GOMSS Nowcast

Release

May 21, 2016

Contents

This is a collection of documentation about the Gulf of Maine and Scotian Shelf (GOMSS) Nowcast project. The focus of the project is the automated production of a daily ocean physics prediction for the GOMSS region using the NEMO version 3.6 ocean model.

CHAPTER 1

About the Project

High-resolution ocean forecasting models have been developed in DFO under the coordination of the DFO-EC-DND inter-departmental CONCEPTS program. The models are based on the Nucleus for European Modelling of the Ocean (NEMO), a state-of-the-art community ocean model framework that includes comprehensive ocean dynamics. Recent results have shown the benefits of upgrading the model codes to NEMO version 3.6. Besides the ongoing work in CONCEPTS to develop 24/7 operational forecasting capacity on EC computers, it is also desirable for DFO researchers to carry out operational tests for technical development, and the results for such tests can also provide open boundary forcing to the nearshore forecasting models under development for various DFO projects.

The objective of this project is to develop software for automatic online documenting of model codes, model configuration and analysis scripts, and routine downloading large-scale model outputs for setting up the initial and boundary conditions of the NEMO models, and atmospheric forcing fields for driving the ocean models.

This project draws inspiration and some software from the Salish Sea NEMO model, its nowcast/forecast system, and the software that powers it.

CHAPTER 2

Contents

2.1 Web Services

The GOMSS Nowcast project relies on a collection of web services to provide version control repository hosting, issue tracking, documentation rendering and hosting, and software package distribution. Together this collection of interlinked web service accounts provide a web portal for the project.

This section described how those web service accounts are set up, managed, and interconnected.

2.1.1 Software Repositories and Issue Tracking

A team account on the bitbucket.org service is used to provide centralized, publicly accessible storage for the Mercurial version control repositories that contain the project's code and documentation. Bitbucket is also used to provide an issue tracker for each repository.

The Bitbucket Cloud Teams page provides information on the concept of Bitbucket Teams.

A base account with the user name `gomss-nowcast-project` for the project was created from the Bitbucket Signup page. The credentials for that account are controlled by Doug Latornell <doug.latornell@43ravens.ca>.

Using the `gomss-nowcast-project` account, a team called `GOMSS-Nowcast` was created with the team id `gomss-nowcast`. The team's overview page is at https://bitbucket.org/gomss-nowcast/.

User Groups Management

The `gomss-nowcast-project` account defaults to being a member of both the `Administrators` and the `Developers` groups of the `gomss-nowcast` team. The *Manage Teams* > *User groups* page was used to add Doug Latornell to the `Administrators` and `Developers` groups.

Members of the `Administrators` can manage what groups other users are members of, and what type of access (Read, Write, or Admin) they have to each of the team repositories.

Members of the `Developers` group have write access to all team repositories, and can create new team repositories.

Repository Issue Trackers

When a new repository is created there is a *Issue tracker* checkbox that can be clicked to provide the repository with an Issue Tracker. See https://bitbucket.org/gomss-nowcast/docs/issues for an example.

If an issue tracker was not enabled when a repository was created it can be added later on the repository's *Settings* > *Issue tracker settings* page.

There are also *Settings > Issues* links to set up lists of Components, Milestones, and Versions that can be used to classify and organize issues.

Repository issue trackers can be used to organizer development tasks, bug reports, and enhancement ideas.

2.2 Creating Documentation

This section describes how documentation directory trees are set up. The special case of the *GOMSS Nowcast Project Documentation* is described separately from *Documentation in Code Repositories* because the former is in a repository that contains only documentation, while documentation in code repositories is only one element of their contents.

2.2.1 GOMSS Nowcast Project Documentation

Version Control Repository Creation

The GOMSS Nowcast project documentation was initialized by creating a public Mercurial version control repository in the gomss-nowcast team account on the Bitbucket web service. The repository from Bitbucket was cloned:

```
$ cd projects/gomss-nowcast/
$ hg clone ssh://hg@bitbucket.org/gomss-nowcast/docs
```

Initial Files and Directories Creation

In an activated *Working Environment* the **sphinx-quickstart** command was used to create a Sphinx configuration file, `conf.py`, a master documentation file, `index.rst` for the project docs, a `Makefile` to control the documenation build processes, and 3 empty directories:

- `_build/` where the rendered docs produced by the build process will be stored
- `_static` where static asset files such as images used in the docs will be stored
- `_templates` where custom templates for the docs pages will be stored

```
$ source activate gomss-nowcast-docs
(gomss-nowcast-docs)$ cd docs
(gomss-nowcast-docs)$ sphinx-quickstart
```

The default setting values offered by **sphinx-quickstart** were accepted with the following exceptions:

```
The project name will occur in several places in the built documentation.
> Project name: GOMSS Nowcast
> Author name(s): GOMSS Nowcast Project Contributors

Sphinx has the notion of a "version" and a "release" for the
software. Each version can have multiple releases. For example, for
Python the version is something like 2.5 or 3.0, while the release is
something like 2.5.1 or 3.0a1.  If you don't need this dual structure,
just set both to the same value.
> Project version: 1
> Project release [1]: 1

Please indicate if you want to use one of the following Sphinx extensions:
...
> intersphinx: link between Sphinx documentation of different projects (y/n) [n]: y
...
```

```
> mathjax: include math, rendered in the browser by MathJax (y/n) [n]: y
...

A Makefile and a Windows command file can be generated for you so that you
only have to run e.g. `make html' instead of invoking sphinx-build
directly.
> Create Makefile? (y/n) [y]:
> Create Windows command file? (y/n) [y]: n
```

Sphinx Build Configuration File

The `conf.py` file was edited to:

- Set configuration options that we want
- Delete the many commented out configuration settings that described defaults that we want to use
- Add come code to make the copyright year range automatically adjust to include future years

Please read the `conf.py` file and the Sphinx build configuration file documentation to learn more.

Master Documentation File

The `index.rst` file was edited to add a description of the project, and to start the documentation `Contents` section with the `CONTRIBUTORS.rst` file.

When the documentation is rendered to HTML the contents of the `index.rst` file become those of the top level `index.html` page.

License Choice

A Creative Commons Attribution 4.0 International License was chosen for the GOMSS Nowcast project documentation because it is a permissive license that requires attribution for use of the documentation. It is appropriate for the contents of the repository because they are primarily text rather than code.

The blob of HTML that provides the necessary license links and metadata was generated on the Creative Commons License Chooser page.

The copyright notice and license citation were added to the `index.rst` file as its `License` section.

Project Contributors File

The `CONTRIBUTORS.rst` file was created and text added to it to provide information about the leadership of the GOMSS Nowcast Project and a list of people who contribute to it.

Documentation Build and Preview

The command described in *Building the Documentation to Preview Changes* were used to do an initial build of the documentation which was previewed in a browser by opening the `docs/_build/html/index.html`.

Initial Version Control Commit

After correction of any typos and/or layout issue revealed by the rendered docs preview, the new documentation files were added and committed to the version control repository with:

```
(gomss-nowcast-docs)$ hg add
(gomss-nowcast-docs)$ hg commit -m"Init repo w/ README, license, CONTRIBUTORS, index pg & Sphinx bui
```

2.2.2 Documentation in Code Repositories

To be written

2.3 Documentation Maintenance

The GOMSS Nowcast project documentation can be edited on-line in the bitbucket.org interface by following the links in the upper right-hand corner of any page of the rendered pages.

This section described the set-up that you need to do if you want to edit the documentation on your own computer, and render it locally to HTML to preview your changes before you commit and push them to the public repository.

2.3.1 Software Versions

The GOMSS Nowcast project documentation is developed and maintained using Python 3.5 or later, and Sphinx 1.4.1 or later.

2.3.2 Getting the Source Files

Use Mercurial to clone the documentation repository from the `gomss-nowcast` team account on bitbucket.org with the command:

```
$ hg clone ssh://hg@bitbucket.org/gomss-nowcast/docs
```

or

```
$ hg clone https://<your_userid>@bitbucket.org/gomss-nowcast/docs
```

if you don't have ssh key authentication set up on Bitbucket.

2.3.3 Working Environment

Setting up an isolated development environment using Conda is recommended. Assuming that you have Miniconda3 or the Anaconda Python distribution installed, you can create and activate an environment called `gomss-nowcast-docs` that will have all of the Python packages necessary to build the documentation on your computer with the commands:

```
$ cd docs
$ conda create -n gomss-nowcast-docs python=3 sphinx
$ source activate gomss-nowcast-docs
```

You can edit the documentation files in any code-type editor. They are plain text files. The files are written in reStructuredText and converted to HTML using Sphinx.

To deactivate the environment use:

```
(gomss-nowcast-docs)$ source deactivate
```

2.3.4 Building the Documentation to Preview Changes

Building the documentation is driven by `docs/Makefile`. With your *Working Environment* activated, use:

```
(gomss-nowcast-docs)$ cd docs
(gomss-nowcast-docs)$ make clean html
```

to do a clean build of the documentation. The output looks something like:

```
rm -rf _build/*
sphinx-build -b html -d _build/doctrees   . _build/html
Running Sphinx v1.4.1
making output directory...
loading pickled environment... not yet created
building [mo]: targets for 0 po files that are out of date
building [html]: targets for 3 source files that are out of date
updating environment: 3 added, 0 changed, 0 removed
reading sources... [100%] index
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
writing output... [100%] index
generating indices... genindex
writing additional pages... search
copying static files... done
copying extra files... done
dumping search index in English (code: en) ... done
dumping object inventory... done
build succeeded.

Build finished. The HTML pages are in _build/html.
```

The HTML rendering of the docs ends up in `/docs/_build/html/`. You can open the `index.html` file in that directory in your browser to preview the results of the build before committing and pushing your changes to Bitbucket.

Whenever you push changes to the `docs` repository on Bitbucket the documentation is automatically re-built and rendered at http://gomss-nowcast.readthedocs.io/en/latest/.

2.3.5 Version Control Repository

The GOMSS Nowcast project documentation source files are available `docs` Mercurial repository at https://bitbucket.org/gomss-nowcast/docs.

2.3.6 Issue Tracker

Development tasks, bug reports, and enhancement ideas are recorded and managed in the issue tracker at https://bitbucket.org/gomss-nowcast/docs/issues.

2.4 GOMSS Nowcast Project Contributors

The Gulf of Maine and Scotian Shelf (GOMSS) Nowcast project is lead by Dr. Youyu Lu in the Ocean Assessment and Prediction Section, Fisheries and Oceans Canada, Bedford Institute of Oceanography.

The following people have contributed code, documentation, etc. to the GOMSS Nowcast project repositories hosted on Bitbucket:

- Youyu Lu <youyu.lu@dfo-mpo.gc.ca>
- Doug Latornell <doug.latornell@43ravens.ca>

CHAPTER 3

License