A language-based approach to document modeling

Yves MARCOUX

GRDS - EBSI - Université de Montréal
C.P. 6128, succ. Centre-ville
Montréal (Québec)
CANADA H3C 3J7
yves.marcoux@umontreal.ca

Abstract

We introduce the notion of *language-based* approaches to document modeling, and describe one particular form, which we call *document-instance standalone legibility* (DISL), based on the ideas that the interweaving of markup and contents in a structured document instance make up "quasi-sentences", and that those quasi-sentences can be directly meaningful to the community of persons dealing with the documents (contents creators, stylers, readers, etc.). Through an example, we show that DISL is possible and that it is different from *descriptive markup*, the approach to markup aimed at maximizing the possibilities of contents reuse. We discuss the implications of DISL for language-based document modeling.

Categories and Subject Descriptors

I. Computing Methodologies / I.7 DOCUMENT AND TEXT PROCESSING / I.7.2 Document Preparation / Markup languages

1. Introduction

1.1 Meaning of a document

What is the meaning of a document? Let us downsize a bit: what is the meaning of a document in a given context, for a given community? Writing it out in mathematics—as did, for example, Kantor (Kantor 2002)—, we are asking: what function S, possibly depending on a context and a community, is such that S(*d*) is the agreed-upon "meaning" (or semantics) of document *d* in the community and context, for all syntactically valid document *d*? This raises the question: what is the *range* (image) of S? This is, of course, the Pandora box. But is there a way to retain something of this simplistic approach to the question, without opening the box?

There are situations where the notion of meaning of a document is irrelevant and does not need to be known or even defined; it could be because the consultation experience is purely sensorial (for example, when enjoying the esthetics of text written in a language we don't understand or visually checking the page breaks in a proof document), or because the experience is intended to exploit imprecision (trial-and-error sketches of an idea during a brainstorming session).

However, there are other situations where a document is definitely intended to mean something precise, no matter how complex and sophisticated (customizable, etc.) the rendering interface might be. This is the case, for example, with legal documents (checks, contracts, legislation, etc.), records (minutes, reports, etc.), and informative documents (newspaper articles, memoranda, etc.). In those cases, it should be possible to establish fairly precisely the "face value" of the document: all and only what the document is intended to mean, regardless of the interface used.

What we propose, in those cases, is to define an agreed-upon function S (dependent on the document type, or maybe *genre*) that will try, not to give the "meaning" of the document in some absolute metaphysical sense, but simply to give a first interpretation of the document that is as precise as possible/desirable for the members of the target community, and independent of any specific interface used to access the document. Instead of trying to put a lot of intelligence in S, we make it simple, but assume that its output will be intelligently interpreted, namely by humans. We will in fact be pushing most of the complexity out of S and into the *interpretation* of its output.

So we suggest the following intepretation "workflow":

d ==(S)==> S(*d*) ==(human)==> "final" meaning of *d*

The "==(S)==>" represents the totally automatic transformation by S, and the "==(human)==>" represents the human interpretation. Note that it still leaves open the possibility that *d* have different meanings for different humans (or for different readings by the same human), which is nice, because otherwise, the approach would have to be dismissed immediately for being too simplistic. We are in fact introducing a first level of interpretation, that is totally mechanical, but paves the way to the "real" interpretation: a second-level interpretation by humans. That second-level interpretation, we leave totally open.

Have we gained anything?

One first advantage of having defined S (we suppose it is computable and we have access to its code) is that we might, by examining its code, find out, for example, that the meaning of a document does not depend on certain of its parts; for instance, S might totally ignore the comments in the document, or even other parts.

Also, since S is agreed-upon among the community members, we will now have the possibility of writing down S(*d*) and of considering it the "face-value" interpretation of *d*. The value S(*d*) is a precise mathematical object, morphologically simple and easy to manipulate symbolically (e.g., by computer); yet, it can still have as subtle, convoluted and/or vague "meanings" as desired, once digested by humans. The level of precision to which S(*d*) goes is entirely dependent on the level of clarity and unambiguity deemed desirable/necessary by the members of the target community.

We have not talked about range(S) yet. The key point is that S(*d*) can contain free-style natural language. As an illustration, we can think of S(*d*) as of a character string (or text file). In general, it could be anything directly interpretable by humans. Note that S(*d*) is intended to serve as a reference interpretation of *d*, not as an ergonomic presentation for humans. In fact, S(*d*) may never actually be displayed to any human in a document lifecycle; but it is there as a reference, maybe to help settle disputes about what the document really means (or meant) at face value. It can also serve as a check at document-creation time, to make sure that what we key-in really means what we think it means.

As an example, suppose we are dealing with company memoranda in XML. Then, S could be a XSLT stylesheet comprising templates that only add text before and/or after element contents. An element like `<DATE>2009-09-10</DATE>` could then be transformed into something like the string `"The date on which this memorandum was--or is to be--sent is: 2009-09-10 (YYYY-MM-DD format)."`. A complete memo would be transformed into a sequence of such sentences.

Of course, if S is going to serve as a reference for the community, it better be known to all ("No one is supposed to ignore the law", as it goes!). Thus, S has to be established and agreed-upon at document-modeling time. Among other things, its definition (algorithm) reflects in an explicit and precise way the postulates and inferences made during document analysis. Note that it makes sense that the output of the S is not presentation-oriented, because document engineers are not expected to be layout/interface specialists. The challenge in designing reading/editing/presentation interfaces for the document will be to "convey" as much as possible of S(*d*), without actually showing it to the user, which would probably be very anti-ergonomic.

1.2 Language-based document modeling

Including the definition of a semantic function (S) as an integral part of the document-modeling process, with the interpretation workflow presented above in mind, is what we call a *language-based approach* to document modeling. Why *language-based*? Intuitively, because S(*d*) must be a "sentence" in some sort of language amenable to direct human interpretation.

Let us compare the notion of "meaning of a document" implicit in language-based approaches with that found in typical approaches. How is the meaning of a document established in a typical structured-document system? In general, the meaning of the various components (elements, attributes, entities, etc.) of the document model (DTD, schema, etc.) is explained in the form of *comments* in the model. Thus, a raw document instance can be interpreted by consulting the relevant explanations for each component (elements, attributes, entities, etc.) encountered. Making the connection between the instance and the explanations is left to the human trying to understand the document. Can we define an S that will do as good as that? We just make S output the concatenation of the document instance (*d*) and whatever documentation (DTD, schema, etc.) is available to humans to make sense of *d* (with maybe a few separators, to make things clear). So, we haven't lost anything by introducing our extra level of interpretation. But can we do better?

Here is an intuitive and informal reason why we think that, yes, we can probably do better. If we consider the sense-making apparatus of the typical approach mentioned above, we see that it relies heavily on the "table-lookup" mechanism. *You want to know what a DATE element means? Just look it up in the "explanations" section.* We believe that this creates a "cognitive distance" between the data and its explanation, distance that is not only present and acting at document-reading time, but also at document-creation time, at style-design time, and even in the modeling process itself (for example, during interviews/negociations with the target community). We suspect such a cognitive distance could be behind a lot of ambiguity, misunderstanding, fuzziness situations observed in structured-document and database systems. We believe that presenting—or at least, envisioning—the interpretation of pieces of data as a natural—or quasi-natural—language sentence, that has a sequentiality to it, maybe a "subject-verb-complement" shape, might diminish that cognitive distance in the design and construction of systems, and improve their usability.

1.3 This article

In this article, we consider a very restricted form of language-based document modeling: the case where S is the *identity function*. That is, we ask ourselves if there are circumstances in which a raw structured-document instance can in itself be an adequate intermediate in the interpretation workflow presented above. It turns out that the answer is yes, at least in certain cases. This is in itself intriguing, and we think it might be indicative of the possibilities of language-based document modeling.

Observe that if a document instance is by itself a sufficient basis for human derivation of meaning, then it must be intelligible (in proper context) without any other source of information. Thus, the remainder of this article is about *document-instance standalone legibility* (DISL). Although this phenomenon bears resemblance to *descriptive markup* and to the "use meaningful names" exhortation found in modeling methodologies, it has never been, to our knowledge, directly studied.

In this article, we will:

1. define DISL;
2. show that DISL is different from descriptive markup;
3. discuss the implications of DISL for language-based document modeling.

The remainder of the article is divided into four sections, the first three of which correspond to the above plan, and the last of which is a short conclusion.

---

2. Document-instance standalone legibility (DISL)

Any markup model (DTD, schema, etc.) determines three things about markup, which are seldom clearly distinguished from one another: *where* markup is used, *why* it is used, and *what* it looks like. The answer to *where* determines the possible topologies (tree structure) of a document, the answer to *why* gives meaning to the various parts of a document, and the answer to *what* determines exactly how the various markers (element, attribute, entity names, etc.) are spelled out in document instances.

"Standard" modeling methodologies (Maler & El Andaloussi 1996; Salminen et al. 1997; Glushko & McGrath 2002) concentrate on the first two components of markup, but do not say much about the third question, that of choosing the exact labeling of the markers (elements, attribute, entity names, etc.) that will be included in the actual document instances.

It could be argued that the *what* question is not very important; after all, it can be linked to considerations as frivolous (though sometimes important) as constraints on storage space or character count (Maler & El Andaloussi 1996, pp.246-7). Also, very simple processing mechanisms (e.g., a stylesheet) can interchange one set of markers with another quite easily. Yet, a judicious choice of markers can bring at least two benefits: increasing the clarity and simplicity of the descriptions of the meaning of the markup (the *why*), and increasing the *standalone legibility* of document instances. For example, it is much simpler and less error-prone to explain to a (English-speaking) community that an element NAME contains a subelement SURNAME followed by a subelement FIRSTNAME, than to say that a Y78 contains a Y79 followed by a Y80, and then explain what each one means. Similarly, <MEMO SCOPE="internal"> is fairly legible without any styling or external explanation, but <Y81 Y82="y83"> is not, although they could mean exactly the same thing.

Let us consider the following analogy. In a relational table, the field names can be understood as column headings and, as such, can be "distributed" to each line of the table. Such "distribution" occurs, for instance, when a record is exposed through the default basic viewing form of the DBMS, in which field names are interspersed with the actual data from a single line of the table. The result of such distribution can be viewed as a sentence from a very rudimentary and restricted language, composed of the database name, the table name, the field names, and the actual record data. For example, if we extract just the succession of words displayed on a basic viewing screen for an employee record, we may end up with the following rudimentary sentence: "Company XYZ; Database: Human Resources; Table: Employees; ID: 1234; Surname: Doe; Firstname: John; City: Montreal; Salary: 50000." (We have added punctuation to represent visual separators such as line breaks and box boundaries.) This sentence clearly falls short of being correct English, but undoubtedly exhibits linguistic properties; for instance, it will be understood by most English speakers, and not understood by most English non-speakers. Let us call such a sentence a *quasi-sentence*.

Note that the quasi-sentence in the example above is the result of collaborative work, performed in (at least) two different times: some words (database and field names) are chosen by the database designer(s) while some others (data) are chosen by contents creator(s). The *order* of appearance of the words is determined by the order of presentation of the fields, which, in a default system-generated form, would be the order of definition of the fields in the table, and thus would also have been established by the database designers. Hence, significant properties of the quasi-sentence are determined at database-design time.

A raw (unformatted) structured document can also be viewed as a quasi-sentence, resulting from the intertwining of model markers (markup) and element contents. The model markers are determined at document-modeling time, and the actual attribute values and element contents are determined at contents-creation time. In contrast to the relational database situation described above, the exact topology of the document (e.g., number of occurrences of repeatable or optional elements) is also determined at contents creation time and, because of the possibility of mixed contents (text segments intertwined with subelements), the interweaving of markup and textual contents can be much more intricate. Also, inasmuch as structured documents contain free-style whole-sentence natural-language contents more "often" than relational databases, quasi-sentences are more likely to comprise passages in free-style whole-sentence natural language in a structured-document setting than in a relational-database setting.

When a raw document instance, without any styling or external explanation, constitutes a meaningful quasi-sentence for the members of some well-defined target community, we say that it has *standalone legibility*, with respect to that community. Standalone legibility is, of course, an imprecise notion; it is not an all-or-nothing phenomenon, it can be attained to a degree.

Note that a quasi-sentence is never intended to be a completely standalone source of information; it is always expected to be interpreted in some specific operational context, if it is to actually "say" to someone something about the real world. First,

it must be presented to a person with adequate knowledge and abilities, but it must also be obtained by that person in response to some operation performed on the actual production version of an information system, securely operated by some known entity, and accessed through a secure connection. Otherwise, the person would not "believe" what the quasi-sentence is saying. But, in the right context, it makes sense to say that the quasi-sentence "means" something to someone. In the remainder of this article, when we say "meaning" of a sentence or of a quasi-sentence, we will always assume that the sentence is produced in a proper operational context, and presented to a person belonging to the target community of the system that produces it. In particular, standalone legibility of document instances (DISL) is always understood to be with respect to the members of some well-identified target community and in a proper operational context.

It should be noted that, although the notion of DISL is most natural and easy to understand when document contents have a "word-like" nature (words, numbers, URLs, etc.), it does not need contents to be free-style natural-language text to make sense. In particular, as we hope the above example has shown, it makes sense even with "data-oriented" database-like markup. It can be argued that, even if contents components are allowed to be multimedia (which of course is not possible in raw XML document instances), DISL still makes sense.

Finally note that we are not suggesting that DISL is necessarily a desirable thing in document modeling. We are studying it solely as an particular case of language-based approaches.

---

3. Descriptive markup and DISL

3.1 Descriptive markup

Descriptive (or *semantic* or *declarative*) markup is a kind of markup in which the meaning attached by the model designer to the different markers of the model is related to the *intrinsic nature* of the information conveyed by the various parts of the document, as opposed to *how* they should be processed by an application. Reuse (or "repurposing") of documents or document parts is facilitated because the markers are not "biased" towards—or tailored to—one particular application, becoming obstacles for all the others. Most experts agree that the benefits of structured documents—notably, the possibility of contents reuse—are more important when descriptive markup is used. Thus, descriptive markup is perceived as something desirable and, not surprisingly, it is central (though not always explicitly so) to all standard document modeling methodologies (Maler & El Andaloussi 1996; Salminen et al. 1997; Glushko & McGrath 2002).

3.2 DISL and descriptive markup are not the same

The following example should suffice to demonstrate that DISL and descriptive markup are not the same. We present two different models for company memos, with one conforming document instance for each. The models are entirely isomorphic up to the choice of markers. Thus, they share common answers to the *where* and *why* questions. These common answers respect the descriptive markup principle. However, the models differ on their answers to the *what* question: the first one uses minimal markers while the second one uses markers especially aimed at DISL. We present the models in DTD form, for simplicity and conciseness. Undeclared elements should be understood to have #PCDATA (text only—no subelement) contents.

First model, with minimal markers:

```
<!ELEMENT m (a, t, s, b)>
<!ELEMENT t (p+)>
<!ELEMENT b (g+)>
<!ATTLIST a a CDATA #IMPLIED>
<!ATTLIST n a CDATA #IMPLIED>
```

Second model, with markers chosen for DISL:

```
<!ELEMENT this-memorandum (authored-
by-a-person-named, is-addressed-to,
  its-subject-is, it-contains-the-
following-paragraphs)>
<!ELEMENT is-addressed-to (a-person-
named+)>
<!ELEMENT it-contains-the-following-
paragraphs (paragraph+)>
<!ATTLIST authored-by-a-person-named
whose-email-address-is CDATA #IMPLIED>
<!ATTLIST a-person-named whose-email-
address-is CDATA #IMPLIED>
```

Instance conforming to the first model:

```
<m>
  <a>Rick Mercer</a>
  <t><p>Janet Baker</p>
     <p a="kr@my.com">Kile
Ryan</p></t>
  <s>The company party</s>
  <b><g>Be there! Or else...</g></b>
</m>
```

Instance conforming to the second model:

```
<this-memorandum>
  <authored-by-a-person-named>Rick
Mercer</authored-by-a-person-named>
  <is-addressed-to>
      <a-person-named>Janet Baker</a-
person-named>
      <a-person-named whose-email-
address-is="kr@my.com">Kile Ryan</a-
person-named>
  </is-addressed-to>
  <its-subject-is>The company
party</its-subject-is>
```

```
  <it-contains-the-following-
paragraphs>
      <paragraph>Be there! Or
else...</paragraph>
  </it-contains-the-following-
paragraphs>
</this-memorandum>
```

Although the two instances "say" exactly the same thing, it is likely that the second one has greater standalone legibility than the first one, at least for most English-speaking communities. Note that our point is only to show that the choice of markers influences DISL, not to suggest that the longer the markers, the better the DISL. In practice, less "extreme" markers (such as "memo", "author", etc.) would probably yield a comparable degree of DISL.

3.3 DISL and meaningful markers

We note that DISL is more than just "using meaningful markers", which is usually how deep modeling methodologies go into treating the *what* question. DISL implies using meaningful markers, but goes further; the document instance, as a whole quasi-sentence, has to make sense for the target community.

3.4 DISL and linguistic bias

It is important to note that DISL does not necessarily imply a linguistic bias in the model. DISL forces recognition of a linguistic reality, but not necessarily unilinguism. Multilinguism could be accomodated in DISL, for example by allowing more than one marker-set in the same model (e.g., with a choice "|" content model at the top level). DISL does however go against the approach of using language-neutral markers (usually numbers), as in the MARC library-catalog formats. In the cases where a language-neutral approach is justified and required, DISL is not possible.

---

4. Discussion

As we have pointed out, there is no reason why DISL should be an objective in document modeling; at the same time, it is an intriguing phenomenon that suggests something fundamental is at stake. But why is it intuitively "right" that S be as simple as possible (the identity function in the case of DISL)? We risk the following informal and intuitive answer. Descriptive markup and contents reuse are based on the idea that documents are informational assets, worth preserving and enriching. If this is the case, is it not natural, then, that documents contain most, if not all, of their value in themselves, independently of outside resources? Certainly, being interpretable by the members of their target community is a significant part of their value, hence, the natural character of DISL. In a way, being able to make do with a simple S "confirms" that the modeling is good.

Being the identity function is obviously a very strong restriction for S; a natural class of functions to study would be, we think, those corresponding to "simple" XSLT stylesheets, possibly with certain morphological restrictions, such as comprising only templates that add text before and/or after element/attribute contents, and perhaps output something when optional elements/attributes are absent.

We believe the notion of language-based document modeling raises many interesting questions. For example, our slight incursion in DISL suggests the following questions:

1. Can a language-based approach be used to define descriptive markup more precisely?
2. If so, is it the case that descriptive markup + significant markers = DISL?
3. Without opening the Pandora box, can we somehow restrict or enrich the range of S to capture more sophisticated aspects of the document semantics? In particular, aspects that would give hints on the required characteristics of adequate presentation interfaces?
4. How would the extension of the range of S to include multimedia components (still and moving images, sound, etc.) affect the sense-making process? Does precision necessarily suffer from expressivity?
5. Can *genre* theory (e.g., Bazerman 1988) be helpful to the document engineer for defining S? Not only can range(S) in itself be considered a genre, but it can adopt communication strategies that exploit other genres, already mastered by the target community (tables with look-up keys being an example of a genre mastered by most people).
6. In general, how should the specification of S interact with the other aspects of document modeling?

---

5. Conclusion

In this article, we introduced the notion of *language-based* approaches to document modeling, and described one particular form, called *document-instance standalone legibility* (DISL), based on the ideas that the interweaving of markup and contents in a structured document instance make up "quasi-sentences", and that those quasi-sentences can be directly meaningful to the community of persons dealing with the documents (contents creators, stylers, readers, etc.). Through an example, we showed that DISL is possible and that it is different from descriptive markup. We then discussed the implications of DISL for language-based document modeling. Finally, we identified a number of questions about language-based document modeling suggested by our analysis of DISL.

References

Bazerman, C. *Shaping Written Knowledge: the Genre and Activity of the Experimental Article in Science*. Madison, WI: University of Wisconsin Press, 1988.

Glushko, Robert J.; McGrath, Tim. *Document Engineering for e-Business*, *Proceedings of the 2002 ACM Symposium on Document Engineering, McLean, Virginia, USA, November 8-9, 2002*, ACM, 2002, <http://doi.acm.org/10.1145/585067>.

Kantor, Paul B. *Setting a theoretical foundation for library and information science*. 2002, <http://www.scils.rutgers.edu/~kantor/601/theory.htm>.

Maler, Eve; El Andaloussi, Jeanne. *Developing SGML DTDs: From Text to Model to Markup*. Prentice Hall PTR, 1996.

Salminen, Airi; Kauppinen, Katri; Lehtovaara, Merja. "Towards a methodology for document analysis." *Journal of the American Society for Information Science*, vol. 48, no. 7, pp. 644-655, July 1997.