LARtk 5

Jorge Almeida

Department of Mechanical Engineering
University of Aveiro
almeida.j@ua.pt

March 9, 2015

What is LARtk?

- Repository containing code produced in the LAR laboratory over the years
- Consists of several ROS packages in ROS catkin format.
- Follows a tree structure divided into meta-packages by function
- Uses Git (`http://git-scm.com/`). In case of access problems please contact almeida.j@ua.pt.
- Current version 5th, although the first to use Git.

Repository structure

hardware/

    ... (documentation)

src/

    applications/
    bases/
    demos/
    hmi/
    navigation/
    perception/
    planning/
    sensors/
    utils/

Description

applications	set of launch files used to launch an application
bases	hardware control
demos	simple demos
hmi	human machine interface
navigation	navigation packages
perception	detection of pedestrian, objects, road, etc.
planning	trajectory planing
sensors	sensor communication
utils	generic utilities

How to use

1. Create the `workincopies` folder inside your `home` folder

2. Clone the repository into the `workincopies` folder:
   - `git clone ssh://USER@lars.mec.ua.pt/home/repositories/lar5 lar5`
   - attention to the final `lar5`

3. Start a catkin workspace in the `lar5` folder
   - in `~/workincopies/lar5/src`, do: `catkin_init_workspace`

4. Compile the LARtk
   - in `~/workincopies/lar5`, do: `catkin_make`
   - replace in the `.bashrc` file, the source of the `setup.bash` to the new `~/workincopies/lar5/devel/setup.bash` file

5. Packages with very specific dependencies will not compile. If any of these packages is required, please read the corresponding `cmake` warnings.

Git

In order to use Git, we first need to create a working branch, follow the following instructions.

1. Update the local code.

   ```
   git pull
   ```

2. Create the working branch

   ```
   git checkout -b NEWBRANCH
   ```

3. Update the server (origin) with this new branch branch

   ```
   git push --set-upstream origin NEWBRANCH
   ```

4. Work normally on the new branch

   ```
   git add FILE
   git commit FILE -m “MESSAGE”
   git push
   ```

Git

The use of branches presents some interesting advantages:

- prevents conflicts between colleagues. Different users work on different branches preventing overlap.
- guaranties the continuous operation of the master branch.

In the future, it will be necessary to incorporate the modifications made in a branch into the master branch. In this case a merge will be required. The merge operation can only be performed by the repository manager.

Additional information on Git please consult:

- `http://git-scm.com/book/en/v2`
- `https://www.atlassian.com/git/tutorials/comparing-workflows/feature-branch-workflow`

Rules

- Uppercase letters are not allowed in the package name.
- Do not create new messages, use the already existing ones!
- Do not replicate functions, search before create!
  1. search on-line for a ROS package.
  2. if it does not exist, search in the LARtk.
  3. only create if none found.
- Follow the conventions already used by the existing packages, in both organization and nomenclature.
- New packages must be created into existing meta-packages (`applications`, `bases`, etc.).
  - use standard tools for package creation, `catkin_create_pkg`.
- The `build` and `devel` folders are part of the normal catkin work-flow. These folders are not under version control.

Rules, code related

- Put source code files inside `PACKAGE/src`.
- Put local header files (for local use only) inside `PACKAGE/src`.
- Put public header files (used by other packages) inside `PACKAGE/include/PACKAGE`.
- All source and header files must start with a mandatory copyright. You can locate the copyright inside the existing files (BSD licence).
- All packages must have a mandatory `mainpage.dox` file. This file is used for doxygen documentation purposes.
- You may incorporate code from external sources given the due attention to the original license.

Final remarks

The branch work-flow prevents code conflicts between users. As such, the use of Git is encouraged to maintain the server up-to-date (even if the source code does not compile).

Not all students will need to create new packages. When working with an already existing package you must add your name to the authors list and replace the maintainer in the `package.xml` file.