



Major funding for CSDMS comes from the National Science Foundation; EAR award number: 1226297.

CSDMS Tools to Promote Visibility of Open-source Model Code

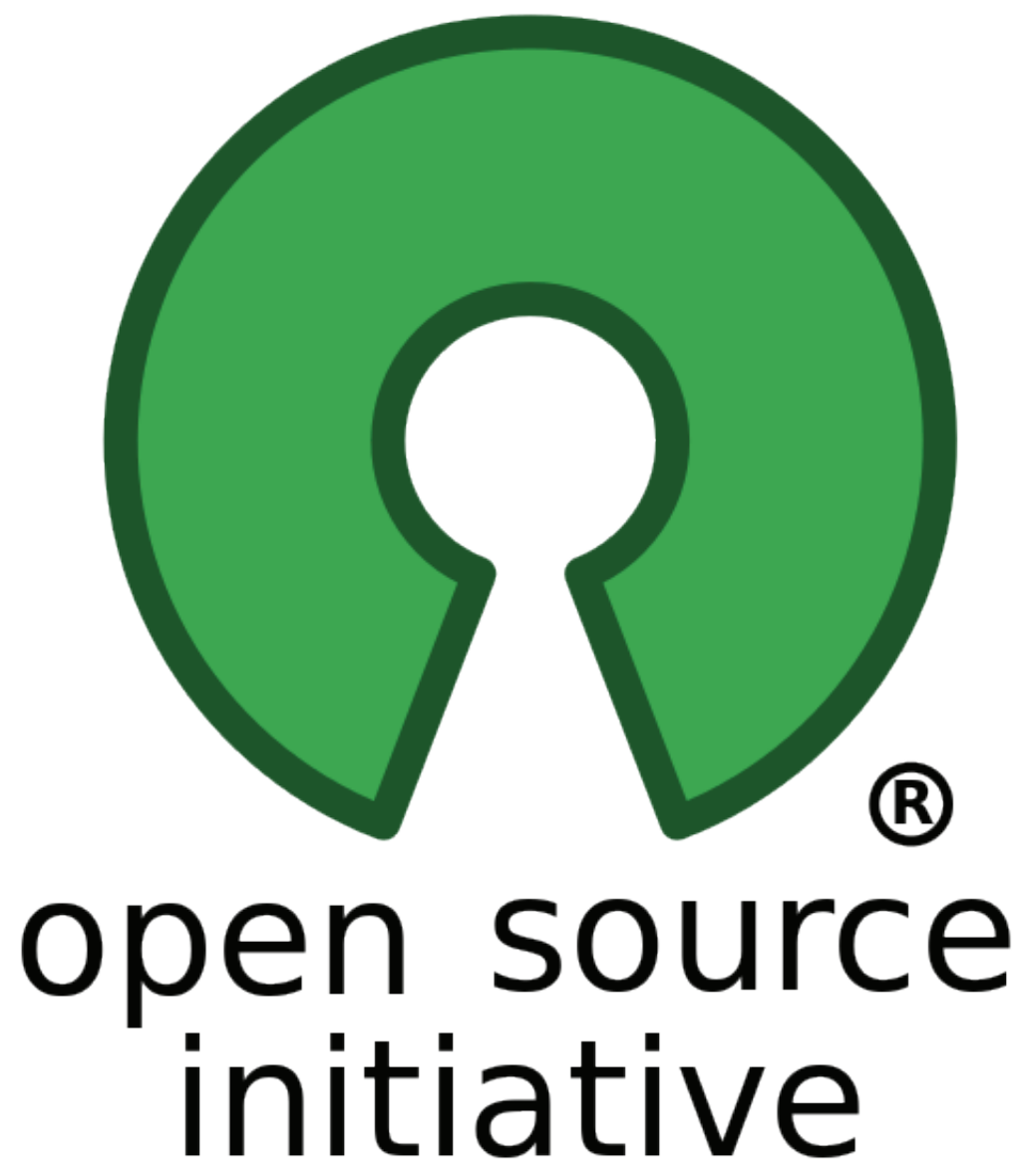


Albert J. Kettner & Irina Overeem

Community Surface Dynamics Modeling System (CSDMS), Institute of Arctic and Alpine Research (INSTAAR), University of Colorado, Boulder CO, USA
Kettner@colorado.edu

Introduction

CSDMS –the Community Surface Dynamics Modeling System- encourages and supports the effort of developers to provide their numerical models as open source codes. Open source code reduces redundancy; it guarantees reuse and it makes research more replicable (Ince et al., 2012). Increasingly, scientific journals require that codes are made available with papers describing modeling results, e.g. AGU journals have adopted this policy. At the same time CSDMS strives to ensure that model developers receive recognition for their work, even when code is submitted which is not (yet) published in a scientific journal. We ensure this by assigning a Digital Object Identifier (DOI) to source code. By adopting the DOI system, CSDMS guarantees the long-term archiving and availability of model source codes. An assigned DOI name will be guaranteed to redirect to the appropriate source code version, and the associated metadata location.



Licensing of source code

The main goal of CSDMS is to offer a community-built and freely available suite of integrated, ever-improving software models and modules that predict the movement of fluids, and the flux (production, erosion, transport, and deposition) of sediment and solutes in landscapes and their sedimentary basins over a broad range of time and space scales. It is important to choose a license for your source code. CSDMS requires that all source code that is freely available through the CSDMS model repository is licensed. Licensing of source code is important as it defines what other users are allowed or not allowed to do, and it could solve liability issues for the developers. Licensing software models and modules ensure you that once freely available made software models and modules remain fully available for the community, even if others are 'upgrading' your code. CSDMS recommends model developers to use the GPL v2 or v3 licenses, which are widely used by free libre / open source software developers because these licenses:

1. provide a better quid-pro-quo for developers,
2. establish collaboration between people,
3. protect developers work,
4. encourage increasing the amount of free software.

Using the GPL licenses will require that all the released improved versions be free software. This means you can avoid the risk of having to compete with a proprietary modified version of your own work. And even if you don't find the restrictions of the GPL license necessary, potential co-developers may, so your project is more likely to be successful if you accommodate them. More information on licensing:

- <http://rosenlaw.com/oslbook.htm> (Open Source Licensing book)
- <http://www.opensource.org/licenses/> (list including information of most open source licenses)
- <http://www.gnu.org/licenses/gpl-2.0.txt> (GLP v2 license)




Model / tool metadata

Meta data is of great importance to documenting submitted codes, as it enhances the future value of your code. With the help of a brief model description form on the CSDMS web portal, developers can provide basic model information grouped in: a) Summary, b) Contact, c) Technical specs, d) In/Output, e) Process, f) Testing, g) Additional info, and h) Component info. This information is kept in a underlying database system, and can be mined such that it displays for example lists of 'hydrological models', or 'all models that are 1D, or making cross queries like: 'coastal models' that run on a 'linux' platform using a 'single processor'. See also: http://csdms.colorado.edu/wiki/Model_download_portal

CHILD

Metadata

Summary	Contact	Technical specs	In/Output	Process	Testing	Other	Component info
Describe processes represented by the model	Basic processes include runoff generation, water erosion and sediment transport, and gravitational erosion and sediment transport. Depending on the application, the user can apply a vegetation-growth module, various tectonic functions, and other options.						
Describe key physical parameters and equations	Too many to list here -- see Tucker et al. (2001a), the CHILD Users Guide, and other documents listed in the bibliography.						
Describe length scale and resolution constraints	In principle, the model can address spatial scales ranging from gullies and small (~1km2) catchments to mountain ranges, as long as setup and parameters are chosen appropriately. Resolutions greater than about 10,000 nodes normally require significant computation time.						
Describe time scale and resolution constraints	The steady flow assumption used by most (not all) hydrology sub-models restricts time scale to periods significantly longer than a single storm. The model has been mostly used to address time scales relevant to significant topographic evolution, though in the case of rapidly changing landscapes (e.g., gully networks) this can be as short as decades.						
Describe any numerical limitations and issues	The fluvial sediment transport equations are quasi-diffusive and typically have orders of magnitude spatial variations in rate coefficient (reflecting differences in water discharge), which makes the system of equations stiff. Small time steps are typically required, which can lead to long compute times for large meshes.						

Model info	
Authors	[Collapse]
Greg Tucker	
Source code	[Collapse]
• Download	
DOI	[Collapse]
• CHILD version: 2010.07.06	
Doi: 10.1594/IEDA/100102	
QR-code	[Collapse]
	
Link to this page	
Other models by this author	[Collapse]
▪ Bedrock Fault Scarp	



Code repository

CSDMS uses GitHub as their software version control system to provide access for those who want to share their model source code regardless of whether a model is still in development, or for stable versions. GitHub keeps track of current and historical versions of source code, it allows for easy download, makes code highly visible, and permits for easy contributions, even from parties that are not directly associated with the developers core team. All one needs to contribute code to GitHub is a Git client version on your computer. As a service to developers who will not desire to maintain a version control system themselves, or are not familiar with GitHub, the CSDMS Integration Facility will maintain the GitHub repository if codes are submitted directly to them.



References

- Hutton E.W.H., Piper, M.D., Peckham, S.D., Overeem, I., Kettner, A.J., Syvitski, J.P.M., 2014. Building Sustainable Software - The CMDMS Approach. <http://arxiv.org/abs/1407.4106>
- Ince, D.C., Hatton, L., and Graham-Cumming, J., 2012. The case for open computer programs. Nature, 482, 485-488.

DOIs for models

DOI or the Digital Object Identifier system is used to identify intellectual property in the digital environment. It is used principally by publishers, and is an implementation of the Handle System for persistent identifiers. The International DOI Federation (IDF) appoints Registration Agencies who allocate DOI prefixes, register DOI Names, and provide the necessary infrastructure to allow registrants to declare and maintain metadata. CSDMS pioneered the use of DOI's for models and collaborated with the Integrated Earth Data Applications (IEDA) of Lamont-Doherty Earth Observatory at Columbia University to issue a DOI. However, with new integrated technology DOIs can be issued directly through a GitHub app at Zenodo (CERN). CSDMS is convinced that models should be cited just like articles and books. Citing models makes it possible to:

- easily reuse and replicate research as other researchers can directly locate the referenced model,
- provide credit to the model developer as citations are harvested by e.g. Web of Science,
- track usage, so that you (or funding agencies) can measure impact.



Only those models of which 1) the source code is submitted to the CSDMS repository, and 2) associated model metadata is formally described, will receive a unique DOI. A DOI will be provided for each new major release of the model, given that this is a major upgrade of the source code. This means that several DOIs could point to the same model, but each to significantly different version of associated codes. As of September 2012, CSDMS has assigned one DOI to the latest source code of each model in the CSDMS repository. The DOI for each model can be found on the metadata page of a model, which is created when filling out a model information form. Models can be seen as 'data' and therefore we endorse citations defined by 'DataCite guidelines'. These are only recommendations for data citations, but following these guidelines closely, CSDMS strongly recommends the following structure for citing a model:

ModelDeveloper (PublicationYear). ModelName, ModelVersion. Identifier

So for example:

Tucker, G., (2010). CHILD version 2010.07.06. doi: 10.1594/IEDA/100102

Promoting community models / tools

CSDMS is all about promoting developed models and tools. Models are highlighted on the front page of the CSDMS website, and highlights are renewed every 1-2 months. Newly received models are now advertised in our CSDMS tweets, directly after they are integrated in the CSDMS repository. This allows @CSDMS followers to keep tab of new submission efficiently. The CSDMS lab repository is ever growing, including more models that are available in the model repository. These labs are meant for university educators and include a brief explanation of a model, how to operate a model as well as some exercises that can be used for a class. With a modest additional effort from a model developer the CSDMS IF is more than happy to integrate a model in the Web Modeling Tool.

Software bootcamps

CSDMS hosted a 'Software Carpentry' bootcamp, directly after the Annual meeting of 2014. Computing is now an integral part of every aspect of science, but most scientists are never taught how to build, use, validate, and share software well. As a result, many spend hours or days maintaining codes less efficient than possible. Software Carpentry is a volunteer organization whose goal is to make scientists more productive, and their work more reliable, by teaching them basic programming skills. The bootcamp was an on-site, daylong workshop that covered core programming skills needed to be a productive data analyzer or model user/developer in a small research team, these included:

- the Unix shell (and how to automate repetitive tasks),
- Python (and how to grow a program in a modular, testable way),
- Git and GitHub (and how to track and share work efficiently).

During the bootcamp, short tutorials alternate with hands-on practical exercises; learners had to work on their own laptops using either native software or a virtual machine, so that they had an active working environment upon completion of the bootcamp.

Figure x. CSDMS Software bootcamp 2014

