

EXPERIMENT SCRIPT FOR THE SCENARIO I (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.

2. 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.

3. EXPERIMENT EXECUTION ACTIVITIES

1. Download the experiment files, available at: [experiment directory](#)
2. Access experiment form containing the access link to your time and effort registration spreadsheet: [registration sheet](#).
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.

4. 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: [transplantation directory](#)

Effort Register Worksheet: For time and effort registration, we provide a spreadsheet for each participant. You can it by the link: [transplantation directory](#)

4.1. DIRECTORIAL STRUCTURE

GROUP A

| _ 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 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

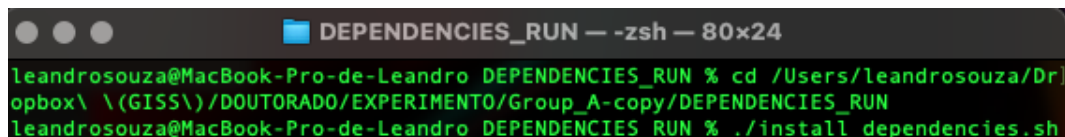
5. DEPENDENCIES INSTALATION

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_install](#)

5.1. INSTALLATION OF DEPENDENCIES (FOR MAC OS)

From the directory [Dependencies_Run](#) execute 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](#)

6. DESCRIPTION OF SYSTEMS AND FEATURE

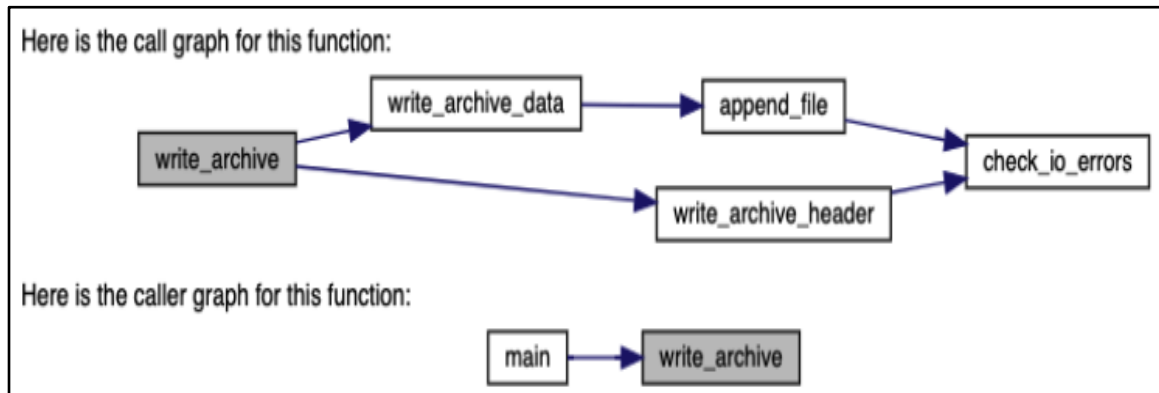
We have divided the participants into two groups (A and B). This script describes the process to be performed by group A. The participant of group A have to try to transfer the feature `WRITE_ARCHIVE` from the *Mytar* system, a file manager in *NeatVI*, an editor of text. When executed, the feature `WRITE_ARCHIVE` 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.

6.1. 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 test the feature. It can be used to assist in process of the feature adapting and executing before insertion. Such test files are available in the experiment directory.



These artifacts are described more detailed in the following sections.

7. UNIT TEST EXECUTION

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 testing informing: 100% percentage of acceptance; Checks: 1. Number of tests performed; Failures: 0 number of failures and errors: 0, number of errors.

8. TRANSPLANT VALIDATION

After transferring the feature to the host system and it is passing on the unit tests, you can perform the final transplant validation test. We have provided a post-operative test suite. The post-operative tests correspond to a set of regression, augmented regression and acceptance tests to exercise the feature transplanted and check if the transplant has not broken the host system.

Executing the post-operative tests:

1. Open the terminal.
2. Access the directory **HOST_TO_TRANSPLANT/NEATVI_1.0/TRANSPLANTATION_TEST_CASES** where you insert the feature source code.
3. Execute the commands: `./test_product_line.sh`

9. SENDING ALL ARTIIFACTS

After the execution of the experiment is completed, 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 worksheet](#).

Rename Group B Directory Complete your name and compact the folder. Then upload the folder with your changes from the [form](#).

If you have a shipping problem, download the artifacts from [transplantation artifacts](#) and report this to the researchers

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