Extracting Document Structure to Facilitate a Knowledge Base Creation for The UML Superstructure Specification*

Mehrdad Nojoumian
School of Information Technology and Engineering, University of Ottawa
mnojoumi@site.uottawa.ca

Timothy C. Lethbridge
School of Information Technology and Engineering, University of Ottawa
tcl@site.uottawa.ca

Abstract

The research presented in this paper aims at facilitating the creation of knowledge bases (KBs) for software specifications, of which the UML superstructure specification is our initial target. Our motivation is that such specifications are dense, repetitive and difficult to use. They are written primarily in semi-structured text, but the structure must be maintained manually as they are edited, resulting in inconsistency. End users cannot use them efficiently because of the duplications, numerous concepts connected only implicitly, and general complexity of the document. Our immediate objective is to generate a KB for the UML specification by extracting knowledge from as many sources as possible in the document such as document structure, embedded natural language, as well as implicit and explicit cross references. In this paper our focus is the first step: extraction of the document's logical structure. Many key concepts of a document are expressed in this structure, which includes the headings of the chapters, sections, subsections, etc. By extracting such a structure in XML format, we can form a good infrastructure for the subsequent KB creation steps.

Keywords: *Document analysis, Logical structure,*

* This research is supported by the IBM Ottawa Software Lab

HTML or XML formats from DOC/RTF, the results also tend to have the same properties.

Unfortunately, after running the program for the different chapters and the whole document as well, it failed. We found out, there is a considerable amount of incorrect tagging. The tool opened each part, chapter, section, etc by “<Sect>” in a proper place of the document but it closed all of these tags by “</Sect>” in the wrong places. The problem was more crucial when we processed the whole document at once because of the accumulative mis-tagging. Here, a sample of this detection is presented:

<Sect number=” 7.3”>
<Sect number=”7.3.1”>
</Sect>
<Sect number=”7.3.2”>
</Sect>
Correct place for closing <Sect number=”7.3”>
<Sect number=”7.4”>
</Sect>
</Sect> **Wrong place**

Therefore, we could not extract the logical structure by this simple approach and decided to develop a new program which is more powerful and capable of detecting such a wrong tagging. In the next part, our successful practice with corresponding results is provided.

5.3. Third implementation approach: leveraging the bookmarks

levD 16 >>BDC /