

Experimental Script - Group A

1. SUMMARY

The purpose of the experiment is to analyze the transplant process (transfer) feature manual between two systems, the donor and the receiver (host), both written in C. For this process, we will compute the necessary activities, time and effort spent by a set of SPL specialists compared to our automated approach proposal. The process consists of four steps: extract, identify, adapt, and insert.

The participants will have access to a set of unit testing, a feature's entry point in the donor (a function) and an insertion point in the host system provided by the researchers. Participants may use the tools (s) that they want to try to perform the process in the shortest time possible. We request to the participants to compute all activities, step, and time spent as well as the tool of the tool used.

It is important to highlight that the experiment will not need to be performed without any breaking, since they compute time and activities correctly with each return to the execution process.

1.1. TRAINING

We have prepared some videos describing each section of this script. You can assist at any time while performing the experiment. We only ask you not to register this time in the time and effort spreadsheet.

1.1. EXPERIMENT EXECUTION ACTIVITIES

1. Download the experiment files, available at:
<https://www.dropbox.com/sh/5b781rzzor7lecp/aabt8SPQSC7G5WYGSUSKP40NA?dl=0>;
2. Access experiment form containing the access link to your time and effort registration spreadsheet:
<https://forms.gle/fqv16kfa68m6kcdy9>;
3. Install dependencies;
4. Execute the Feature transferring process;
5. Execute the unit test;
6. Perform system testing in the post-transference receiver system;
7. Send files to researchers via the experiment form;
8. Signal the end of the experiment to researchers.

2. EXPERIMENT DIRECTORY

Execution Directory: As part of the experiment preparation process, the participant should download the files contained in the folder corresponding to the group in which he was allocated. All files needed for the experiment can be downloaded by the link:

<https://www.dropbox.com/sh/5b781rzzor7lecp/aabt8spqsc7g5wygsuskp40na?dl=0>.

Effort Register Worksheet: For time and effort registration, we provide a spreadsheet for each participant. You can it by the link: <https://forms.gle/fqv16kfa68m6kcdy9>

2.1. DIRECTORIAL STRUCTURE

A GROUP

- | _ Dependencies_run -> Dependency Installation Files
- | _ Documentation -> Contains the Donor and host system documentation if the participant wants to better understand the systems.
- | _ Owner -> Contains the source code of the donor system.
- | _ Host_original -> Contains the source source code of the receiving system (host)
- | _ Host_to_transplant -> Contains the host source code to receive the feature
- | _ Test_Suite -> Temporary directory for feature and unit tests
- | _ Feature -> Temporary directory of the feature source code insertion

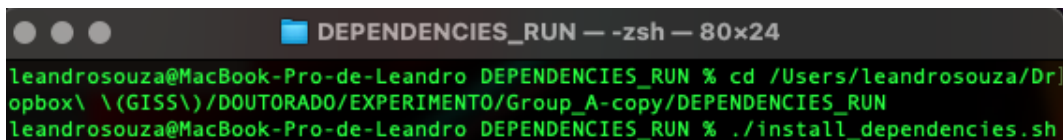
3. INSTALLATION OF DEPENDENCIES

Please install some dependencies required to execute unit tests. **Do not worry about installing these premises, at the end you can run the script that will be automatically generated after the facilities in the directory will be installed Installation of dependencies (dependencies_run)**

We have prepared a shell script to install the dependencies automatically from the directory: `Dependencies_Run` execut

3.1. INSTALLATION OF DEPENDENCIES (FOR MAC OS)

From the directory `Dependencies_Run` execut the following command: `./install_dependencies.sh`



```
leandrosouza@MacBook-Pro-de-Leandro DEPENDENCIES_RUN % cd /Users/leandrosouza/Dr[
opbox\ \ (GISS\)/DOCTORADO/EXPERIMENTO/Group_A-copy/DEPENDENCIES_RUN
leandrosouza@MacBook-Pro-de-Leandro DEPENDENCIES_RUN % ./install_dependencies.sh
```

Terminal screen with CD commands e `./install_dependencies.sh`

This command will install the following dependencies:

- [Autoconf](#)
- [libtool](#)
- [Pkg-config](#)
- [Check](#)
- [Glib](#)

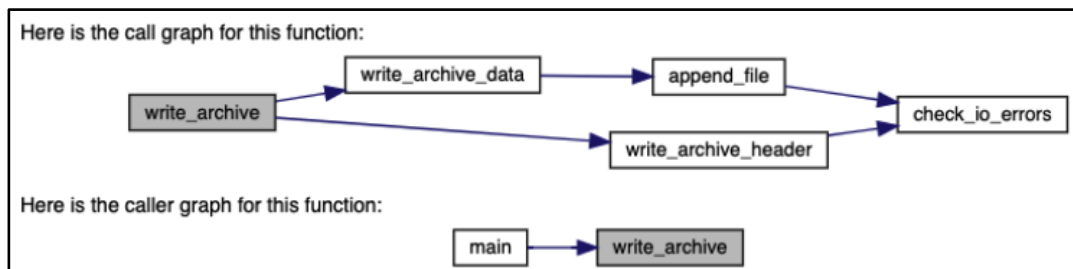
4. DESCRIPTION OF SYSTEMS AND FEATURE

We have divided the participants into two groups (A and B). This script describes the process to be performed by group A. The participant of group A have to try to transfer the feature `WRITE_ARCHIVE` from the Mytar system, a file manager in NeatVI, an editor of text. When executed, the feature `WRITE_ARCHIVE` copies all contents of a file and writes it in other with a different format. To do this, the feature must receive as input at least the name of input file and the target output file.

4.1. AVAILABLE ARTIFACTS

As they are distinct systems it will be necessary to adapt the code that implements the entry point of the feature (ie the Write_Archive function) for the insertion point in the Neatvi system, based on the following information:

- Feature inserting point: The WRITE_ARCHIVE function of the Append.C file
- Host system insertion point: point defined by notation with `__ADDGRAFTHERE__JUSTHERE` in the Ex.C file
- Call_GRAPH: The Feature call graph available.
- Test_Suite: A set of unit tests used to exercitize the feature. It can be used to assist in process of the feature adapting and executing before insertion. Such test file are available in the experiment directory.

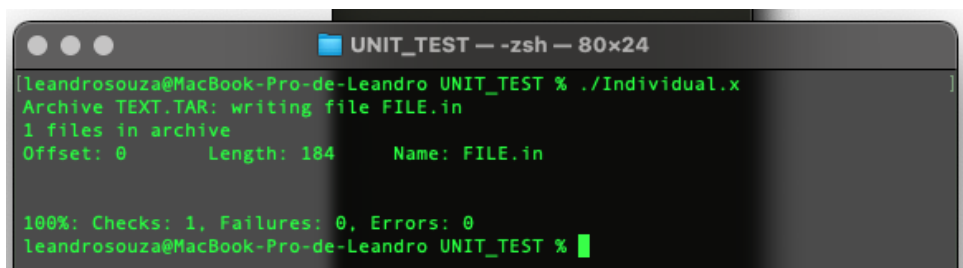


These artifacts are described more detailed in the following sections.

5. EXECUTION OF THE UNIT TEST SUITE USED TO EXERCISE THE FEATURE DURING THE ADAPTATION PROCESS

When identifying, extract the code elements and perform the necessary adaptation you can perform the unit tests available in the Test_Suite.C file. For this, you must execute the script in `run_test.sh`

If Feature is working properly, you will receive the message:



```
UNIT_TEST -- -zsh -- 80x24
[leandrosouza@MacBook-Pro-de-Leandro UNIT_TEST % ./Individual.x
Archive TEXT.TAR: writing file FILE.in
1 files in archive
Offset: 0      Length: 184      Name: FILE.in

100%: Checks: 1, Failures: 0, Errors: 0
leandrosouza@MacBook-Pro-de-Leandro UNIT_TEST %
```

Terminal with unit test departure stating: 100% percentage of acceptance; Checks: 1. Number of tests performed; Failures: 0 number of failures and errors: 0, number of errors.

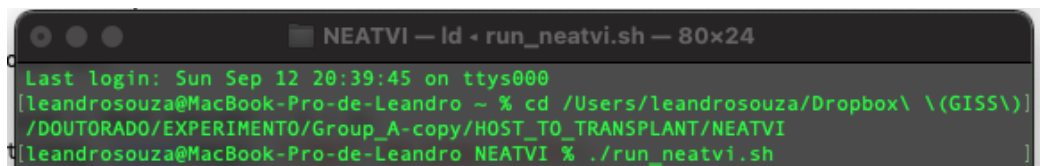
6. TRANSPLANT VALIDATION TEST

Once the feature transfed has been passing on the unit tests, you can perform the system test.

Performing the Neatvi text editor:

1. Open the terminal

2. Access the host_to_transplant/neatvi directory where you entered the code that implements the feature.
3. Run the commands:
./run_neatvi.sh inside the Neatvi Directory.

A terminal window titled "NEATVI - Id - run_neatvi.sh - 80x24". The terminal shows the following commands and output:

```
Last login: Sun Sep 12 20:39:45 on ttys000
[leandrosouza@MacBook-Pro-de-Leandro ~ % cd /Users/leandrosouza/Dropbox\ \(GISS\)
/DOUTORADO/EXPERIMENTO/Group_A-copy/HOST_TO_TRANSPLANT/NEATVI
[leandrosouza@MacBook-Pro-de-Leandro NEATVI % ./run_neatvi.sh
```

Terminal with CD commands e ./run_neatvi.sh

This will compile and execute the Neatvi system, with the name Text.txt



Tela do NEATVI

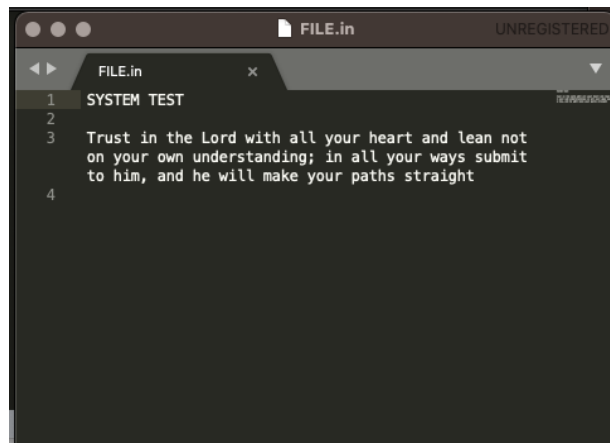
Performing the feature at Neatvi:

1. Enter *i* to enter the insertion mode
2. Enter the name of the file that will be read, *file.in* and key enter. This will copy the contents of the *file.in* file to *text.txt*
3. Key [ESC] and enter the command: *WA*. This command will run the transplanted feature and must create a *text.txt* file with the *file.in* file contente



Tela do NEATVI com os comandos para execução da feature

1. Open the text.txt file and file.in file to check if the text was copied correctly.



7. DATA SENDING

Upon completion of the execution of the experiment make sure that it computed all the time elapsed at each stage of the process. Sign up to the researchers to end the activity and time registration spreadsheet.

Rename Group A Directory Complete Your Name Upload the folder with your changes by the link: <https://www.dropbox.com/sh/tbc0srah8w601qm/aaayo0oqtjhfjkjsozeja?dl=0>

We greatly appreciate the time and availability to perform this experiment, vital for the progress of our research.