Appendix A

XML – EXTENSIBLE MARKUP LANGUAGE AND PARSING TREES

-:Example of XML

Listing 3.2: XML Example

```
<?"XML version="1.0?>
<document>
</ toc>
<"chapter numbering="no>
<section>Introduction to...</section>
<chapter/>
<chapter>
<title>My 2nd chapter</title>
<section>
...Continuing the text
<section/>
<chapter/>
<document/>
```

The first line of an *XML* document (as in Listing 3.1) is an *XML* declaration specifying the version of *XML* being used. Document is the root node and the level under it consists of one toc element and two chapter elements. Here the element toc has no content and is therefore opened and closed in the same tag by finishing off with a slash before the end bracket, >. Since *XML* can be used as a document format .[the order is important [David Hall- 2005

Name Conflicts

:(This XML document carries information about a table (a piece of furniture

```
<table>
  <tr>
  <td>Apples</td>
  <td>Bananas</td>
  <tr/>
<table/>
```

If these two XML documents were added together, there would be an element name conflict because both documents contain a <table> element with different content and .definition

```
<table>
```

```
<name>African Coffee Table</name>
<width>80</width>
<length>120</length>
<table/>
```

Solving Name Conflicts using a Prefix

:This XML document carries information in a table

```
<h:table>
<h:tr>
<h:td>Apples</h:td>
<h:td>Bananas</h:td>
<h:tr/>
<h:table/>
```

:This XML document carries information about a piece of furniture

```
<f:table>
<f:name>African Coffee Table</f:name>
<f:width>80</f:width>
<f:length>120</f:length>
<f:table/>
```

Now there will be no name conflict because the two documents use a different name for their <table> element (<h:table> and <f:table>). By using a prefix, we have .created two different types of <table> elements

Solving Name Conflicts using a Prefix

:This XML document carries information in a table

```
<h:table>
<h:tr>
<h:td>Apples</h:td>
<h:td>Bananas</h:td>
```

```
<h:tr/>
<h:table/>
```

:This XML document carries information about a piece of furniture

```
<f:table>
<f:name>African Coffee Table</f:name>
<f:width>80</f:width>
<f:length>120</f:length>
<f:table/>
```

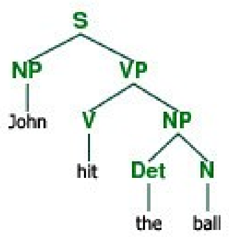
Now there will be no name conflict because the two documents use a different name for their <table> element (<h:table> and <f:table>). By using a prefix, we have .created two different types of <table> elements

Parser Tree

♦ **Basic Description**

A parse tree is made up of nodes and branches. Below is a linguistic parse tree, here representing the English sentence "John hit the ball". (Note: this is only one possible parse tree for this sentence; different kinds of linguistic parse trees exist.) The parse tree is the entire structure, starting from S and ending in each of the leaf nodes (John, :[hit, the, ball). We use the following abbreviations in the example [ref

- .S for sentence, the top-level structure in this example
- NP for noun phrase. The first (leftmost) NP, a single noun "John", serves as .the subject of the sentence. The second one is the object of the sentence
- .VP for verb phrase which serves as the predicate
- ."V for verb. In this case, it's a transitive verb "hit
- ."Det for determiner, in this instance the definite article "the
- .and N for noun



[Figure1: Parse Tree [Wikipedia-Parser

In a parse tree, each node is either a root node, a branch node, or a leaf node. In the example to the right, S is a root node, NP and VP are branch nodes, while John, hit, the, and ball are all leaf nodes [Wikipedia-Parser].