Costa Rica Institute of Technology

Computer Engineering

Operating Systems Principles

Prof.: Diego Vargas

05/01/2018

Project #2:

Robotic finger

1. Motivation

With the development of this project the students will be able to integrate in a single assignment the creation of a physical device, a device driver and all the software layers required to interact with the physical device through the device driver with the creation of different software layers. With this assignment the students will understand the communication flow used at Operating Systems level to communicate with the hardware layers in a computer system.

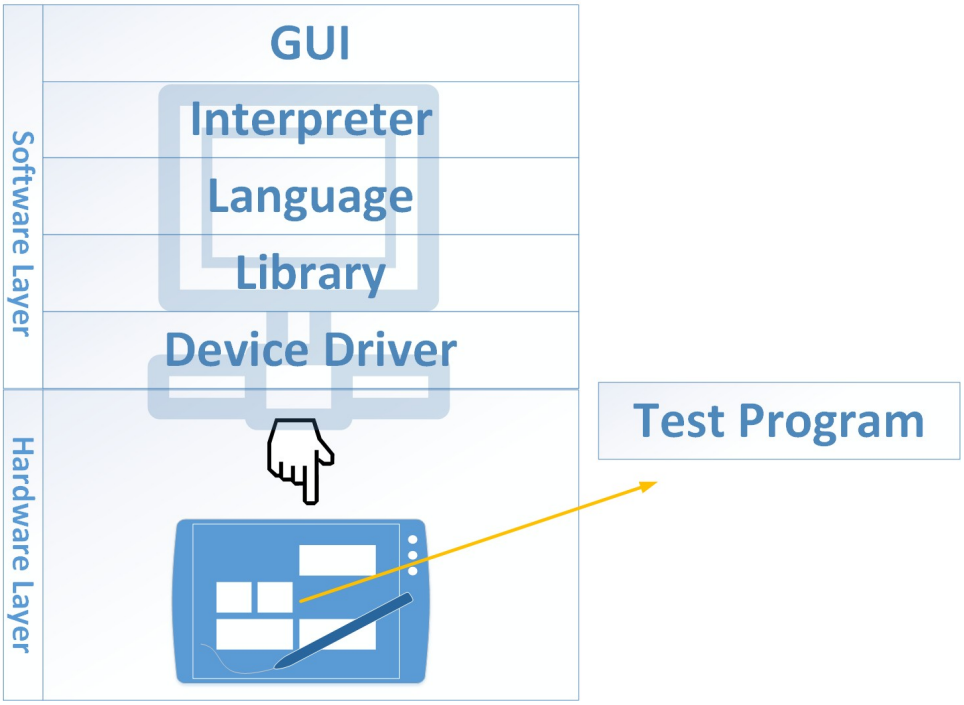
2. Objective

As part of this project the students will develop a physical device (a robotic finger) that will interact with a smart phone and/or a tablet along with the corresponding software layers: device driver, device library, the creation of a small language to interact with the robotic finger, an interpreter program to configure the robotic finger and a test program to evaluate the capabilities of the robotic finger. The main objective is to understand how the Operating Systems interact with physical devices.

3. General information

- Value of this assignment: 20%
- Program name: **roboticFinger**.
- This project can be implemented in groups of 3 people. However it will be evaluated individually, during the review session random questions will be made to any of the students.
- Any fraud will be scored with 0 and will be processed according to TEC regulations.

4. Functional requirements



A. Physical device

You will have to create a robotic finger, using any desired embedded device (ensure that you document the reasons of your selection). You will have to choose the physical interface to interact with the computer, it can be any of the following: USB, Parallel port, or communications port (COM), in this case, make sure you also justify your selection.

This robotic finger will be used to automate physical tests over a smart phone or tablet. It will be interacting with the screen of the corresponding smart phone or tablet emulating a human finger. We will have the following kind of instructions available:

- ➔ **Touch**: on this kind of instruction the robotic finger will approach the screen, touch the screen and then will reverse to its initial position.
- ➔ **Press**: on this kind of instruction, the robotic finger is going to approach the screen and is going to press it during a specified amount of time, then it will reverse to its initial position.
- ➔ **Move&Press**: on this kind of instruction, the robotic finger is going to 1) approach the screen and move the finger in any of the X and Y axes, 2) is going to press the screen, 3) it will reverse the finger to its original position, 4) it's going to get ready to repeat the process.

B. The device driver

You will develop a device driver in C programming language that will be working on any Linux Operating System. This device driver will take care of providing to the upper layers several primitives that will allow the interaction with the physical device.

The interaction with the physical device should be done using this device driver, otherwise this project will be scored with 0. Keep in mind this device driver needs to be a Linux kernel module.

C. The device library

You will have to create a library in any desired language (make sure you document the reasons of your language selection). This library will allow the implementation of the functions provided by the device driver, this means this library will be the one interacting directly with the device driver specified in the above point.

D. The language

It will be required to describe a common language that will let us describe any of the instructions specified in section **A**. This language will have to manage the following concepts (and any other concept you consider necessary):

- ➔ Instruction type (touch, press, touch&move)
- ➔ X,Y initial position
- ➔ X,Y final position

E. The interpreter

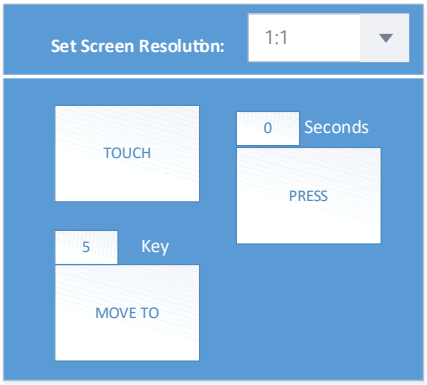
You will develop a program that will implement the small language just described and will make possible the configuration/setup of the robotic finger.

The interpreter syntax is shown as follows:

$ roboticFinger -c configuration

Important: Notice this client is going to receive the configuration parameters from a terminal session ($ means this is an unprivileged shell) using a configuration file.

F. The graphical user interface



You will develop a program that will be used to communicate with the interpreter and will provide a user interface to interact with the robotic finger. It will be sending direct instructions to the robotic finger from the user interface.

This user interface can be developed in any desired language and has to communicate with the interpreter written in C.

Next you can find sample image of the required functionality of this GUI.

G. The physical test program

This is going to be the program used to test the robotic finger functionality. It will consists of a numerical keyboard (similar to the one used by the BNCR in its Internet banking).

The software is going to generate a random PIN of 6 digits and the robotic finger should type this PIN in order to pass the physical test.

The program is going to have several screen resolutions:

- ➔ **1x1:** This is the minimal screen resolution. It will divide the screen in 1cmx1cm matrices.
- ➔ **2x2**: It will divide the screen in 2cmx2cm matrices.
- ➔ **4x4:** It will divide the screen in 4cmx4cm matrices.

Ensure there's an elegant way to choose between the different screen resolutions while preparing this program for execution.

The program interaction should be fast enough to demonstrate a fluid behavior.

In case it's not clear: this software is going to be an application used in the smart phone or tablet side.

5. Technical requirements

- This project has to be implemented using the C programming language for GNU/Linux(gcc)
  - Make sure a make file is created.
  - This covers the device driver, the language, the interpreter
- The device library can be implemented in any desired language but it has to work under GNU/ Linux as well.
- The user interface can be created in any desired language, justify your selection.

- The test program will be have to be compatible with the OS you have running in the tablet and/ or smart phone that will be used during the physical test automation.
- There are no restrictions on the language used in your embedded device to create the robotic finger circuit.

6. Documentation

Follow the instructions below in order to document this project. **You don't have to print the documentation**, it will have to be delivered in digital format (PDF).

In any of the sections, you could re-use some of the information in this document if required.

- Introduction
  - Do an overview of device drivers
    - How they work?.
    - How are they implemented?
  - Provide a brief specification of the project, what it does, how it works. You are free to re-use some of the information provided in this document.
- Development environment
  - Provide all the details of all the tools that have been used during the development of this project.
- Continuous learning attribute analysis
  - Document the continuous learning soft attribute required for the career certification program, create a section explaining how this project helped with the continues learning process, contemplating the following:
    - You should demonstrate you are capable of integrating information or knowledge to reach the learning objectives
    - You should propose solutions to problems being innovative, creative.
- Provide the details of your program design
  - Provide a description of the code of your different programs.
  - Provide the details of the design of your project using UML.
  - Provide the design details of your circuit created to perform the physical test automation.
- Instructions of how to use the program.
  - Provide detailed instructions of to setup and use the program and the robotic finger.
- Student activity log:

    - Include here the details of the activities performed by each of the students in this assignment.
    - Include one line per each activity, providing details like:
        - Description of the activity.
        - Amount of time in hours/minutes spent on each activity.
        - Total amount of hours spent in the development of this project per student.
        - **Use a time-sheet format** (or table) to present this activity log.
- Project final status
    - Provide a detailed status of your project.
    - Provide details of any issues, limitations or challenges you may have faced during the development of the project.
    - Document any known issues or bugs (if any).
- Conclusions
- Suggestions, recommendations.
- References
    - Provide all the references used during the development of this project. Use APA format (or any other formal format) and make sure all the references documented in that section are actually used in your documentation. Failure to create the references properly will be scored with negative points in the documentation review.
- Make sure your **source code** is well **documented**.
    - Explain what your functions/procedures do, all their parameters and document your expressions and data structures.
- Digital documentation:
    - Include your documentation inside the .tar.gz file that will be created with all the source code and executables of this project.
    - Upload the .tar.gz file to your shared directory in Google Drive along with the corresponding timestamp, sha1 hash and your digital signature of the document.
    - Follow the instructions in the next section with the details of the structures of the directories you must use.

7. Deliverables

- Source code and executables of your programs. The programs should be in compliance with the specifications in the **Technical requirements** section.
- Documentation (sources and pdf).
- Use the following structure inside your “Proyectos” directory in Google Drive:
    - Proyectos:
        - <carne>-proyecto2.tar.gz: compressed file with the following contents:
            - Documentation: this should be a directory with both the PDF and source files (.tex or .md) for the documentation.
            - Program: this should be a directory with your program source code and executable.
            - Additional_files: this is an optional directory, it should have any additional file you think it would be required.
        - <carne>-proyecto2.tar.gz.asc
        - <carne>-proyecto2.tar.gz.tsr
        - <carne>-proyecto2.tar.gz.sha1

8. Evaluation

- The robotic finger (physical device): 20%
    - Touch: 5%
    - Press: 5%
    - Move&Press: 10%
- The device driver: 20%
- The device library: 5%
- Robotic finger language design: 5%
- Implementation of the interpreter and GUI: 20%
- The test software implementation: 10%
- Documentation using Latex or Markdown: 20%
    - Internal documentation: 3%
    - Continuous learning attribute documentation: 5%
    - The rest of the score will be determined by the professor during the review process.

Extra points:

- Documentation (source code, internal and external documentation) in English: 5%

9. Additional considerations

The programs and documentation are in separated items in the evaluation but the following restrictions apply:

- If documentation is not delivered, you will have automatically a score of 0 in this project.
- If the source code is not compiling, you will get a score of 0. Make sure you provide a functional source code.
- **The program has to be programmed in gcc for GNU/Linux. Failure to do this will give you a score of 0 in this assignment.**
- **The interaction with the physical device should be done using the device driver. Failure to do this will give you a score of 0 in this assignment.**

Also the take the following under consideration:

- The professor will be reviewing the documentation out of the revision session.
- During the review session, the program deliverables could be downloaded from any of the students shared directories chosen by the professor.
- Each group will have up to 20 minutes to present the project during the review session and perform the technical defense of the work. The responsibility of presenting and defending all the work relies on each of the students, so it is recommended to have everything ready and handy prior the review session.
- Every error or warning displayed during the review session considered to be part of your technical validations during the development of the programs, will be punished on your final project score with -2 points.
- Each group will be responsible of the equipment that will be used during the review session. In case of issues to take your own equipment, you will have to notify the professor with 2 days in advance (prior the review session) so the required equipment can be properly coordinated.
- The following people could participate during the review session:
    - Other professors.
    - Coordinator of Computer Engineering.
    - Assistant.

10. Delivery Date

May 28th of 2018 before 23:59:59 GMT-6.

11. Revision Date

The review date of this project will be May 29th and May 31st at the end of the class. During the review you will have to download the work from your shared directory and execute your work.