Namespaces in XML

Author: Mihai Rotaru

Date: 1 Dec 2011

Namespaces, a familiar concept to C++ and Java programmers, have a similar role in XML - to prevent name clashes. The problem of name clashes has the potential to be more acute with XML due to it's versatility and flexibility. XML has a wider and more difficult to anticipate application domain, which makes namespaces a very important component to ensure a way of mitigating potential name conflicts.

For example, a file containing information about a module taught at a university might have the following structure:

```
1 <?xml version="1.0"?>
2 <module>
3 <ID>CMT3315</ID>
4 <tutor>Ray Adams</tutor>
5 </module>
```

And another file could contain information about a modules in a software system:

```
1 <?xml version="1.0"?>
2 <module>
3 <name>Graphical User Interface</name>
4 <priority>high</priority>
5 </module>
```

These documents could be included in the same XML document, via external entities - which would lead to two elements having the same name, but at the same time being completely unrelated. The name of XML elements is described as the element's 'type' in the XML standard (http://www.w3.org/TR/REC-xml/#sec-starttags), and all elements at the same hierarchical level are likely to be assumed as belonging to the same type.

Namespaces provide a way of differentiating between elements with the same name, but which are unrelated. The document containing both the above elements would end up looking like this:

```
01 <?xml version="1.0"?>
02 <root xmlns:t="teaching" xmlns:s="software_systems">
03 <t:module>
04 <ID>CMT3315</ID>
05 <tutor>Ray Adams</tutor>
06 </t:module>
```

```
07 <s:module>
08 <name>Graphical User Interface</name>
09 <priority>high</priority>
10 </s:module>
11 </root>
```

This is a well-formed document, and there are no ambiguities since the two different types of module elements are placed in different namespaces, therefore avoiding name conflicts.