the cell nested in the innermost sub-table in the hierarchical matrix. $(I_x;J_x)$ refers to the coordinate position of a cell in the outer nested table at the xth layer, and the cell is used to nest the (x+1)th layer matrix. Fig. 2 shows an example of a matrix tree.

2.2.2 Constraining cells by attributes

In real-world document usage, the cells (or symbols) in a document are often complex and require multiple properties to define it. In this section, the set of attributes describing the cell is divided into five categories, as shown in Table 1, which are used to express a cell in five aspects: identification, data model, sub-cell, instance, and presentation style. Eq. (2.7) describes a set of attributes (A) for any cell (or symbol σ) in the MTree.

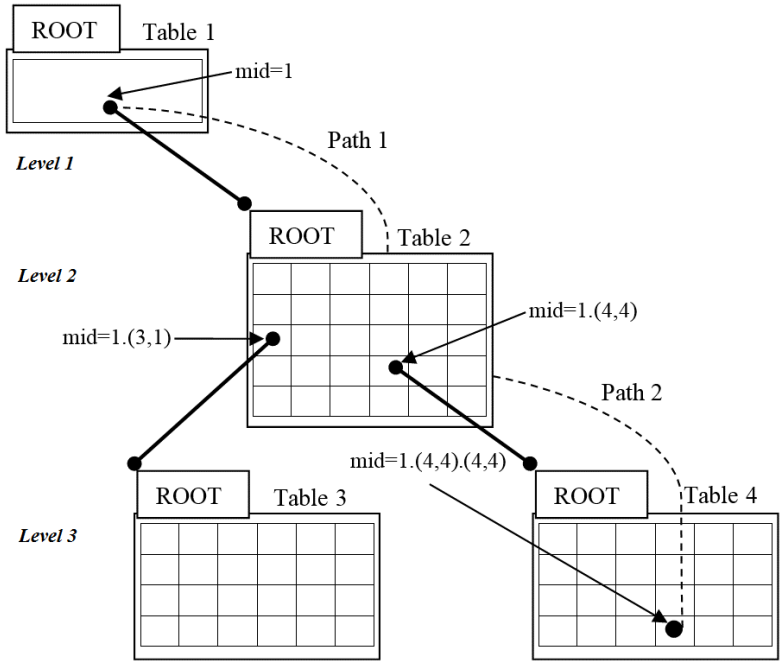


Fig. 2. An example of a matrix tree.

$$A ::= ref, pid, term, size, st, max, min, pos, cho, sel, imp, pri, inst, iof, dt, op, f, r, disp, style \quad (2.7)$$

In particular, PID and REF are used to form a sign σ, where PID is the form of σ, and REF is the intrinsic meaning and denotation of σ, which follows the dual symbol model theory of Saussure [25]. The other attributes are the modifiers for σ, where ST, POS, MIN, MAX, CHO, SEL, IMP, and PRI are used to describe the functionality and connotation of σ, while DT, OP, F, and R are used to instantiate an abstract σ to create a concrete statement or fact, like "sign $\sigma[pid, ref, op, dt, f, r] < iidorliteral >$". For example, the statement "fridge is white" can be expressed as: $\sigma[pid =$ "$(1,3).(1,2)$", $ref =$ "$5107df022918$", $op =$ "is"$] < 5107df02f309 >$, where in the dictionary CoDic, "iid = 5107df022918" means "refrigerator", and "iid = 5107df02f309" means "white" [23].

2.2.3 XML implementation of Tabdoc Grammar

To concretely create and edit a semantic document, Tabdoc Grammar, described above, is implemented in Sign Description Framework (SDF) [23] in XML, called SDF Tabdoc, so that it is compatible with most existing document systems. SDF Tabdoc has only one XML element, called $< sign >$, which defines a concept with a set of attributes shown in Table 1. $< sign >$ is nested to construct a sign tree to define a semantic document. In Fig. 3, the XML schema for SDF Tabdoc is described, which is simple, universal and context-free.

Table 1. Attributes of a cell (or a sign).

Attribute (feature) Name	Definition	Description
(1) The location and meaning of the current cell		
Template identifier	tid = uid:tid_nq	Unique template id of a user
Instance identifier	nid = tid:nid_nq	Unique instance id of a user
Position identifier	pid = cid	Location of a cell (or sign)
Reference(mandatory)	ref = $iid_1, \ldots, iid_n \subset CoDic$; $ref = iid \leftarrow dt \subset CoDic$	$iids \subset CoDic$ to define meaning of current *pid*. It refers to a *iid*-ed data type
Term (mandatory if ref = dt, else optional)	term : $literal \leftarrow iid$ of dt	Term iff σ is a data type and specific literal term in local language must be provided
(2) The sign type of the current cell and its associated sub-cells		
Sign type (or data model)	st = cell | col | row | table for current cell	sign type such as table, row, column and cell
Compound sign size	size = (i, j) | j | i	the size of a sign type
Cell position (mandatory if table)	cid (or pos) = (i, j) | j | i	i represents the row number, j represents the column number
(3) Constraints on sub-cells in the current cell (optional)		
Occurrence	min = (0, n)	Sub-cell min occurrence
	max = (0, *)	Sub-cell max occurrence
Sequence	sequential sibling order	Sub-cell sequential order
Choice	cho = 1 | n, for current; sel = yes | no, for child	Single or multiple selection on sub-cells
Importance	Imp = yes | no, for current; pri = n, for child	Describe the importance and priority order of sub-cells
Semantic relation	sem = (semantic relation type)	Reserved attribute used to define semantic relationships
Anchor	anc = [pos*]	Pointing to related cells
(4) Constraints for the current cell instance (optional)		
Value type (omit if not allowed)	st = val	A value type as an instance of another sign type
Instance	inst = instance	Instance of a cell or sign
Instance of	iof = instance of	Instance of another sign/cell
Data type	dt = xml schema data type and Tabdoc defined data types	The data type is used to limit how the value of the current cell is instantiated
Operand	op = is | empty | lte | lt | gte | gt | likely | not | reciprocal | factorial | ceiling | floor | any atomic verb or compound verb	An operand is to express action, state, willingness, possibility, and judgement of current node, and is to affect the instance of current cell
Formula	f = formula	Formula for instance
Rule	r = rule x, or r = if $x_1, \ldots, x_n$ then y	A well-formed rule deriving instance of current cell
Linguistic grammar	lg = [S+V+O | P | ...]	A reserved attribute for constrained grammar to phrases or sentences
(5) Limit the display of the contents of the sign		
Display	disp = inst | term | ref	How to display current cell
Style	style = css	How a sign is presented

2.3 Semantic De-contextualization

Semantics is the language meaning assigned to a term, phrase, sentence, paragraph and even an article. Document semantics includes meaningful terms and semantic relations between terms [5]. Semantic relations include the explicit relation (*e.g.*, instance relation) and implicit relation (*e.g.*, linguistic grammar) between any lexical terms of a document. Thus, ensuring consistent understanding of a document needs to guarantee the accurate interpretation of the concepts of all terms and the relations between them. The idea to achieve semantic de-contextualization is simple. Each meaningful term is semantically assigned with a unique *iid* referring to a unique meaning of a term in a CONEX Dictionary (called CoDic), which follows the CONEX collaboration principle [18, 23].

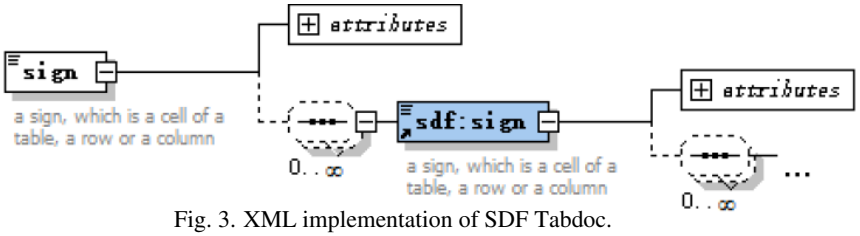


Fig. 3. XML implementation of SDF Tabdoc.

(Please see full XML Tabdoc at: www.cis.umac.mo/~jzguo/pages/tabdoc/sdfd-schema-v10-2017-03-08.xsd)

Each type of semantic relation also semantically makes reference to a unique *iid* that is a unique meaning of a relation in CoDic. Thus, the semantics of any document designed are context-free because it has been transformed to a set of hierarchically sequence of *iids*.

2.3.1 Complexity conceptualization strategy

A semantic document can be partitioned into several separated semantic units for easy understanding. Each semantic unit is conceptualized as a particular compound concept type. If we could tabularize any complex semantic document in an array of rows and columns, cells filled with atomic concepts will be built hierarchically into compound concepts.

Definition 1 (Atomic concepts, AC) *An atomic concept is a collaborative sign (cosign) with atomic meaning, which is defined as:*

$$AC = (S, W, D, L) \tag{2.8}$$

where W is any character string with a common identifier S and definition D. L is a natural language describing W and D. Atomic concepts are collaboratively built by experts from different domains. For example, (0x5107df00da93, "employee", "contributes labor and/or expertise to endeavor of an employer and is usually hired to perform specific duties which are packaged into a job", "English") is an atomic concept.

Definition 2 (Compound concepts, CC) *A compound concept defines an association/relationship among a set of atomic concepts (AC), which is defined as:*

$$CC = (GID, AC_{i_1}, \ldots, AC_{i_n}, CCT, AC_{j_1}, \ldots, AC_{j_m}) \tag{2.9}$$

where any compound concept CC with type CCT is identified by a group identifier GID; AC_{i_1},...,AC_{i_n} is a sponsor of CCT, while AC_{i_1},...,AC_{i_n} is a consumer. For example,cc_1=(001,invoice, part-of, product, price, tax, buyer, seller) means that an invoice has product, price, tax, buyer and seller as its components. In applications, CCT (e.g., part-of in the example) is replaced by its identifier.

Compound concept types (CCT) represent the relationship between atomic concepts, which is defined as:

$$CCT = (S_{cct}, D_{cct}) \tag{2.10}$$

where S_{cct} is a common relation identifier for a compound concept type; D_{cct} is a definition of S_{cct}. The more compound concept types are defined, the more complex semantic documents can be represented. There are many types of relationships between concepts in real natural languages. Therefore, compound concept types are diverse and difficult to be listed thoroughly. This paper mainly uses eight types of compound concepts imported from information science [26] for Tabdoc document representation. They are *reference relation, part-of relation, parallel relation, calculation relation, sequence relation, progressive relation, choice relation* and *instance relation*, whose definitions can be found in [27].

When a document creator draws a sketch of the document template according to the Tabdoc Grammar, our method relies on a tabular control on the computer screen to aid in the expression of mind-thinking, concept selection, and relationship building. Specific steps are as follows.

Step 1: Create a cell on the computer screen as the root cell of the row, column, or table, and fill in the value of the property list in that cell.
Step 2: Use the Tabdoc Editor to extend the root cell to a cell, row, column, or table.
Step 3: Traverse each empty cell and specify the value of its attribute list to define them and establish a semantic relationship (*i.e.*, CCT) between multiple cells.
Step 4: Repeat step 3 until users no longer need to specify attributes for creating sub-tables, sub-rows, sub-columns, or sub-cells.

During specifying attribute values of each cell, a document user applies the context-free common terms $\tau \subset CoDic$ [23, 18] as the semantic references of all signs σ, such that a meaning $m : (iid \leftarrow \tau) = (iid \leftarrow \sigma)$. Since each term (or sign) meaning in a document template is unique and context-free in terms of an "*iid*", the document template or the semantic pattern (*i.e.*, the multiple relations between sign *iids*) is also unique and context-free. This allows any document template to be personal to an individual user but still semantically interoperable.

2.3.2 Semantic interpretation

This paper uses inference on rules for semantic interpretation, which means it uses rules to capture the semantics of a document. Thus, rule execution is the process of understanding of a semantic document. Using rules as a semantic carrier has two benefits. The first is the freedom in rule design and creation due to the generality of the context-free rule syntax [28, 29]. Second, from a technical perspective, user-defined semantic relation types and attributes can be automatically transformed into patterns of rules to participate inference for semantic interpretation without requiring new function definition and recompilation. Users only need to declare transformation paradigms to define how they are transformed into rule patterns. For example, in Drools, a ".dslr" file can be transformed into rule patterns via a corresponding ".dsl" file. The rule semantics is also context-free due to the direct inheritance from the *IID*-based context-free Tabdoc document.

Stage a) Extracting logical structure of a Tabdoc document In the first stage of semantic interpretation, a Tabdoc document is logically understood via a Vector Tree Model (see

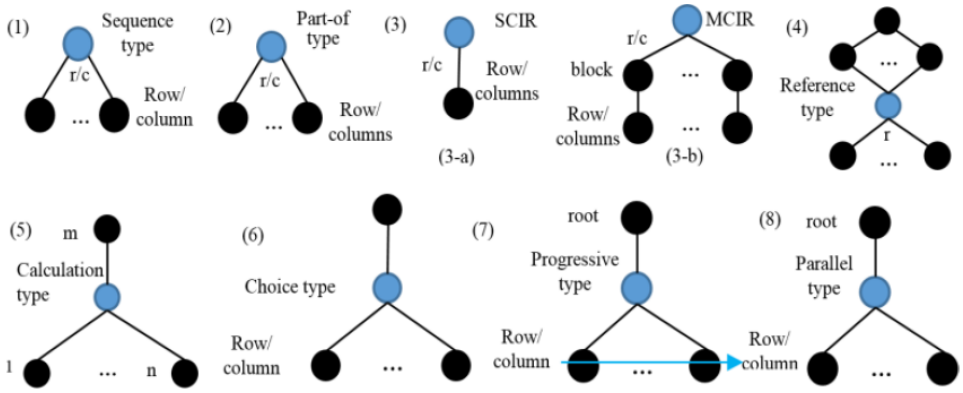


Fig. 4. Sub-VTree structures of CCTs. SCIR is short for single column instance relation. MCIR is short for multiple columns instance relation [27].

Definition 3). The logical structure is the hierarchical conceptual relation between semantic units in a Tabdoc document.

Definition 3 (Vector tree, VTree) *Any document that can be tabularized is further logically represented by a vector tree* τ*:*

$$\tau = (I_1^1, I_i^1, \ldots, I_i^k, \ldots, I_m^n), \tag{2.11}$$

where (1) each node in τ *is represented by* I_p^q*; (2) the level of a node in* τ *is* $q \in (1, \ldots k, \ldots n)$*; (3) sibling nodes are of the same level and represented by* $p \in (0, \ldots i, \ldots m)$*; (4) the parent of a node at level k is represented by a vector* $(I_1^1, \ldots, I_i^{k-1})$*; (5) the children of a node at k level is a set of vectors* $(I_1^1, \ldots, I_i^{k+1})$ *and (6) root of* τ *is a one-dimensional vector* (I_1^1) *with* $k = 1$ *and* $i = 1$*.*

Each node in a Vector tree (VTree) represents a concept in a Tabdoc document. The position of a node in VTree refers to its corresponding cell identifier, which ensures that VTree is mappable with the Matrix Tree (MTree). Based on different compound concepts defined in the complexity conceptualization strategy (Section 2.3.1), a Vector Tree (VTree) Building Algorithm is proposed to construct the logical structure of a Tabdoc document. The steps of VTree building algorithm are as below:

Step 1: Based on atomic concepts and the semantic relations between them, basic sub-VTrees (*e.g.*, Fig. 4) are first constructed.
Step 2: Based on the semantic relations between compound concepts and sub-VTrees obtained from Step 1, complex sub-VTrees are constructed.
Step 3: Repeat Step 2 until all compound concepts are as independent semantic units.
Step 4: Combine all complex sub-VTrees to form a complete VTree by defining a shared root node.

Stage b) Semantic interpretation algorithm (SIA) Based on the improved Rete algorithm, this section proposes a novel semantic interpretation algorithm. The improved Rete algorithm is implemented by a network built on interconnected nodes. Each node represents one or more tests in the predecessor (LHS) of a rule, and each node has one or two inputs and any number of outputs. The network will handle the facts that are added to or deleted from the working memory. The input and output nodes are located at the top and bottom of the network, respectively. The various types of nodes described above together form a Rete network, and the network is the way in which the improved Rete working memory works.

Semantic Interpretation Algorithm (SIA)
Input: Tabdoc document (Doc)
Output: Tabdoc parser results
1) Declare the fact template (FT) of Doc;
2) Create facts (F_{Doc}) based on FT and Doc;
3) Insert F_{Doc} into a working memory (WM_{Doc});
4) Load rules (R_{Doc}) created from Tabdoc document and rules (R_{Exe}) for the execution of Tabdoc document into Product Memory (PM);
5) Based on WM_{Doc} and PM, create knowledge session (KS) that continuously communicates with rule engine (RE) by loading new facts and rules;
6) Run *Rete_improved algorithm* to activate rules by facts through pattern matching;
7) Execute each activated rules by agenda through conflict resolution strategies.

The novel semantic interpretation algorithm consists of several steps as follows:

Step 1: Based on the logical structure constructed by VTree Building Algorithm, it is easy to declare the fact templates (FT) of a Tabdoc document. This step is similar to defining which concept should be a class name and which concepts should be its attributes.
Step 2-3: Based on the FT extracted in Step 1 and the instantiated content in the Tabdoc document (Doc), facts are created and inserted into the working memory (WM_{Doc}) that is analogous to a database to store facts transformed from a Tabdoc document.
Step 4: This step aims for rule creation. According to the source and functionality, rules have three categories. The first category is $Rule_{Doc}(R_{Doc})$ created by Tabdoc documents. This kind of rule has three sub-types: (1) rules for querying information in a Tabdoc document; (2) rules for creating facts (*e.g.*, a user-filled single choice); (3) rules for computing arithmetic results of mathematical formulas. The second category is $Rule_{Exe}$ (R_{Exe}), which is used for program execution. This means the document parser program is also represented in the form of rules. In this way, document processing does not rely on a fixed document parser program (*e.g.*, DOM, SAX) or limited document layout structure. $Rule_{Exe}$ also includes three sub-types: (1) rules for loading facts parsed from Tabdoc document into working memory; (2) rules for assessing which facts do not meet the limitations and requirements in the Tabdoc document; (3) rules for loading the priority of different rules. The third category is business rules and processes, which is beyond the discussion of this paper.

Improved Rete Algorithm (Rete_improved)
Input: productions (rules)
Output: Rete network (combination of Alpha net and Beta net)
1) Create a root node;
2) Add a rule (Alpha node starts from 1, Beta node starts from 2):
a. Read a pattern from the rule (*i.e.*, an att-val pair in the LHS), and then check the type of the pattern (*e.g.*, predicate);
b. Check whether the corresponding alpha node exists for the pattern. If it exists, record the node location, and if not, add the pattern as a new alpha node to the network and create an alpha memory table based on the pattern;
c. Repeat step (b) until all patterns have been processed;
d. Create Beta nodes as follows: The left input node to Beta(2) is Alpha(1), and the right input node is Alpha(2). The left input node to Beta(i) is Beta(i-1), while the right input node to Beta(i) is Alpha(i), i>2, and inline the memory tables of the two parent nodes (*i.e.*, Beta(i-1) and Alpha(i)) of Beta(i) to become its own memory table;
e. Repeat step (d) until all Beta nodes have been processed;
f. Action (Then) part is encapsulated as a leaf node (Action node) as the output node of Beta(n);
3) Repeat 2) until all rules are processed.

Step 5-7: Load working memory and production memory into pattern matcher and run Rete_improved algorithm to execute pattern matching between facts and the condition parts (LHS, left-hand side) of rules. Each matched rule will be marked as an activation status, which will be waiting for execution of the header parts (RHS, right-hand side) of the rules. The execution procedure is controlled by an agenda that takes responsibility of resolving conflicts by using various conflict resolution strategies (*e.g.*, priority), which is out of the research scope of this paper.

3. IMPLEMENTATION: TABDOC EDITOR

This section designs a tabular document editor, Tabdoc Editor, to implement Tabdoc approach. By the Tabdoc Editor, the semantics of any document are context-free because it has been transformed to a set of hierarchically sequences of iids.

3.1 Tabdoc Editor Prototype

The Tabdoc Editor can implement Tabdoc documents under Tabdoc Grammar and parse its semantics via a generalized inference engine. Tabdoc Editor consists of components of *Table Console*, *Property List*, *Tabdoc Parser*, *Semantic Input Method (SIM)* [5] and *CONEX dictionary (CoDic)* [5] for document creation and components of *Inference Engine (including VTree building algorithm and SIA algorithm)* for semantic interpretation.

Table Console creates tables (including cells, rows and columns) based on the property values and instance values entered by document template designers and users. Property List provides both template designers and users a form to define and operate each

cell of a tabular document. Based on a Tabdoc template, a template user can instantiate it into a Tabdoc instance. Tabdoc template is checked for its well-formedness under Tabdoc schema by Tabdoc Parser. Tabdoc instance is also checked for its validity when instantiating a Tabdoc template. In Tabdoc Editor, template designers and users should adopt SIM to input signs from CoDic. All inputted words are restricted to CoDic unless the inputs are literals.

After a Tabdoc document is received, the novel vector tree (VTree) building algorithm is used for the construction of logical structure through the properties defined during the document creation. By preorder traversing the tree structure, we analyze the type of each node through the sign type property (*st*) to distinguish whether it is an entry, or a label, or a semantic relation type. When we read a label (*i.e.*, property *st*=*cell*), we can extract its identifier value from its property *ref* (*i.e.*, reference) and then search in the CoDic dictionary to get its concept. Its corresponding value cell can be identified based on the logical structure through the positional property *iof* (*i.e.*, instance of) and the instance property *ins* (*i.e.*, instance value). The semantic relation between them can be acquired from the property *op* (*i.e.*, operand). Thus, when we read the sub-tree structure of a semantic relational type, we can construct the semantic relation components, such as "entry-label pair" or "label-label pairs".

3.2 An Example of a Ticket Booking Document That Can Be Cross-Contextually Understood

This subsection uses as an example a ticket booking document that is written in Chinese by a party and then sent to another party that can only understand English. The Tabdoc approach presents a solution to resolve the heterogeneous semantic interoperability problem by neutralizing concept meanings through identifying any concept in use as context-free IIDs. It has realized the solution via a meaning transformation of "Chinese instance $\rightarrow$ context-free IID-based instance $\rightarrow$ English instance". By this meaning transformation, any semantic document can be exchanged without the loss of meaning accuracy. For example, "Fig. 5 $\rightarrow$ IID-based document as shown in Fig. 6 $\rightarrow$ Fig. 7".

Fig. 8 shows a number of (1) rules mapped from received flight reservation documents (*e.g.*, Fig. 7), and (2) rules automatically created for document execution, which is used as input to the inference engine in the Tabdoc Editor of the document recipient. In Fig. 9, each term in rules is substituted by their term identifier (IID), which guarantees each term has an unambiguous meaning during the inference procedure.

4. EXPERIMENTS

4.1 Datset Description

To evaluate the performance of the Tabdoc approach, the experiments will use two datasets.

Dataset 1: The data source comes from the data set Troy200 [30]. It builds 10,000 forms based on 10 different government statistics sites. Fig. 10 shows some samples from the data set.

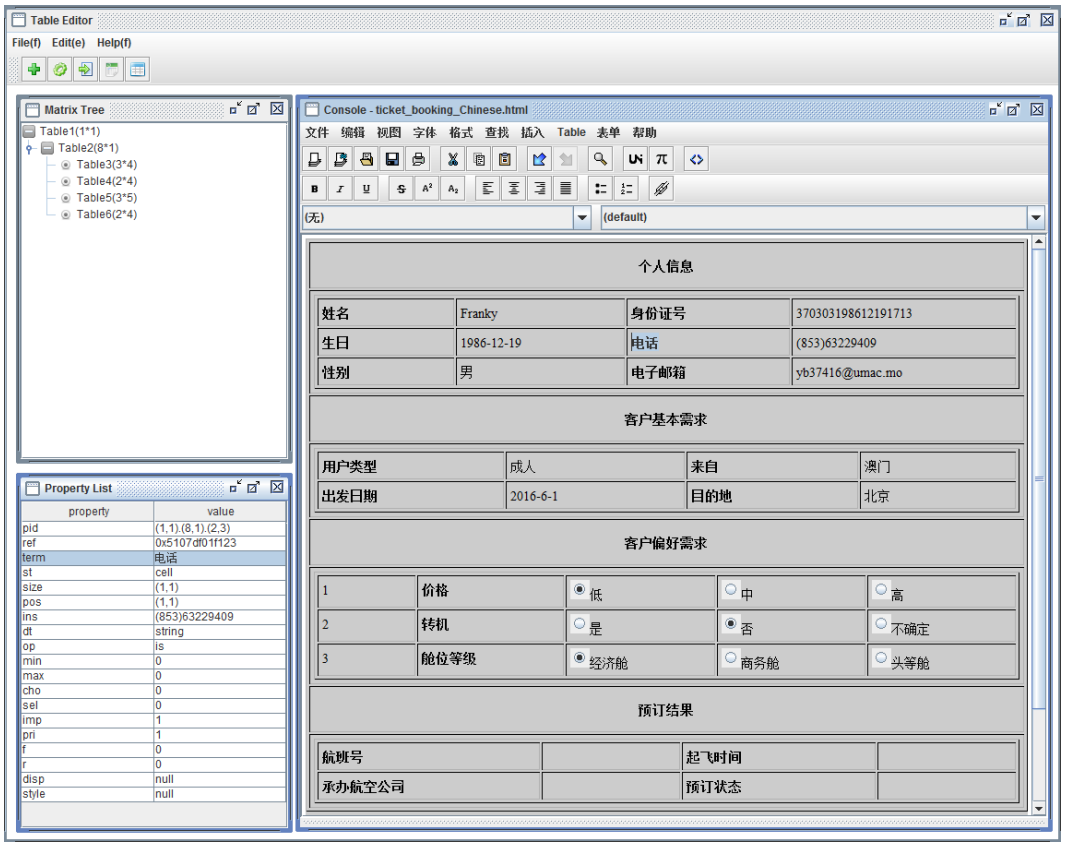


Fig. 5. A Tabdoc document displayed in Chinese.

```
<?xml version="1.0" encoding="UTF-8"?>
<sdf:doc xmlns:sdf="std:sdf:doc:table:schema-2017-3-8" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="std:sdf:doc:table:schema-2017-3-8
file:///E:/l%e8%ae%ba%e6%96%87%e6%8a%95%e7%a8%bf/10.IEEE%20Transaction%20on%20industrail%20informatics/13.%e5%a4%a7%e4%bf%ae/7.%20%e8%87%aa%e
5%b7%b1TII%e5%a4%a7%e4%bf%ae/6.%20%e5%ae%9e%e9%aa%8c/%e8%bf%87%e8%80%81%e5%b8%88%e5%ae%9a%e4%b9%89%e7%9a%84schema/sdfd-schema-v
10-2017-03-08.xsd">
  <sdf:head/>
  <sdf:body>
    <sdf:sign st="cell" pos="(1,1)">
      <sdf:sign st="column" size="(8,1)">
        <sdf:sign st="cell" pos="(1,1)" ref="0x5507df01edfb 0x5107df01551a"/> <!--personal information-->
        <sdf:sign st="table" pos="(2,1)" size="(3,4)">
          <sdf:sign st="cell" pos="(1,1)" ref="0x5107df01b82c" op="is"/> <!--name-->
          <sdf:sign st="value" pos="(1,2)" iof="(1,1)" >0x5707df01b833(Franky)</sdf:sign><!--0x5707df01b833 is name type.-->
          <sdf:sign st="cell" pos="(1,3)" ref="0x5107df014957" op="is"/> <!--id number-->
          <sdf:sign st="value" pos="(1,4)" iof="(1,3)" dt="sdf:serialNumber">370303198612191713</sdf:sign>
          <sdf:sign st="cell" pos="(2,1)" ref="0x5107df0042d7" op="is"/> <!--birthday-->
          <sdf:sign st="value" pos="(2,2)" iof="(2,1)" dt="xs:date" >1986-12-19</sdf:sign><!--0x5707df01b833 is name type.-->
          <sdf:sign st="cell" pos="(2,3)" ref="0x5107df01f123" op="is"/> <!--Phone-->
          <sdf:sign st="value" pos="(2,4)" iof="(2,3)" dt="sdf:serialNumber">(853)63229409</sdf:sign>
          <sdf:sign st="cell" pos="(3,1)" ref="0x5107df0114d9" op="is"/> <!--Gender-->
          <sdf:sign st="value" pos="(3,2)" iof="(3,1)" dt="xs:string" >0x5107df01910e</sdf:sign><!--male-->
          <sdf:sign st="cell" pos="(3,3)" ref="0x5107df00d18e" op="is"/> <!--Email-->
          <sdf:sign st="value" pos="(3,4)" iof="(3,3)">0x5107df00d18e(yb37416@umac.mo)</sdf:sign><!--0x5107df00d18e is email type-->
        </sdf:sign>
```

Fig. 6. An IID-based Tabdoc document instance.

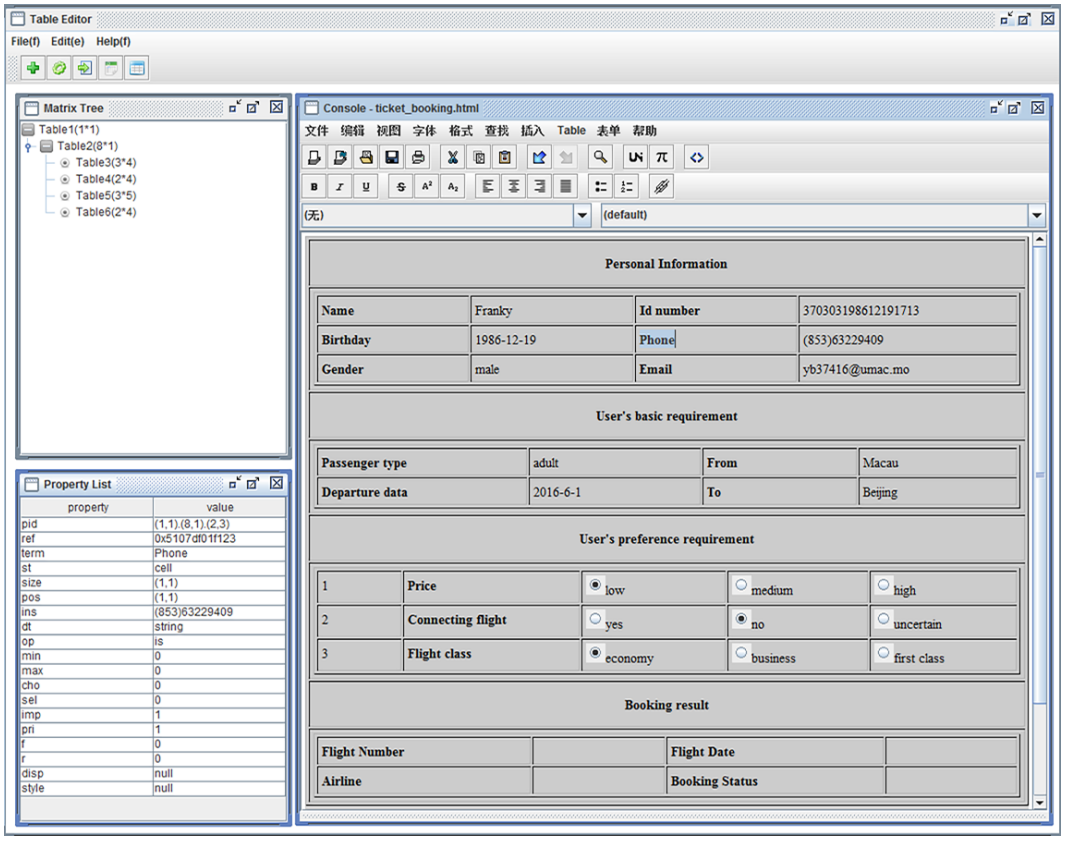


Fig. 7. A Tabdoc document displayed in English.

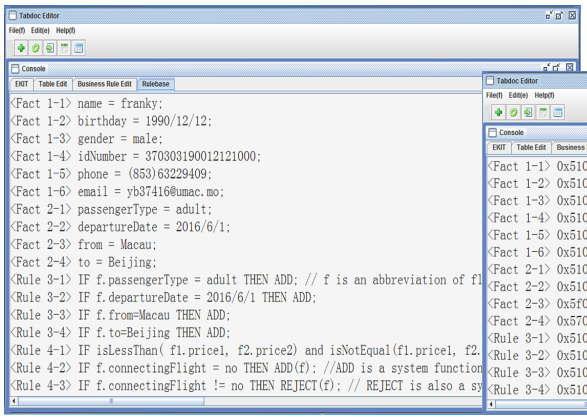


Fig. 8. Rule set.

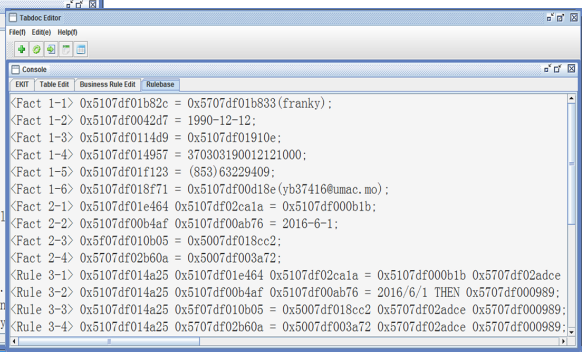


Fig. 9. IID-based rule set.

Item	FY1995	FY2000	FY2003	FY2004	FY2005	FY2006
Domestic						
Letters	24262.9	26114.4	24804.5	23493.5	22666.1	22284.2
Parcels	400.2	310.5	698	1,429.70	2,075.00	2,317.40
International						
Sent	122.8	106	84.2	81.1	77.5	75.7
Letters [1)]	119.9	104.3	82.6	79.6	76.1	74.2
Parcels	2.9	1.7	1.5	1.5	1.5	1.5
Received	280.9	298.1	237.7	215.5	210.9	202.3
Letters [1)]	277.3	295.7	235.6	213.4	208.9	200.4
Parcels	3.6	2.4	2.1	2.1	2	1.8

(a) JAPEN STAT 2007 T078

Year	Area harvested			Production		
	Alfalfa	All other hay	All hay	Alfalfa	All other hay	All hay
	1,000 acres	*1,000 acres*	*1,000 acres*	*1,000 acres*	*1,000 acres*	*1,000 acres*
1993	24,673	35,016	59,689	80,115	66,584	146,699
1994	24,138	34,677	58,815	81,130	69,006	150,136
1995	24,404	35,360	59,764	84,138	70,101	154,239
1996	24,206	36,963	61,169	79,139	70,640	149,779
1997	23,551	37,533	61,084	78,535	74,001	152,536
1998	23,672	36,404	60,076	82,310	69,470	151,780
1999	24,055	39,165	63,220	84,385	75,322	159,707
2000	23,077	36,777	59,854	80,347	71,574	151,921
2001	23,822	39,699	63,521	80,327	76,437	156,764
2002 [1]	23,135	41,362	64,497	73,824	77,138	150,962

(b) USDA_NASS_ASA_2003

December 31,	**2010**	2009
Commercial Airplanes	**$19,663**	$18,616
Boeing Defense, Space & Security:		
Boeing Military Aircraft	**4.025**	4.041
Network & Space Systems	**953**	1.023
Global Services & Support	**1.582**	1.51
Total Boeing Defense, Space & Security	**6.56**	6.574
Boeing Capital Corporation	**3.861**	4.538
Other segment	**937**	872
Unallocated items and eliminations	**34.682**	29.228
Total	**$65,703**	$59,828

(c) BOEING_2010

Fiscal year	Technology Trade				Exports value / Imports value
	Exports		Imports		
	Value (billion yen)	Annual increase rate (%)	Value (billion yen)	Annual increase rate (%)	
1990	339.4	3.0	371.9	12.7	0.91
1995	562.1	21.6	391.7	5.7	1.43
2000	1,057.9	10.1	443.3	8.0	2.39
2002	1,386.8	11.2	541.7	-1.2	2.56
2003	1,512.2	9.0	563.8	4.1	2.68
2004	1,769.4	17.0	567.6	0.7	3.12
2005	2,028.3	14.6	703.7	24.0	2.88

(d) JAPEN_STAT_2007_T074

Fig. 10. Samples of tabular documents.

Dataset 2: it includes 103 Excel/pdf spreadsheets from 8 different real-world sources. These resources are available at http://cells.icc.ru/.

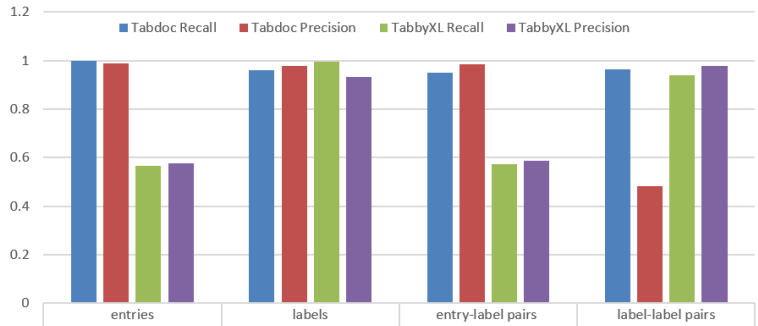
4.2 Experiment on the Extraction of Logical Structure

This experiment is designed to test the performance of our approach in the extraction or identification of logical structure of tabular documents.

Table 2 compares the semantic extraction results between our method and TabbyXL [60] on Dataset 1. In this experiment, the extracted semantic information includes entries, label, entry-label pairs, and label-label pairs, which we will evaluate them based on accuracy and recall. Based on the given standard answer, it can be seen from Fig. 11 that the overall performance of our method is better compared to TabbyXL. This is because our method not only analyzes the style and positional relationships between cells, but also focuses on the semantic relationships between them.

Figs. 12 and 13 present the comparison between Kim's approach [31] and ours on the performance of logical structure extraction on Dataset 2. It is evaluated based on the number of correctly extracted table and cells, respectively. From Figs. 12 and 13, compared with Kim's work, our approach has improved performance in the number of correctly extracted tables and cells. This is due to the fact that our method uses semantic coherency of tabular documents for logical structure extraction rather than relying on visual coherency used in Kim's approach. For example, Kim's approach cannot handle complex tables whose basic table attributes are entirely different or that have sub-tables with a variety of shapes. Besides, it also cannot handle mixed-cell tables which have both an attribute and its value in a cell. However, our approach can identify the logical structure via analyzing semantic relations in a cell or between cells, which results in higher precision.

[1] TabbyXL is a well-designed system for transforming spreadsheet data from arbitrary to relational tables [30].



	entries	labels	entry-label pairs	label-label pairs
Tabdoc Recall	0.99846375	0.95948815	0.94939315	0.96343
Tabdoc Precision	0.9868634	0.97818923	0.9833569	0.48203713
TabbyXL Recall	0.56421286	0.99681365	0.57358867	0.9385774
TabbyXL Precision	0.57554984	0.9331678	0.58553874	0.97926766

Fig. 11. Comparison between Tabdoc and TabbyXL on 10000 tables.

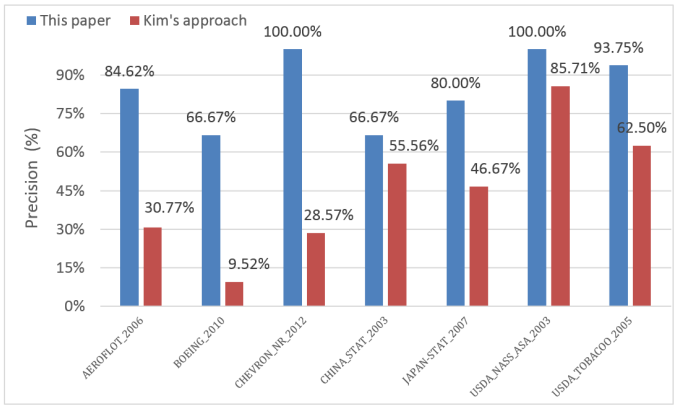


Fig. 12. Comparison on Kim's method [31] and VTree building algorithm (based on tables).

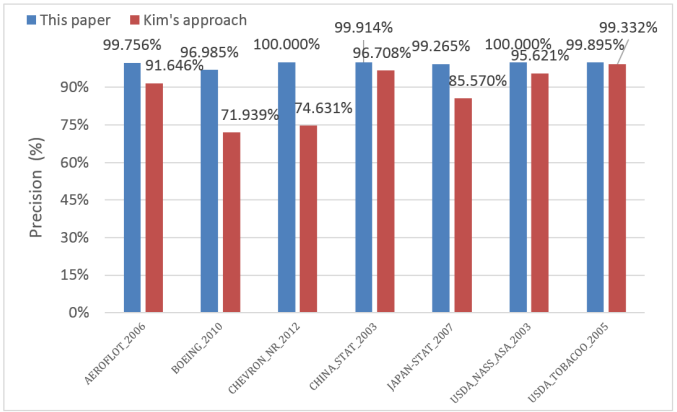


Fig. 13. Comparison on Kim's method [31] and VTree building algorithm (based on cells).

Table 2. Semantic extraction results on different data sets.

		Ours		TabbyXL[1]	
		recall	**precision**	**recall**	**precision**
1000 tables	**entries**	0.99887496	0.9886995	0.557238	0.56586784
	labels	0.9661523	0.98093355	0.99608135	0.9456978
	entry-label pairs	0.9587415	0.9848073	0.5653395	0.57415676
	label-label pairs	0.966807	0.49610612	0.9429573	0.9832521
2000 tables	**entries**	0.99838287	0.9887263	0.55136806	0.5589173
	labels	0.96590227	0.9774259	0.9965263	0.9513356
	entry-label pairs	0.9551515	0.98490626	0.55888325	0.5672646
	label-label pairs	0.96321756	0.48515102	0.937525	0.978406
4000 tables	**entries**	0.9984625	0.9871636	0.5613499	0.57221806
	labels	0.96093625	0.97756535	0.9968091	0.9349015
	entry-label pairs	0.94995564	0.9835748	0.57062995	0.58213043
	label-label pairs	0.9622082	0.48596027	0.9417261	0.9784609
6000 tables	**entries**	0.99837136	0.986387	0.5524677	0.56444687
	labels	0.95850134	0.97882205	0.9964951	0.92856467
	entry-label pairs	0.9495335	0.98349714	0.5624463	0.57497126
	label-label pairs	0.9592783	0.48518354	0.9335942	0.97528535
8000 tables	**entries**	0.9982866	0.9847126	0.5593329	0.5736038
	labels	0.9542778	0.9784198	0.99682605	0.9192822
	entry-label pairs	0.945225	0.9814876	0.57087404	0.5855311
	label-label pairs	0.96478486	0.47499672	0.9403283	0.97854984
10000 tables	**entries**	0.99846375	0.9868634	0.56421286	0.57554984
	labels	0.95948815	0.97818923	0.99681365	0.9331678
	entry-label pairs	0.94939315	0.9833569	0.57358867	0.58553874
	label-label pairs	0.96343	0.48203713	0.9385774	0.97926766

4.3 Experiment on Semantic Consistency Maintenance

This experiment aims to test the extent to which our Tabdoc method consistently conveys the meaning of semantic documents between different contexts. The experiment uses information entropy theory as an evaluation index. In order to calculate the uncertainty of word understanding, Eq. (4.1) gives the definition of the term entropy.

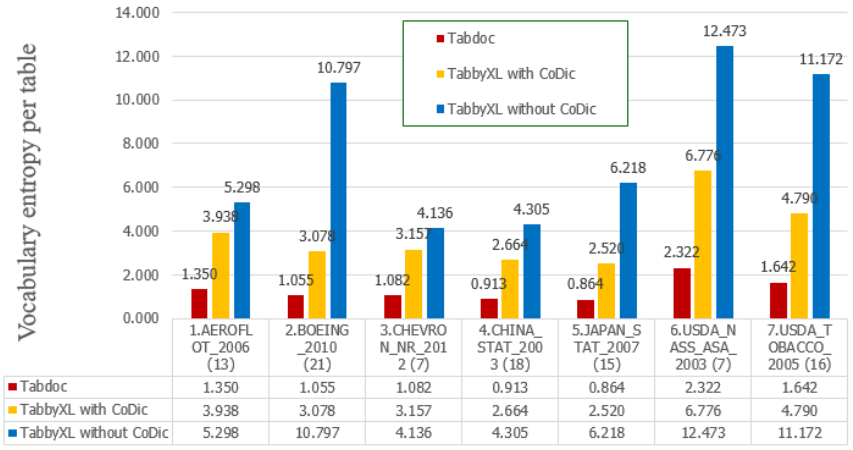
$$Entropy[\log_2(X)] = -\sum_{x \in X} p(x) \log_2 p(x) \tag{4.1}$$

where X is the set of x and $p(x)$ is the probability that the ith possible value of the source information x appears. The magnitude of the value of entropy is positively correlated with the magnitude of the uncertainty. If a word has only one meaning in a semantic document, its term entropy is zero.

Similarly, we define document entropy in Eq. (4.2) to calculate the uncertainty of document comprehension, which is equal to the average term entropy $Entropy_t$ multiplied by the number of terms Num_T, plus the average relational entropy $Entropy_r$ multiplied by the number of relations Num_R.

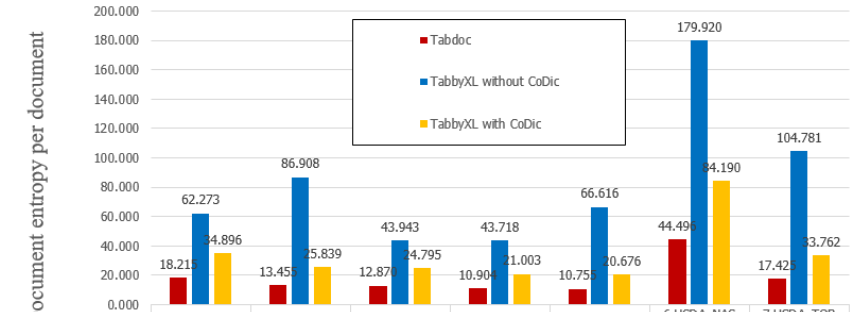
$$Entropy_{doc} = Entropy_t * Num_T + Entropy_r * Num_R \tag{4.2}$$

As can be seen from Fig. 14, the average term entropy in a semantic document represented by Tabdoc is smaller than the average term entropy by TabbyXL. Furthermore, TabbyXL with CoDic has a lower term entropy than TabbyXL without CoDic. This is



	1.AEROFLOT_2006 (13)	2.BOEING_2010 (21)	3.CHEVRON_NR_2012 (7)	4.CHINA_STAT_2003 (18)	5.JAPAN_STAT_2007 (15)	6.USDA_NASS_ASA_2003 (7)	7.USDA_TOBACCO_2005 (16)
Tabdoc	1.350	1.055	1.082	0.913	0.864	2.322	1.642
TabbyXL with CoDic	3.938	3.078	3.157	2.664	2.520	6.776	4.790
TabbyXL without CoDic	5.298	10.797	4.136	4.305	6.218	12.473	11.172

Fig. 14. Comparison of vocabulary entropy in a table on Dataset 1.



	1.AEROFLOT_2006 (13)	2.BOEING_2010 (21)	3.CHEVRON_NR_2012 (7)	4.CHINA_STAT_2003 (18)	5.JAPAN_STAT_2007 (15)	6.USDA_NASS_ASA_2003 (7)	7.USDA_TOBACCO_2005 (16)
Tabdoc	18.215	13.455	12.870	10.904	10.755	44.496	17.425
TabbyXL without CoDic	62.273	86.908	43.943	43.718	66.616	179.920	104.781
TabbyXL with CoDic	34.896	25.839	24.795	21.003	20.676	84.190	33.762

Fig. 15. Comparison of document entropy on Dataset 1.

because, if there is no collaborative dictionary such as CoDic, the level of uncertainty in the meaning of the word will increase.

Fig. 15 compares the document entropy generated by the Tabdoc method and TabbyXL with/without CoDic. Since the semantic relationship is a tuple consisting of two terms and one predicate, and the term entropy of our Tabdoc method is low, the average entropy of the semantic relationship of the Tabdoc method is smaller than the average entropy of TabbyXL. In addition, the more retrieval results of semantic relations, the richer the understanding of semantic documents and the less semantic semantic entropy. As shown in Fig. 15, since the Tabdoc method has a lower term entropy and relational entropy, its average document entropy is much lower than TabbyXL. In summary, Tabdoc has more reliable semantic consistency.

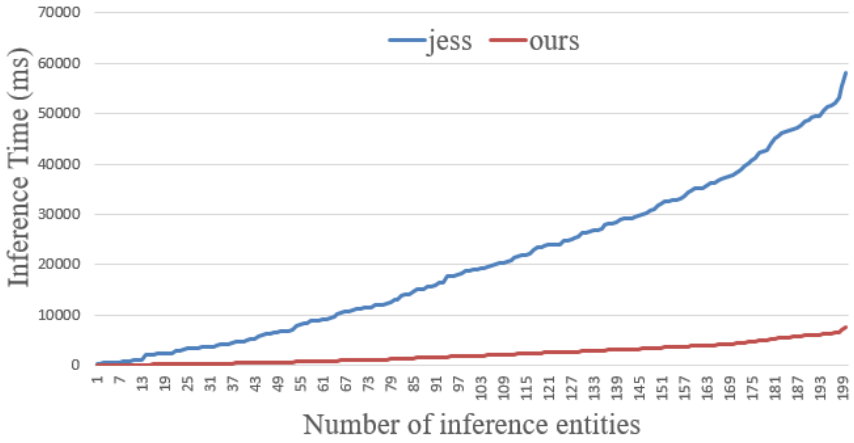


Fig. 16. Inference time comparison on Dataset 1.

4.4 Experiment on the Efficiency of SIA

In order to test the efficiency of the inference algorithm, the experiment compared the inference time of the original Rete algorithm and our improved Rete algorithm for 14791 inference items on Dataset 1. They use the same rules in their respective inference engines. In the preprocessing stage, we batch delete the value parts in the 200 documents, leaving the document template as a query document and the original document instance as a fact database.

As shown in Fig. 16, the experimental results show that the number of query entries greatly affects the inference time of the original Rete algorithm (for example, implemented by Jess). However, the inference time of the improved Rete algorithm increases slowly. When the number of documents from Dataset 1 tends to 200 with 14791 items for semantic inference, the inference time of the improved Rete algorithm is about one-eighth of the inference time of the original Rete algorithm. This shows that our improved Rete algorithm is more efficient than the traditional Rete algorithm in unit time.

Another experiment shows that the increase in the number of facts will raise the inference time. As shown in Table 3, When the number of facts is more than 10^5, the performance of the improved Rete algorithm is far better than that of the orignial algorithm (*e.g.*, implemented by Jess). For example, when executing four queries on 10^5 facts, the time cost of the improved Rete algorithm is approximately one half of that of Jess. Fig. 17 visually shows the trend of the inference time of the two methods.

5. COMPARISON WITH RELATED WORK

Current studies on document representation, as shown in Table 4, use different techniques for representing document syntax, document template and document instance. For better understanding, their limitations, in general, can be summarized in three aspects. First, it's not easy to automatically embed and extract meanings in a document. For example, it is difficult to automatically convert a document written in natural language

Table 3. Comparison of inference time on Dataset 2

Query number	Fact number	Rete_improved	Jess
4 queries	10	1264 ms	1626 ms
	100	1374 ms	1739 ms
	1000	1470 ms	1915 ms
	10000	2155 ms	2391 ms
	100000	3638 ms	8463 ms
Query number	**Fact number**	**Rete_improved**	**Jess**
13 queries	10	1280 ms	1536 ms
	100	1385 ms	1599 ms
	1000	1455 ms	1774 ms
	10000	2143 ms	1872 ms
	100000	3560 ms	8428 ms

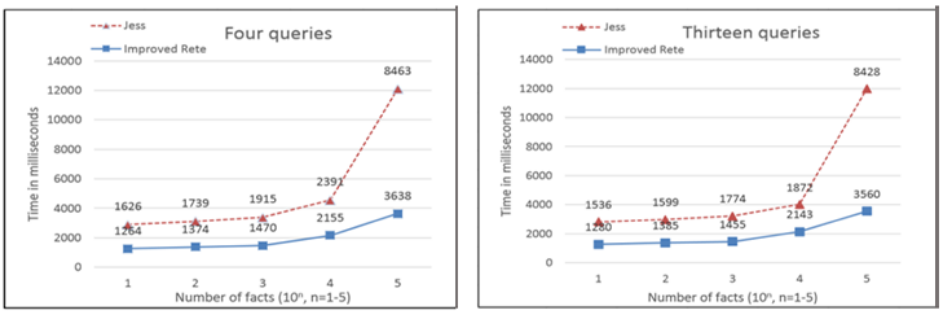


Fig. 17. Inference time comparison between Improve Rete algorithm and Jess on Dataset 2.

to a machine-processable format (such as RuleML [32, 33]). Second, constructing semantic documents needs a lot of extra work. For example, [34] proposes a semantic disambiguation solution to annotate XML documents (*e.g.*, tags and values) by using a machine-readable semantic network (*e.g.*, WordNet) as a common knowledge base. However, it may be time-consuming and sometimes needless because there may be no need to disambiguate unambiguous terms. Third, maintaining semantic consistency between heterogeneous document systems is not easy. For example, to achieve interoperability, [20] requires precise mapping between entities of different ontologies, and [35] requires similarity computation between keywords in a received document and equivalent terms in a domain-wide ontology. Both of them hardly reach a trade-off between low computational demand and effective semantic interoperability.

Compared with existing works, the Tabdoc approach proposed in this paper is novel and can exchange semantic documents across heterogeneous contexts. To achieve it, the Tabdoc approach has adopted a "divide and conquer" strategy to simplify a complex semantic document in levels of vocabulary, relation and document, where each level is context-free.

6. CONCLUSIONS

To facilitate cross-context semantic document interoperation, this paper proposes a novel Tabdoc approach with a "divide-conquer strategy" to separate a complex semantic document into three levels. By using a novel de-contextualization strategy on these

Table 4. Comparison of approaches to document represenation.

Researches	Syntactic document representation	Document template representation	Document instance representation	Application areas	Drawback/critics
Rule format [33, 32]	RuleML to derive document syntax	rules in deontic and defeasible logic	Rule set re-represent documents in specific domains	Contract document [33]; Prediction and decision making [32]	Semantic loss in contexts; inconvenient semantic extraction and representation
Ontology [36, 20, 37, 38] [35, 39, 40]	Metadata [37, 39]; xml schema and ontologies [36, 20, 40]; weighed concept vector [38]; semantic vector [35]	Ontological model	Protégé ontology editor [37]; WickOffice editor [20]; concept vector to depict instances incorporating semantic associations with domain ontology [38, 35]	Searches [36, 37]; proposal writing [20]; document classification and clustering [38, 35, 40]; contract representation [39]	Semantic loss among different contexts; inconvenient semantic representation
XML + Ontology [41, 42, 43, 44]	XML syntax to define ontological specification	Ontological model [41, 42]; semantic Tree [43, 44];	Existed XML documents to be annotated by a common semantic network/ontology	Web services [41]; information retrieval [43, 44]; clinical field [42]	Semantic loss among different contexts; inconvenient semantic representation
Tree/graph [45, 46]	Free text in natural languages	Weighted graph-based structure [45]; parse tree [46]	Weighted-graph representation of instance by TextGraph editor [45]; instance representation by context-free grammar [46]	Text analyzing and text mining	inconvenient semantic representation
Collaborative approach: Conex, CODEX [13, 23, 18]	XPM to derive vocabulary syntax of CONEX and document syntax of CODEX	CODEX to collaboratively create document template	CONEX to collaboratively create vocabulary; CODEX to autonomously create instance	CONEX for vocabulary; CODEX for document	CODEX - template must be collaboratively created; CONEX - great effort to create the vocabulary
Autonomous approach 1 [5]	Docsyn of Dosign for document syntax and semantic structures	Docsem of Dosign for SEBD schema on SFASFA editor	Ussign approach to derive SEBD instance on SFASFA editor	Document	Document constrained by a triple model and lacks of flexibility
Autonomous approach 2 (This paper)	Tabdoc Grammar	Tabdoc Template	Autonomously create instance documents via Tabdoc Editor	Document	Each document template is individually designed

three levels, complexity and cross-context problems have been resolved. In our approach, Tabdoc Grammar has been devised as a universal semantic document syntax, which is context-free and XML-compatible. This ensures any document can be syntactically interoperable between contexts. By composing a semantic document as a semantic matrix tree, that is, a set of hierarchically arranged tables that depict semantic relations among common concepts, the semantic document becomes context-free for semantic interoperability. Each common concept is denoted by a unique concept internal identifier (IID) in CoDic, which is a dictionary collaboratively designed by concept designers and easily used by users without semantic ambiguity. Complexity conceptualization strategy assigns each compound concept with a type of semantic relation (CCT), which is also denoted by

a unique IID in CoDic. For document understanding, the newly proposed semantic interpretation algorithm automatically parses the meaning of a Tabdoc document via an improved Rete algorithm for semantic inference. In summary, the approach proposed in this paper has the following expectations.

- It is the first time a matrix tree is proposed to represent a complex semantic document. Such a representation simplifies semantic documents and provides a theoretical methodology to semantic document study.
- A universal document grammar is offered that can be implemented in XML. This provides a syntactic foundation for semantic document interoperability.
- It represents the semantics of a document as different types of rules, which enables a user to create and modify semantics easily and to be adaptable to various changes in semantic interpretation. Therefore, it meets the frequently changing requirements in industrial applications and increases the flexibility of semantic representation and system development.

The newly proposed Tabdoc approach has shed light on collaboration in industries and in e-business. It has provided a paradigm according to which semantic documents can be edited, read, exchanged and processed without rigid standards. In the future, we will apply the Tabdoc approach to more web-based applications (for example, cross-context interaction between devices in the IoT). In addition, it is necessary to extend Tabdoc to a more comprehensive language to accommodate more complex semantic documents.

ACKNOWLEDGMENT

This research is supported by both the National Natural Science Foundation of China (Grant No. 61802079) and the Guangzhou University (Grant No. 2900603143).

REFERENCES

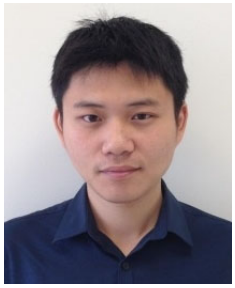
1. Y. Y. Tang, H. Ma, D. Xi, X. Mao, and C. Y. Suen, "Modified fractal signature (mfs): a new approach to document analysis for automatic knowledge acquisition," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 9, 1997, pp. 747-762.
2. E.-T. Lin and C. Zhou, "Modeling and analysis of message passing in distributed manufacturing systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 29, 1999, pp. 250-262.
3. J. Liu, K. He, and D. Ning, "Web service aggregation using semantic interoperability oriented method," *Journal of Information Science and Engineering*, Vol. 28, 2012, pp. 437-452.
4. X. Li, Y. Fan, Q. Z. Sheng, Z. Maamar, and H. Zhu, "A petri net approach to analyzing behavioral compatibility and similarity of web services," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 41, 2011, pp. 510-521.
5. G. Xiao, J. Guo, Z. Gong, and R. Li, "Semantic document exchange for e-business: Trends and issues," in *Proceedings of IEEE 12th International Conference on e-Business Engineering*, 2015, pp. 147-153.

6. P. Qin and J. Guo, "A novel machine natural language mediation for semantic document exchange in smart city," *Future Generation Computer Systems*, Vol. 102, 2020, pp. 810-826.
7. G. Xiao, J. Guo, Z. Gong, and R. Li, "Semantic input method of chinese word senses for semantic document exchange in e-business," *Journal of Industrial Information Integration*, Vol. 3, 2016, pp. 31-36.
8. J. Luo, B. Meng, C. Quan, and X. Tu, "Exploiting salient semantic analysis for information retrieval," *Enterprise Information Systems*, Vol. 10, 2016, pp. 959-969.
9. F. Ramli and S. A. M. Noah, "Building an event ontology for historical domain to support semantic document retrieval," *International Journal on Advanced Science, Engineering and Information Technology*, Vol. 6, 2016, pp. 1154-1160.
10. J.-U. Heu, I. Qasim, and D.-H. Lee, "Fodosu: multi-document summarization exploiting semantic analysis based on social folksonomy," *Information Processing and Management*, Vol. 51, 2015, pp. 212-225.
11. B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. Ngu, and A. K. Elmagarmid, "Business-to-business interactions: issues and enabling technologies," *The International Journal on Very Large Data Bases*, Vol. 12, 2003, pp. 59-85.
12. Y. Yan, J. Zhang, and M. Yan, "Ontology modeling for contract: Using owl to express semantic relations," in *Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference*, 2006, pp. 409-412.
13. J. Guo, "Inter-enterprise business document exchange," in *Proceedings of the 8th International Conference on Electronic Commerce: The New e-Commerce: Innovations for Conquering Current Barriers, Obstacles and Limitations to Conducting Successful Business on the Internet*, 2006, pp. 427-437.
14. Y. Charalabidis, F. Lampathaki, and D. Askounis, "Unified data modelling and document standardization using core components technical specification for electronic government applications," *Journal of Theoretical and Applied Electronic Commerce Research*, Vol. 3, 2008, pp. 38-51.
15. Y. Kabak and A. Dogac, "A survey and analysis of electronic business document standards," *ACM Computing Surveys*, Vol. 42, 2010, p. 11.
16. M. Erdmann and R. Studer, "How to structure and access xml documents with ontologies," *Data and Knowledge Engineering*, Vol. 36, 2001, pp. 317-335.
17. J. Shen and Y. Yang, "Extending rdf in distributed knowledge-intensive applications," *Future Generation Computer Systems*, Vol. 20, 2004, pp. 27-46.
18. J. Guo, "Collaborative conceptualisation: towards a conceptual foundation of interoperable electronic product catalogue system design," *Enterprise Information Systems*, Vol. 3, 2009, pp. 59-94.
19. J. Guo, L. Xu, Z. Gong, C.-P. Che, and S. S. Chaudhry, "Semantic inference on heterogeneous e-marketplace activities," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 42, 2012, pp. 316-330.
20. M. Rico, M. Taverna, M. Caliusco, O. Chiotti, M. Galli *et al.*, "Adding semantics to electronic business documents exchanged in collaborative commerce relations," *Journal of Theoretical and Applied Electronic Commerce Research*, Vol. 4, 2009, pp. 72-90.
21. R. da Silva Machado, R. B. Almeida, D. Y. L. da Rosa, J. L. B. Lopes, A. M. Pernas, and A. C. Yamin, "Exehda-hm: A compositional approach to explore contextual in-

formation on hybrid models," *Future Generation Computer Systems*, Vol. 73, 2017, pp. 1-12.
22. K. Taha and R. Elmasri, "Xcdsearch: An xml context-driven search engine," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, 2010, pp. 1781-1796.
23. J. Guo, *Collaborative Concept Exchange*, VDM Publishing, Germany, 2008.
24. J. Kim, D. Jeong, and D.-K. Baik, "A translation algorithm for effective rdb-to-xml schema conversion considering referential integrity information," *Journal of Information Science and Engineering*, Vol. 25, 2009, pp. 137-166.
25. F. De Saussure, W. Baskin, and P. Meisel, *Course in General Linguistics*, Columbia University Press, NY, 2011.
26. C. S. Khoo and J.-C. Na, "Semantic relations in information science," *Annual Review of Information Science and Technology*, Vol. 40, 2006, p. 157.
27. S. Yang, J. Guo, and R. Wei, "Semantic interoperability with heterogeneous information systems on the internet through automatic tabular document exchange," *Information Systems*, Vol. 69, 2017, pp. 195-217.
28. S. Yang and J. Guo, "A novel approach for cross-context document reasoning in e-commerce," in *Procedings of the 6th IEEE International Conference on Software Engineering and Service Science*, 2015, pp. 1018-1025.
29. Z. Ma and X. Wang, "Rule interchange in the semantic web," *Journal of Information Science and Engineering*, Vol. 28, 2012, pp. 393-406.
30. A. O. Shigarov, "Table understanding using a rule engine," *Expert Systems with Applications*, Vol. 42, 2015, pp. 929-937.
31. Y.-S. Kim and K.-H. Lee, "Extracting logical structures from html tables," *Computer Standards and Interfaces*, Vol. 30, 2008, pp. 296-308.
32. A. Tsadiras and N. Bassiliades, "Ruleml representation and simulation of fuzzy cognitive maps," *Expert Systems with Applications*, Vol. 40, 2013, pp. 1413-1426.
33. G. Governatori, "Representing business contracts in ruleml," *International Journal of Cooperative Information Systems*, Vol. 14, 2005, pp. 181-216.
34. S. Decker, S. Melnik, F. Van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks, "The semantic web: The roles of xml and rdf," *IEEE Internet Computing*, Vol. 4, 2000, pp. 63-73.
35. R. Costa and C. Lima, "Knowledge representations with ontology support for collaborative engineering in architecture engineering and construction," *Journal of Information Technology in Construction*, Vol. 19, 2014, pp. 434-461.
36. L. Carr, T. Miles-Board, A. Woukeu, G. Wills, and W. Hall, "The case for explicit knowledge in documents," in *Proceedings of ACM Symposium on Document Engineering*, 2004, pp. 90-98.
37. H. Eriksson, "The semantic-document approach to combining documents and ontologies," *International Journal of Human-Computer Studies*, Vol. 65, 2007, pp. 624-639.
38. L. Bing, S. Jiang, W. Lam, Y. Zhang, and S. Jameel, "Adaptive concept resolution for document representation and its applications in text mining," *Knowledge-Based Systems*, Vol. 74, 2015, pp. 1-13.
39. V. Rodríguez-Doncel, J. Delgado, S. Llorente, E. Rodríguez, and L. Boch, "Overview of the mpeg-21 media contract ontology," *Semantic Web*, Vol. 7, 2016, pp. 311-332.
40. G. Boella, L. di Caro, L. Humphreys, L. Robaldo, P. Rossi, and L. van der Torre, "Eunomos, a legal document and knowledge management system for the web to provide

relevant, reliable and up-to-date information on the law," *Artificial Intelligence and Law*, Vol. 24, 2016, pp. 245-283.
41. K. Thirunarayan, "On embedding machine-processable semantics into documents," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, 2005, pp. 1014-1018.
42. C. Sáez, A. Bresó, J. Vicente, M. Robles, and J. M. García-Gómez, "An hl7-cda wrapper for facilitating semantic interoperability to rule-based clinical decision support systems," *Computer Methods and Programs in Biomedicine*, Vol. 109, 2013, pp. 239-249.
43. J. Tekli, N. Charbel, and R. Chbeir, "Building semantic trees from xml documents," *Web Semantics: Science, Services and Agents on the WWW*, Vol. 37, 2016, pp. 1-24.
44. J. Tekli, "An overview on xml semantic disambiguation from unstructured text to semi-structured data: Background, applications, and ongoing challenges," *IEEE Tran- sactions on Knowledge and Data Engineering*, Vol. 28, 2016, pp. 1383-1407.
45. H. Mousavi, D. Kerr, M. Iseli, and C. Zaniolo, "Mining semantic structures from syntactic structures in free text documents," in *Proceedings of IEEE International Conference on Semantic Computing*, 2014, pp. 84-91.
46. M.-C. de Marneffe *et al.*, "Generating typed dependency parses from phrase structure parses," in *Proceedings of the Language Resources and Evaluation Conference*, Vol. 6, 2006, pp. 449-454.

Shuo Yang received the Master's degree in Software Engineering from the Dalian Jiaotong University, China, in 2013. He was awarded a doctorate degree in Software Engineering, University of Macau, in 2017. Currently, he is a Lecturer in Guangzhou University. His research interests include semantic interoperability and semantic inference with AI technology, mainly applied to the fields of e-commerce, e-marketplace and clinical area.



Ran Wei received the doctorate degree in Bio-Informatics from Rutgers University, Newark, New Jersey, USA, in 2018. He received the Master's degree in Computer Science from University of California, Irvine, USA, in 2019. His research interests include bio-informatics and artificial intelligence.