Serialization

This is a school project. The objective is to train ourselves to serialize several tasks in parallel. The subject of the project can be found (in french) in the git repository.

This project is developed in two languages : C and D.

Both projects have the same conception. The only difference is the needed libraries.

All points of subject are respected.

C

The C part of the project needs the gcc compiler and libxml2 (tested on version 2.9.4) library.

To install them, install packages "gcc" and "libxml2-dev" (on debian)

```
sudo apt-get install gcc libxml2-dev
```

D

The D part of the project need ldc2 compiler.

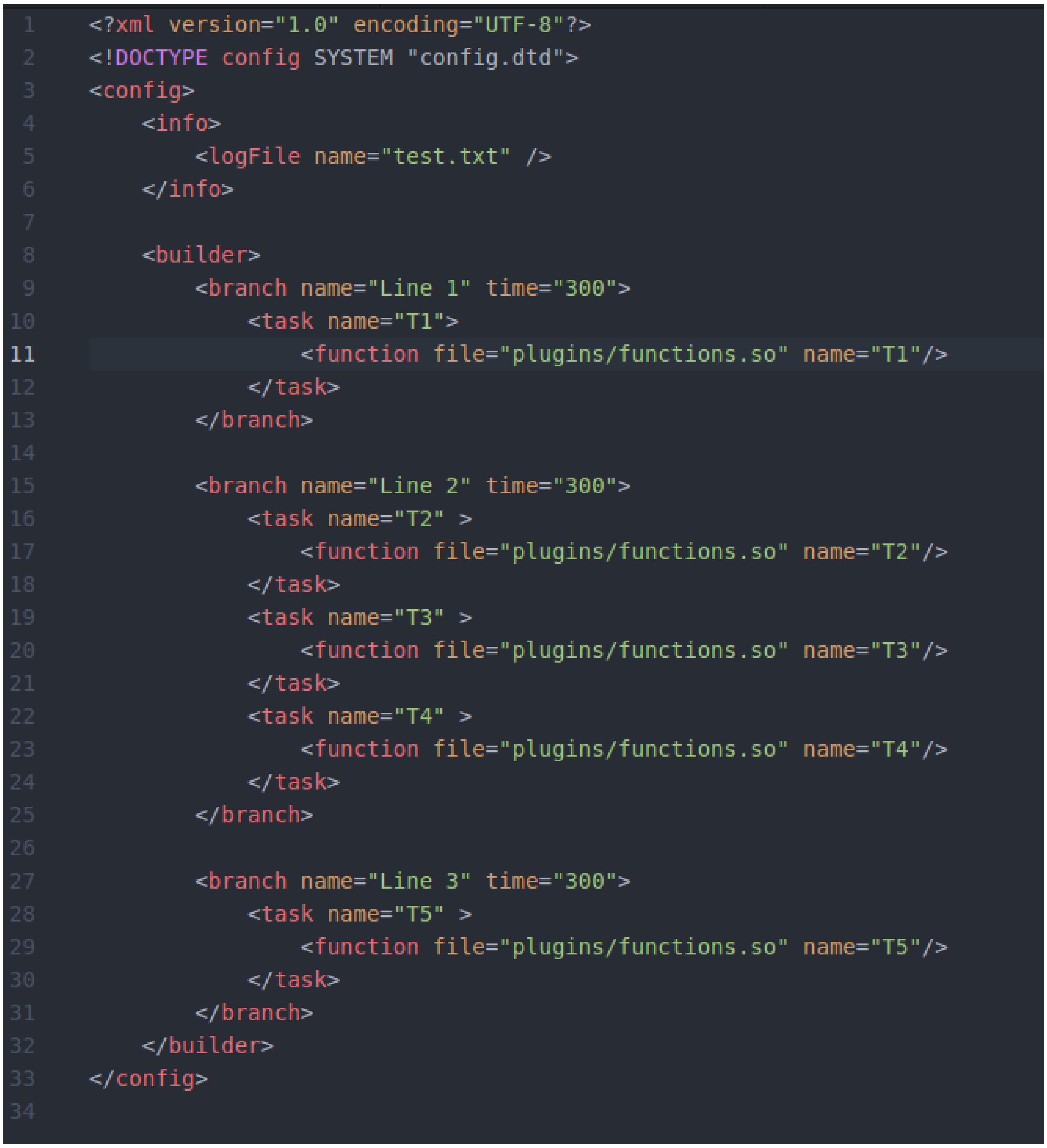
To install it, install package "ldc" (on debian)

```
sudo apt-get install ldc
```

Usage

To test the program, you need a XML File which uses these DTD rules and a compiled dynamic library with needed functions (It is possible to use existing example xml and library/ies).

This XML File has to look like this:

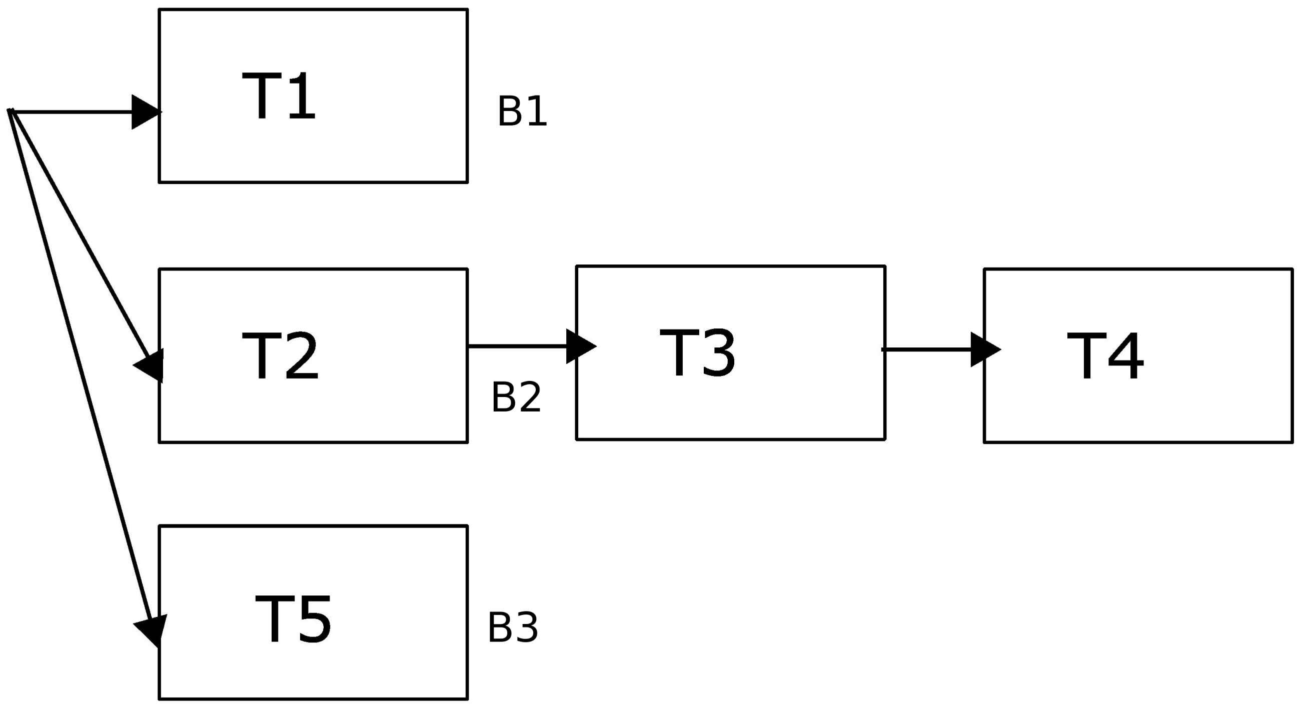


```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!DOCTYPE config SYSTEM "config.dtd">
3   <config>
4       <info>
5           <logFile name="test.txt" />
6       </info>
7
8       <builder>
9           <branch name="Line 1" time="300">
10              <task name="T1">
11                  <function file="plugins/functions.so" name="T1"/>
12              </task>
13          </branch>
14
15          <branch name="Line 2" time="300">
16              <task name="T2" >
17                  <function file="plugins/functions.so" name="T2"/>
18              </task>
19              <task name="T3" >
20                  <function file="plugins/functions.so" name="T3"/>
21              </task>
22              <task name="T4" >
23                  <function file="plugins/functions.so" name="T4"/>
24              </task>
25          </branch>
26
27          <branch name="Line 3" time="300">
28              <task name="T5" >
29                  <function file="plugins/functions.so" name="T5"/>
30              </task>
31          </branch>
32      </builder>
33  </config>
34
```

It contains the following information :

- The path to the log file (Line 5) that will be generated by the program
- The name of each branch and its maximum time (lines 9, 15 and 27)
- The name of each task placed in its branch (lines 10, 16, 19, 22 and 28)
- The name of the function called in each task and its file path (lines 11, 17, 20, 23 and 29). The file name corresponds to the dynamic library declared previously.

This XML File correponds to this next diagram (where B1, B2 and B3 are respectively Lines 1, 2 and 3)



To launch the program, go in the right folder (C or D). Compile with the command line

```
make
```

And execute with

```
./serialization path_to_xml_file
```

How it works

The program begins with the parsing to XML file given in argument.

Next, that will create the different structures/objects that the program need. That will also load the different functions of library/ies with the "dlfcn" library then launch different threads corresponding to tasks which will call these functions.

Theses threads are blocked by a condition variable.

Next, the program creates a thread by branch. Each branch will have the variable of its first task unlocked and begin a timer. When a task finishes, the branch go on to the next task if there is one or restart if it was the last task. If the timer exceeds the maximum time of the branch, the current task is stopped and the branch restarts.

At each end of a branch loop, a log message is created on log file in the form of:

```
Line 1 : T1 -> T2 -> T3 -> End 500
```

Where "Line 1" is the branch name, "T1 -> T2" is the execution file, "End" meaning that the branch finished successfully and "500" is the execution time of this loop in milliseconds.

If the branch loop was in fact aborted, the message would look like:

```
Line 1 : T1 -> T2 -> Aborted 750
```

Where the last task (here T2) is the aborted task.