EC-Earth 3 Development Guide

Introduction

EC-Earth is an Earth System Model as much as it is a continuous scientific software development project. Moreover, EC-Earth is developed by a geographically distributed group of scientists and their institutions, which have diverse backgrounds and interests. This amplifies the need for an efficient and coordinated organisation of the software development process since the EC-Earth model and community is simply to complex for an ad-hoc management.

This document is written to aid scientists that want to contribute to the EC-Earth model development. It takes into account best practices and common solutions of the scientific software development community in general and adapts procedures where appropriate.

Please read this Guide *before* engaging in code changes! This is important to avoid double work and frustration both for yourself and other members of the EC-Earth community. One of the basic principles used here is the *Share Early, Share Often* mantra, which emphasises the collaborative aspect of software development and helps to detect failures early in the process when it is still easy to correct them.

Engaging in the Model Development Process

The reason for getting involved in the EC-Earth development is typically either the need to improve the current model code (bug fixing), or the intention to add new features. In either case, the EC-Earth Development Portal (`https://dev.ec-earth.org`) is the place to get started.

Whether a bug needs fixing or a new feature is desired, it is always good practise to start by creating a corresponding **issue** at the Development Portal. This has a number of advantages:

- **The activity is announced to the whole EC-Earth community.**
  This helps to avoid double work as others may plan or have already started similar activities.
- **Other community members are invited to contribute.**
  Early feedback in form of discussions or testing is made possible in order to help finding the best solution and detect and correct errors as soon as possible.
- **The issue at the Development Portal serves as documentation.**
  As the issue evolves, the information helps others to test the development and makes later integration into the main stream easier.
- **The activity may be included in future release planning.**
  The information gathered in the issue helps to assess the level of consensus and testing about the development.

When creating an issue, it should be reported to the appropriate tracker, which is one of *Information request*, *Bug report*, *Change request*, *Merge request*, N*ew feature request*, or *General input*. Make also sure that as much relevant information as possible is provided, such as *Category*, *Status*, *Priority*, and *Assignee*. Do not hesitate to fill in this fields as it helps to understand the issue better and note that the details can be changed even at a later stage.

The Subversion Repository and Development Branches

All stages (even very early ones) of the model development process should be version controlled. There is no other way to ensure robust, reliable, and transparent development. It is not advisable to wait until a certain development activity is “completed” before it is put under version control. Code development outside the version control system will make it hard to retrace, document, or share the work. Poorly documented changes are difficult to assess, test, and integrate at a later stage.

The EC-Earth model development is based on the Subversion (SVN) version control system and an SVN server is maintained as part of the Development Portal. The other parts of the Development Portal are integrated with SVN, which makes it easy, for example, to reference SVN revisions in issues or Wiki pages. The EC-Earth SVN repository is hosted at the URL

`https://svn.ec-earth.org`

The main areas for model development activities in the EC-Earth SVN repository are called development branches. A branch is a distinct line of development, and SVN provides support for any number of branches at a time. The EC-Earth 3 development branches reside within the SVN repository at

`https://svn.ec-earth.org/ecearth3/branches/development`

A structure is maintained inside this directory in order to make it easy to locate individual development branches: There is a yearly sorting and each branch name is prefixed by the revision number of the repository at the time when the branch was created. This yields the following naming convention:

`https://svn.ec-earth.org/ecearth3/branches/development/YYYY/rNNNN-name`

The name part of the branch is to be chosen by the developer and should be descriptive.

Working with development branches basically implies the following procedure:

1) Create development branch
2) Repeat branch development cycle
    2.1) Merge latest trunk changes
    2.2) Modify
    2.3) Test
    2.4) Commit
3) Merge and abandon development branch

The first and last steps involve the so called **trunk**, which is the main line of development. As opposed to the development branches, the trunk is supposed to represent a stable status of current model development. Such a stability criteria is not imposed on development branches. The trunk is maintained by the **Core developers** group of the Development Portal. Development branches can be created and modified by any member of the **Contributing developers** group.

Each of the above steps is explained in more detail in the following sections.

Creating a New Development Branch

A development branch is created by "branching off" the trunk, i.e. the current status of the model development. This is achieved by the following SVN command:

```
> svn cp https://svn.ec-earth.org/ecearth3/trunk \
    https://svn.ec-earth.org/ecearth3/branches/development/YYYY/rNNNN-my_branch
```

Basically, this recursively copies the trunk, creating a new development branch called `rNNN-my-branch` in the development branch area of the current year. As the above command proceeds, one will be asked for a comment, which should be something like "Creating development branch for *my changes*".

It is recommended to always copy the complete trunk when creating a new branch! There are two reasons:

- Even if the intention is to change only a small part of the EC-Earth sources, there is always a chance that the modifications extend to other files later on. A common example is to start modifying some source code, realising later that changes in the run scripts are needed as well. If one didn't start the branch with the complete trunk, this is very hard to correct at a later stage.
- Merging becomes much (much!) easier if all development branches basically look the same. This makes the administration of the repository simpler and eases the merging of changes into the trunk.

There is no need to be worried about disk space for branches in the repository as SVN does "lazy copies", i.e. only modifications will actually take up more space.

Working Cycle in a Development Branch

Once a development branch has been created, work can proceed to implement the new feature, fix a bug, etc. It is important to consider any input and provide feedback via the issue that corresponds to the development activity.

Merging Latest Trunk Changes

An important characteristic of collaborative software development projects is the fact that development never stops outside one's own area of interest. This is why a development branch must be regularly synchronised with the trunk. Failing to synchronise with the trunk for a long time results in an isolated development and difficulties to feed back the development to the trunk at a later stage.

Before merging trunk changes, make sure that no local changes are present in the local working copy of the development branch (use `svn status`). Next, let the working copy receive any existing trunk changes by issuing the command

```
> svn merge https://svn.ec-earth.org/ecearth3/trunk
```

The next step is to test whether the development branch is still functional with the changes received by the previous command. If that is the case, the trunk changes need to be committed in the development branch:

```
> svn commit -m "Merged rABC:XYZ from trunk"
```

where `ABC` and `XYZ` should be replaced by the revision numbers that the `svn merge` command above reported.

Again, it is important to keep the development branch in sync with the trunk to make sure that the modifications are compatible with the current model status and to avoid complications at a later stage.

Changing Files

...

Committing Changes

...

Commit Comments: Do's and Don'ts

- Describe primarily **what** you've changed and **why**, not how.
  (How the changes look like is recorded anyway and can be checked with `svn diff`)
- Don't assume that others know or need to know as much as you about your changes. Don't be too cryptic or too detailed.
- Do not add your name! (that is done automatically)
- Do not add the date or time! (that is done automatically, too)

Feeding Back to the EC-Earth Trunk

...

Preparations

Testing

Merge request