Python example

import xml.dom.minidom

Model\_Template= xml.dom.minidom.parse( “TestModel.xml”)

x = Model\_Template.getElementsByTagName("CAD\_System\_and\_Version");  
xlen = x.childNodes.length;  
y = x.firstChild;  
txt = "";  
for (i = 0; i <xlen; i++) {

// Node type 1: Element (eg. CAD\_System), type 2: Attribute (eg. Rhino). There are few other node types, but I don’t think we need them  
  if (y.nodeType == 2) {  
    txt += y.nodeName + "<br>";  
  }  
  y = y.nextSibling;  
}

Here are some links for the use of DOM in C. We could discuss these more tomorrow.

<https://docs.oracle.com/cd/B28359_01/appdev.111/b28394/adx_c_parser.htm#ADXDK19768>

<http://gdome2.cs.unibo.it/>

Here is an example of the use of a C library for manipulating XML files:

/\*

Simple test with libxml2 <http://xmlsoft.org>. It displays the name

of the root element and the names of all its children (not

descendents, just children).

On Debian, compiles with:

gcc -Wall -o read-xml2 $(xml2-config --cflags) $(xml2-config --libs) \

read-xml2.c

\*/

#include <stdio.h>

#include <string.h>

#include <libxml/parser.h>

int

main(int argc, char \*\*argv)

{

xmlDoc \*document;

xmlNode \*root, \*first\_child, \*node;

char \*filename;

if (argc < 2) {

fprintf(stderr, "Usage: %s filename.xml\n", argv[0]);

return 1;

}

filename = argv[1];

document = xmlReadFile(filename, NULL, 0);

root = xmlDocGetRootElement(document);

fprintf(stdout, "Root is <%s> (%i)\n", root->name, root->type);

first\_child = root->children;

for (node = first\_child; node; node = node->next) {

fprintf(stdout, "\t Child is <%s> (%i)\n", node->name, node->type);

}

fprintf(stdout, "...\n");

return 0;

}