Environmental Modelling & Software 61 (2014) 59–64

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

A Computational Model Library for publishing model documentation and code

Nathan D. Rollins, C. Michael Barton, Sean Bergin, Marco A. Janssen[*], Allen Lee

School of Human Evolution and Social Change, Arizona State University, PO Box 872402, Tempe, AZ 85287-2402, USA

A R T I C L E   I N F O

Article history:
Received 23 February 2014
Received in revised form
29 June 2014
Accepted 30 June 2014
Available online

Keywords:
Model archive
Open source
Computational modeling
Documentation
Agent-based modeling

A B S T R A C T

We present a repository for disseminating the computational models associated with publications in the social and life sciences. The number of research projects using computational models has been steadily increasing but the resulting publications often lack model code and documentation which hinders replication, verification of results and accumulation of knowledge. We have developed an open repository, the CoMSES Net Computational Model Library, to address this problem. Submissions to the library can be original models accompanying publications or replications of previous studies. Researchers can request that their models undergo a certification process that verifies that the model code successfully compiles and runs and that it follows documentation best practices. Models that pass the certification process are assigned persistent URLs and identifiers. We present the basic components of our repository, discuss our initial experiences with the library, and elaborate on future steps in the development of this cyberinfrastructure.

1. Introduction

Interest in computational modeling of social and biological systems—i.e., algorithmic models (Miller and Page, 2007)—as a tool to understand the complex dynamics of social-ecological systems is growing rapidly with an increasing number of research projects generating models of diverse phenomena (e.g. Alessa et al., 2006; Edmonds and Meyer, 2013; Helbing, 2012). The *Network for Computational Modeling in the Social and Ecological Sciences* (CoMSES Net) Computational Model Library (CML) <http://www.openabm.org> is an open-access public archive, established with support of the US National Science Foundation, where the code and documentation of computational models can be published by researchers. CoMSES Net is a scientific community of practice for sharing knowledge and establishing standards for computational modeling in research (Janssen et al., 2008). Recent commentaries in leading scientific venues make a strong case for including computing within the broader scientific tradition of research transparency (Barnes, 2010; Joppa et al., 2013a,b; Morin et al., 2012; Peng, 2011). This is especially important as computing—and especially modeling—is moving from being an analytical tool to a fundamental aspect of scientific research. To ensure that the scientific community can evaluate and build on the rapid growth of model-based research, it is imperative that adequate descriptions of a model and the model source code itself be publicly available.

A key criterion for transparency in science is the potential for others to replicate a set of research procedures. In order to replicate model-based research, it is necessary to have a sufficiently detailed description of a computational model, its operation, and objectives; the model code in human-readable and compilable (i.e., *source*) format; and an accessible runtime environment (or sufficient information to recreate such an environment) in which to reproduce model-based research procedures.

Ideally a description of the computational model should accompany any publication about model-based research or be referenced in the publication. However, model descriptions vary greatly in the level of detail provided, and there is considerable inconsistency in which details are even included in model descriptions. CoMSES Net and the CML strongly encourage researchers to use standardized formats, like the one developed by Grimm and colleagues (Grimm et al., 2010, 2006; Polhill et al., 2008), to ensure that model descriptions are adequate for potential replication.

Even with detailed, standardized descriptions of models, however, the program code itself is needed to exactly replicate particular operations, especially when novel and unexpected outcomes

[*] Corresponding author. Tel.: +1 480 727 7067; fax: +1 480 965 7671.
E-mail addresses: Nathan.Rollins@asu.edu (N.D. Rollins), Michael.Barton@asu.edu (C.M. Barton), Sean.Bergin@asu.edu (S. Bergin), Marco.Janssen@asu.edu (M.A. Janssen), Allen.Lee@asu.edu (A. Lee).

http://dx.doi.org/10.1016/j.envsoft.2014.06.022
1364-8152/© 2014 Elsevier Ltd. All rights reserved.

are claimed. Currently, it is often very difficult to obtain model code, even with helpful model developers. This has been shown in an increasing number of replication studies that were based on the description of the model in the original article (e.g. Hales et al., 2003; Polhill et al., 2005; Wilensky and Rand, 2007; Meadows and Cliff, 2012) or even an available version of the source code (Janssen, 2007; 2009). Most replication studies are able to verify the original results but often uncover contestable assumptions and implementations which challenge the robustness of the conclusions of the original paper (e.g. Hales et al., 2003; Meadows and Cliff, 2012). Access to the original source code also makes it possible to operationalize a model's assumptions, goals, and procedures with different algorithms, computer languages, or programming platforms. These can lead to cumulative improvements in model performance or help explain why different results are obtained in different computing environments.

Computational models are microcosms of dynamic, complex systems which makes it difficult to understand some aspects of models without actually running them. This requires information about the requirements needed to run a model and access to the runtime support infrastructure (i.e., dependent software libraries and/or interpretive environments). The combination of readable code, metadata, and runtime environment are critical for permitting researchers to build on each other's work so that scientific computation can evolve and improve. However, the dissemination of model description and code does not fit neatly into established publication formats.

Descriptions of computational models that are sufficiently detailed for replication can exceed the length of an associated article in word-limited journals, and a printed version of computational model source code can be even longer, making it impractical to publish this vital information in such a manner. Even if provided as the increasingly common online supplemental materials to articles, model source code is useless in most common typeset publishing formats. If not available as a compilable text file, it must be laboriously retyped or reformatted with the potential for mistakes that could alter the original code in undesirable ways. Moreover, most journals still do not permit online submission of usable files that would permit dissemination of the program code, and virtually none provides environments in which scientific code can be evaluated—either during review or after publication. To remedy this situation, Peng (2011) recommends the creation of code repositories, maintained by scientific communities of practice "...to provide a single place to which people in all fields could turn to make their work reproducible." (Peng, 2011, p. 1227). Existing generic source code and version management repositories (i.e., for any program code and not just computational models) include BitBucket (https://bitbucket.org/), GitHub (https://github.com/), and Google Code (https://code.google.com/) but these require the technical expertise to set up a site for each model published, and are not designed with the goal of long-term digital preservation. There are platform-specific repositories for specific modeling software platforms (as often are available for other software like MatLab) such as the NetLogo Modeling Commons (http://modelingcommons.org/) and the Repast Symphony archive (http://sourceforge.net/p/repast/repast-simphony-models). These repositories generally are created to support and promote particular software packages and their user communities. They also are not aimed at long-term preservation and dissemination of scientific modeling code. Finally, there are a number of domain specific efforts to create platforms to share models such as ECOBAS for ecological modeling (Strube et al., 2008), CESM for atmospheric dynamics (Gent et al., 2011) and CSDMS for surface dynamics (Peckham et al., 2012).

As with other aspects of scientific practice, there must be incentives for researchers to dedicate the time and effort required to write detailed model descriptions, providing source code and associated metadata, and ensuring the accessibility of the necessary runtime environment. These may be more effective if they can be embedded within the system of incentives already established in the academic and research world: public recognition by peers through citation and recognition by employers and scholarly organizations as evidence of valuable research activity. Related incentives include requirements by funding agencies and journals to sufficiently document and disseminate model-based research (Morin et al., 2012; Peng, 2011).

2. CoMSES Net Computational Model Library

The CoMSES Net Computational Model Library provides a framework to address many of these and related issues relevant to encouraging greater transparency in scientific computing. The CML provides a structured but easily accessible venue for the dissemination of model code, metadata, and associated descriptive documentation in the form of a digital library rather than a source code repository and versioning environment like GitHub. Models are represented as searchable entries in the digital library, much like books are represented in online catalogs, with a title, author(s), and brief description. Model library entries can also display visual previews of a model as an image or short animation. A formatted citation is displayed for each entry so that anyone who uses the model can credit its creator(s) in a familiar way for academia and science. A user can download a model from an entry, similar to downloading a digital manuscript from a library entry. An example of a model entry is shown in Fig. 1; the original entry can be found (and model downloaded) at <http://www.openabm.org/model/3580>. See Supplemental Information for more details about model library entries.

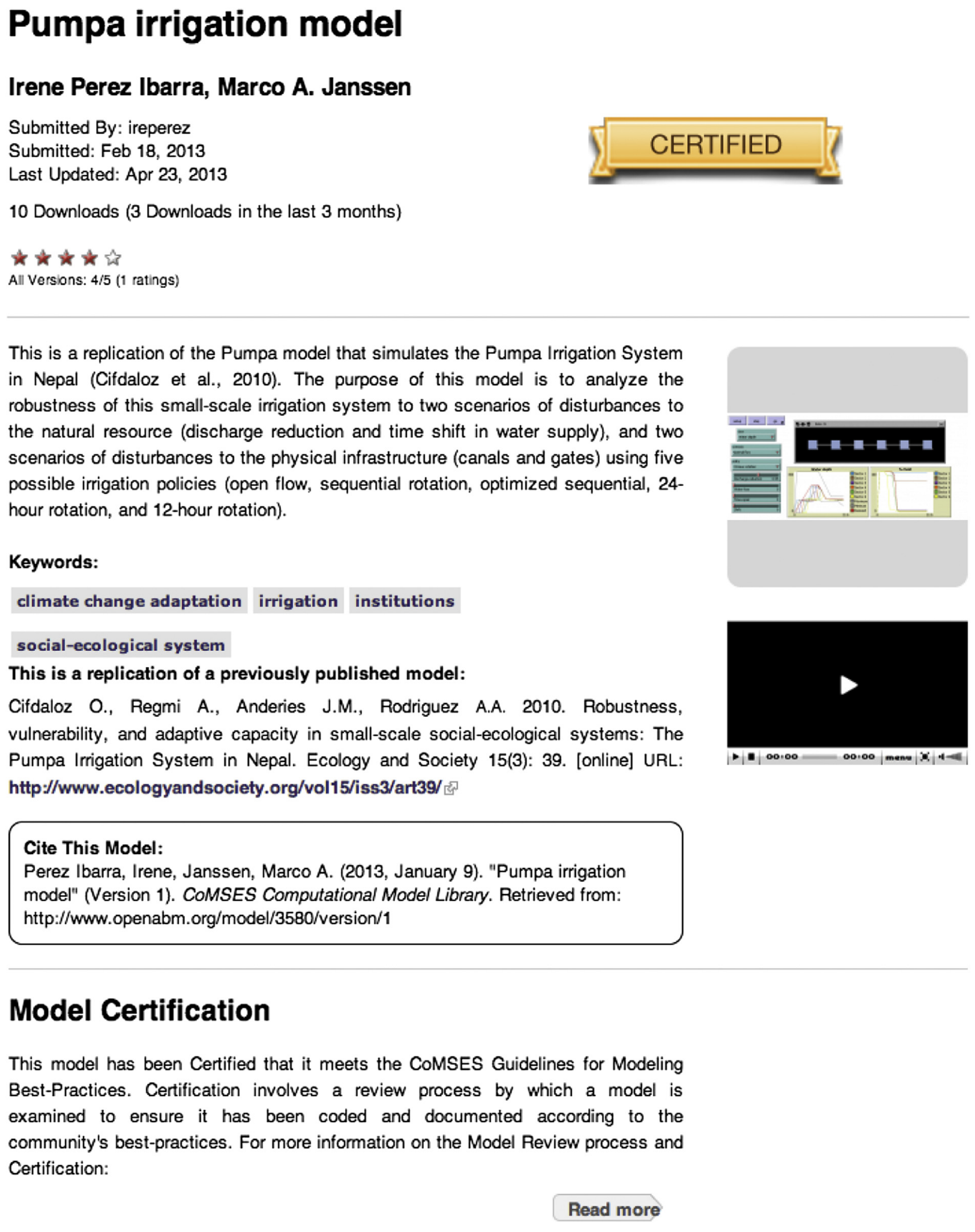
As described in more detail below, peer-reviewed models are assigned permanent digital record identifiers (equivalent to commercial DOI's). Models can be peer-reviewed as part of a journal article review or independently of an article review (see details below). Below, we describe the model library in more detail, and discuss the benefits of publishing models.

3. Details of the Computational Model Library

3.1. Scope of the CML

A primary aim of the CML is the preservation and dissemination of scientific code for computational models applied in the life and social sciences. By "computational models" we refer to models based on algorithms, such as agent-based models, cellular automata, network models, and discrete event simulation rather than equation-based models. Currently, most models in the library are agent-based models .

Unlike archives and community sites for particular software platforms, models in the CML are not restricted to any specific software platforms or programming languages, in order to make the library open to submissions from a wide scientific audience. Language and platform preferences change over time, of course, and a once-popular language may end up in a niche or abandoned. However, because the human-readable source code and documentation is published in the library, the model itself can remain useful for other scholars even if the platform it was originally designed for falls into disuse. Hence, we encourage the publication in the CML of models written in older, obscure languages so that they have the potential to be replicated in newer, more common languages and frameworks. A consequence of this approach is that although a model contributor must specify the programming platform and version used for the implementation, models in the CML cannot be



Pumpa irrigation model

Irene Perez Ibarra, Marco A. Janssen

Submitted By: ireperez
Submitted: Feb 18, 2013
Last Updated: Apr 23, 2013

10 Downloads (3 Downloads in the last 3 months)

★★★★☆
All Versions: 4/5 (1 ratings)

This is a replication of the Pumpa model that simulates the Pumpa Irrigation System in Nepal (Cifdaloz et al., 2010). The purpose of this model is to analyze the robustness of this small-scale irrigation system to two scenarios of disturbances to the natural resource (discharge reduction and time shift in water supply), and two scenarios of disturbances to the physical infrastructure (canals and gates) using five possible irrigation policies (open flow, sequential rotation, optimized sequential, 24-hour rotation, and 12-hour rotation).

Keywords:
climate change adaptation | irrigation | institutions | social-ecological system

This is a replication of a previously published model:
Cifdaloz O., Regmi A., Anderies J.M., Rodriguez A.A. 2010. Robustness, vulnerability, and adaptive capacity in small-scale social-ecological systems: The Pumpa Irrigation System in Nepal. Ecology and Society 15(3): 39. [online] URL: http://www.ecologyandsociety.org/vol15/iss3/art39/

Cite This Model:
Perez Ibarra, Irene, Janssen, Marco A. (2013, January 9). "Pumpa irrigation model" (Version 1). *CoMSES Computational Model Library*. Retrieved from: http://www.openabm.org/model/3580/version/1

Model Certification

This model has been Certified that it meets the CoMSES Guidelines for Modeling Best-Practices. Certification involves a review process by which a model is examined to ensure it has been coded and documented according to the community's best-practices. For more information on the Model Review process and Certification:

Fig. 1. Example entry in the Computational Model Library.

run online like models in the Netlogo Modeling Commons (http://modelingcommons.org). Scientists wishing to run a model in the CML must install the relevant modeling platform on their own computer. To help facilitate access to runtime environments in which to operate older models, CoMSES Net has begun a program to also archive older versions of publicly available model platforms.

3.2. Current statistics for the CML

The CML is a dynamic resource, of course, and these statistics provide a snapshot at the time of writing. Currently, 212 models are published in the library and the number of models submitted has increased each year since the library was opened, five years ago. An additional 79 models in the CML are currently unpublished, most awaiting peer-review. A majority of the models archived at the CML are written in NetLogo (167 models), with the second most common platforms being Repast (23 models) and Java (19 models). However, 21 other platforms represented among the published models, including Cormas, C++, and Python. More importantly, the CML does not prohibit the archival of models written in novel formats and provides an interface that allows for any type of model to be archived.

3.3. Publishing models and metadata in the CML

Because a primary goal of the CML is to preserve and disseminate scientific code, the base requirements for publishing a model are minimal. In order to upload model source code, the CML requires an author to:

- enter the name of the person publishing the model,
- enter a title for the model,
- enter a brief description of the model,
- upload a file containing the model code,
- upload a file containing the model documentation,
- select the desired open source license, and
- indicate the software environment needed to run the code.

Drop-down menus offer a wide selection of run-time environments and software licenses an author can choose from. If a runtime environment is not listed, the author can enter it manually. If a preferred open-source license is not available in the drop-down menus, it can be added by submitting a request to the CoMSES Net team through the website contact page. Because the CML is an open-access repository of readable source code, only open-source licenses can be applied to the code, but we strongly recommend that all published models be licensed.

While a simple text file may serve as model documentation, we recommend that authors use a standardized way to describe their model. There is no universally agreed approach is doing this (see Müller et al., 2014), but we recommend the ODD protocol (Grimm et al., 2010, 2006; Polhill et al., 2008) to document their agent-based models and other computational models. Use of a common documentation protocol enhances the readability of the documentation and ensures that all relevant details about a model implementation are explicitly available and described. In our experience the ODD protocol meets this aim well, but other forms of documentation are also acceptable.

Optionally, the name of the model author(s) (sometimes different from the individual submitting a model for publication), example datasets for validation, screenshots and videos, keywords, and additional metadata may be published along with model code. Authors can also indicate that a model is associated with a published journal article (see peer-review below). If the model is a replication of another model (e.g., in a different modeling platform, or with changes to the original code), authors are asked to provide a reference for the original model.

When a new model is submitted to the CML it is initially assigned a status of *unpublished* and only visible to the model author and any individuals with whom the author chooses to directly share the model URL; the model and its documentation cannot be seen by visitors to the library or indexed by internet search engines. Keeping a model unpublished in the library may be useful for authors who are submitting a paper to a journal and want to make the model code and documentation available to reviewers but not to the general public.

When an author chooses to publish his/her model, it becomes visible in the CML search results and accessible to all visitors to the library; it is also exposed to public search services like Google Search and Google Scholar. All models published in the CML receive properly formatted bibliographic citations, displayed prominently with the information for each model, that can be used in publications and a researcher's CV. Peer-reviewed models (see below) also receive persistent resource identifiers from *handle.net* (equivalent to commercial DOI's). A step-by-step tutorial for publishing a model in the CML can be found at <http://www.openabm.org/page/model-library-tutorial>.

3.4. Quality control through peer-review

Peer review is a widely accepted means of quality control in science, and its application to scientific computing is being actively debated (Peng, 2011; Joppa et al., 2013a, 2013b; Sliz and Morin, 2013). CoMSES Net and the CML encourage peer-review of computational models as a way to ensure the publication of high-quality, useful scientific software, although publication of non-reviewed code is permitted to meet the objectives of code preservation and dissemination. Peer-reviewed models are identified in the library. The CML provides two mechanisms for model peer-review: review as part of a manuscript review process for a journal, and specific review of model code by members of the CoMSES Net community. A model may be reviewed in either or both ways.

When a manuscript describing model-based science is submitted to a journal for peer-review, the author(s) can also submit the associated model(s) to the CML, but leave them unpublished and provide the model(s) URL to the journal editor. In that way, reviewers of the manuscript can also download and review the model code. When the manuscript is published, the model too can be published and the associated paper cited in the model library entry. CoMSES Net is partnering with journals to encourage authors of model-based research to publish their code to the CML, providing a bridge between traditional publication and code dissemination. At the time of writing, one journal (*Ecology & Society*; http://www.ecologyandsociety.org) requires submitting authors to archive agent-based models to the CML, and two other journals recommend that authors archive computational models of papers to the CML or similar archives: *Journal of Artificial Societies and Social Simulation* (http://jasss.soc.surrey.ac.uk/JASSS.html) and *Advances in Complex Systems* (http://www.worldscientific.com/worldscinet/acs).

In addition to facilitating peer review of models accompanying journal papers, the author of a model submitted to or published in the CML can request that it be reviewed independently, without being associated with a published manuscript. Currently, model reviewers are selected from among the approximately 1200 scientists who are members of the CoMSES Network. Potential reviewers are identified on the basis of their expertise in different modeling platforms and their experience in creating (and publishing) computational models. This new peer-review protocol was initiated during 2013 and members of the CoMSES Net management team are serving as review editors. However, we plan for the CoMSES Net Executive Board (elected by the network membership last year) to select a model review editor from among the larger membership on a rotating basis as is commonly done for academic journals.

During this model-centric review, the model documentation and program code is reviewed for consistency and clarity. Peer reviewers assess whether the model code is well-written, clean, and well-commented. The documentation is reviewed for readability and completeness. It must be possible to run the model with the instructions provided by the author. Model reviewers may request revisions to a model or its documentation. Models that pass review are labeled as "certified" in their library entry. Thus a certified model passes guidelines on model documentation, and it does not refer to correctness of model implementation. Although the model-centric review was only recently made available, six models have already been reviewed and certified through this process. A simplified workflow of the model publishing and peer-review process is shown in Fig. 2.

Peer-review identifies models in the library that meet higher levels of verification and metadata standards (e.g., Grimm et al., 2006; Parker et al., 2008; Polhill et al., 2008). Related to this and following protocols pioneered by PLOS and other online journals, the CML has a community-oriented comment and rating system that allows the broader research community to contribute to the recognition of strong models.

As mentioned above, peer-reviewed models are assigned a persistent internet resource identifier—a *handle*—through the *Handle System*, a component of the Corporation for National Research Initiatives Digital Object Architecture (http://handle.net). The Handle System is an international umbrella organization that assigns such identifiers to digital objects. A *DOI* is a commercial implementation of the Handle System that is widely used for journal articles. Model handles are "minted" under an agreement with the Arizona State University Libraries, which has permits to assign these resource identifiers. Regardless of future changes in server architecture or website design, access to the models in the CML will be maintained and protected. Handles indicate that a model has passed peer-review and is recognized as adhering to best practices for model documentation.

3.5. Benefits to model authors

As social scientists, we recognize the importance of leveraging multiple benefits/rewards systems to encourage collaboration and collective action to develop a public good like the CML. This is the reason why the CML represents computational models as citable examples of (peer-reviewed) scientific research that can be listed in bibliographies, CVs, and annual evaluation reports. Explicit association with peer-reviewed journal articles, certification, and the minting of handles for such models provides further public recognition that published models are scholarship worthy of long-term preservation, enhancing the reputation of the author. Certified models are regularly featured on the main home page of the CoMSES Network, and in the *CoMSES Digest* that is distributed to all members of the network. Published models also are submitted to the Google Scholar system, providing an additional level of visibility for an author's model-based research. The area in each library entry for commentary and rating by the modeling community likewise rewards the submission of higher quality models. Finally, each model entry displays the number of times a model has been downloaded since submission and during the past three months further indicating its value to the broader scientific community.

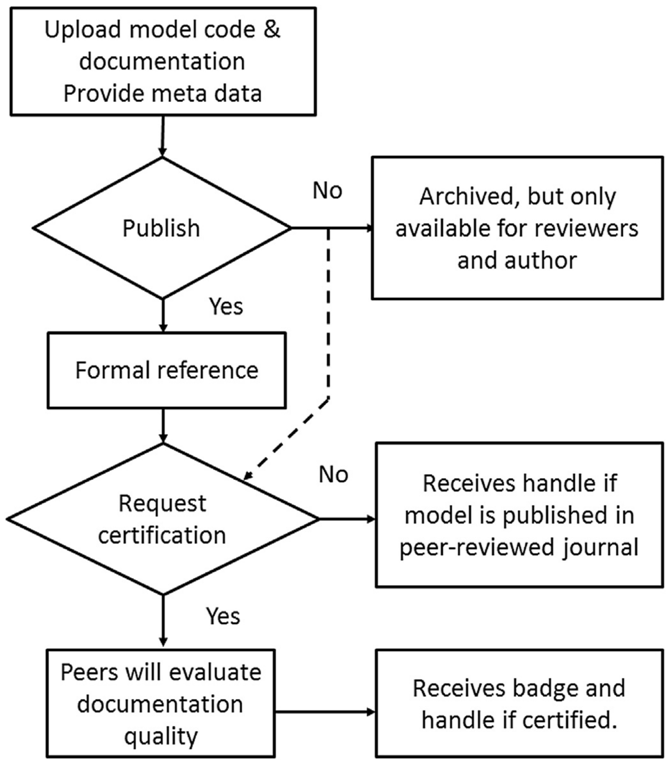

Upload model code & documentation Provide meta data
Publish
No
Archived, but only available for reviewers and author
Yes
Formal reference
Request certification
No
Receives handle if model is published in peer-reviewed journal
Yes
Peers will evaluate documentation quality
Receives badge and handle if certified.

Fig. 2. Simplified workflow for publishing a peer-reviewed computational model.

3.6. Technical implementation

The CML is implemented as an open source for the Drupal 7 web content management system (CMS) <https://drupal.org/>. The CML code is publicly available through CoMSES' Github account at <https://github.com/orgs/openabm-comses/>. Drupal is an open source software platform with a large and active support community <https://drupal.org/community>. The CML model files and metadata are stored on a networked server filesystem with automated offsite backups and disaster recovery mechanisms in place. Future development plans include automated integration with the Arizona State University Library's Digital Repository (https://repository.asu.edu) so that reviewed and certified computational models will be exported and archived redundantly in the digital repository in a durable format (e.g., XML).

4. Discussion

Science thrives because the culture of disseminating methods and results of research leads to the rapid accumulation of knowledge. The same principles apply to scientific computation, which is rapidly growing (Hayes, 2004; Wing, 2008). The community of scholars that develop and use computational models in the life and social sciences can jointly benefit from adopting a common practice of publishing models. If academic history is a guide, self-organized communities of scientists will be most successful at providing solutions to transparency in scientific computing—as they have done for the dissemination of other research.

We recognize that simply creating online code archives like the CML does not by itself promote the open sharing of modeling code and associated knowledge. Researchers may be reluctant to invest the effort to document their model code and share it openly for a variety of reason (Barnes, 2010), and journals may be equally hesitant to require submitted manuscripts to include well documented model code so that the results can be verified and replicated. However, such reluctance is already disappearing among some scientific communities. For example, within experimental economics only manuscripts with a detailed experimental protocol (often computational in nature) are accepted for review (see for example the author instructions of the journal *Experimental Economics*; Springer Publishers). Funding agencies are also increasingly requiring transparency of their sponsored research. The US National Science Foundation considers datasets, software, patents, and copyrights to be research products in addition to peer-reviewed publications and requires these products to be citable and accessible. All NSF sponsored projects since 2011 must have a data management plan. This requirement does not yet hold for computer source code nor is there any cyberinfrastructure capable of facilitating this. The CML is helping to fill this gap.

It is clear that positive incentives are needed to encourage scientists to share their code (Janssen et al., 2008). Particularly important are mechanisms to ensure that creators receive credit for scientific computing as they do in other scientific practices. One reason the CoMSES Net CML assigns permanent resource identifiers (DOIs or handles) to published source code is to encourage the citation of models in journal papers. Universities and research institutions need to recognize published scientific code as evidence of professional development. It is important to note that credit in research professions is awarded for *dissemination* (i.e., publication) rather than for simply carrying out research activities; this norm needs to apply equally to scientific computing to promote the knowledge sharing that Morin et al. (2012) call for.

CoMSES Net is committed to advancing this agenda with continuing refinement of the CML system. We are looking forward to extending the library to enable linkages to external code repositories that researchers may be using to store their models and archiving models located on standalone websites and servers, and by establishing an archive of runtime environments that will make it easier to learn from legacy code into the future. Though not a trivial task to accomplish, it is vital that the knowledge capital invested into these models not be lost when servers of individual scientists go offline or their websites are moved. Another extension of the CML under consideration is a library of code modules, which are well-defined model components that are can be reused in other models. This will require standardization of metadata on data transfer between modules. In mature fields of computational modeling such a library can be a way to improve quality of model code further. Other extensions under development with support from a new international *Digging into Data Challenge* award are the development of interconnected, web-based analysis and visualization tools to evaluate model outputs. CoMSES Net represents a community of practice for model-based science whose agenda includes promoting open sharing of knowledge about computational modeling.

The extent to which computing is transforming 21st Century science makes it critical that it be included in the culture of intellectual exchange that sets science apart from earlier knowledge systems. To make researchers aware of emerging opportunities for disseminating scientific code, we propose an online clearinghouse for repositories established by scientific communities of practice. An effective starting point could be a simple web page, listing repository URLs, hosted by a national or international organization like the AAAS or NAS. The need for transparency in scientific computing is clear; we must now embed this in the practice of normal science.

Acknowledgments

We would like to thank the National Science Foundation for their support via grants (NSF BCS-0623162 and GEO-0909394). We also want to acknowledge the Center for Social Dynamics and Complexity, the Center for the Study of Institutional Diversity and the University Technology Office at ASU for their contributions to developing the initial Open ABM web portal. Finally we would like to thank the Arizona State University Library for their assistance with digital preservation issues, especially Philip Konomos, Matt Cordial, Stephen Tanner, and Mimmo Bonanni.

Appendix A. Supplementary data

Supplementary data related to this article can be found at http://dx.doi.org/10.1016/j.envsoft.2014.06.022.

References

Alessa, L., Laituri, M., Barton, C.M., 2006. An “all hands” call to the social science community: establishing a community framework for complexity modeling using agent based models and cyberinfrastructure. J. Artif. Soc. Soc. Simul. 9.
Barnes, N., 2010. Publish your computer code: it is good enough. Nature 467, 753.
Edmonds, B., Meyer, R. (Eds.), 2013. Simulating Social Complexity: a Handbook. Springer-Verlag, Berlin.
Gent, P.R., Danabasoglu, G., Donner, L.J., Holland, M.M., Hunke, E.C., Jayne, S.R., Lawrence, D.M., Neale, R.B., Rasch, P.J., Vertenstein, M., Worley, P.H., Yang, Z.L., Zhang, M., 2011. The community climate system model version 4. J. Clim. 24, 4973–4991.
Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M., Müller, B., Pe'er, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., Rossmanith, E., Rüger, N., Strand, E., Souissi, S., Stillman, R.A., Vabø, R., Visser, U., DeAngelis, D.L., 2006. A standard protocol for describing individual-based and agent-based models. Ecol. Model. 198, 115–126.
Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., Railsback, S.F., 2010. The ODD protocol: a review and first update. Ecol. Model. 221, 2760–2768.
Hales, D., Rouchier, J., Edmunds, B., 2003. Model-to-model analysis. J. Artif. Soc. Soc. Simul. 6.
Hayes, B., 2004. Computing science: undisciplined science. Am. Sci. 92 (4), 306–310. http://dx.doi.org/10.2307/27858416.
Helbing, D. (Ed.), 2012. Social Self-organization: Agent-based Simulations and Experiments to Study Emergent Social Behavior. Springer-Verlag, Berlin.
Janssen, M.A., 2007. Coordination in irrigation systems: an analysis of the Lansing-Kremer model of Bali. Agric. Syst. 93, 170–190.
Janssen, M.A., 2009. Understanding artificial anasazi. J. Artif. Soc. Soc. Simul. 12, 13.
Janssen, M.A., Alessa, L.N., Barton, C.M., Bergin, S., Lee, A., 2008. Towards a community framework for agent-based modeling. J. Artif. Soc. Soc. Simul. 11.
Joppa, L.N., McInerny, G., Harper, R., Salido, L., Takeda, K., O'Hara, K., Gavaghan, D., Emmott, S., 2013a. Troubling trends in scientific software use. Science 340, 814–815.
Joppa, L.N., Gavaghan, D., Harper, R., Takeda, K., Emmott, S., 2013b. Optimizing peer review of software code – response. Science 341, 237.
Meadows, M., Cliff, D., 2012. Reexamining the relative agreement model of opinion dynamics. J. Artif. Soc. Soc. Simul. 15 (4), 4. http://jasss.soc.surrey.ac.uk/15/4/4.html.
Miller, J.H., Page, S.E., 2007. Complex Adaptive Systems: an Introduction to Computational Models of Social Life. Princeton University Press, Princeton, N.J.
Morin, A., Urban, J., Adams, P.D., Foster, I., Sali, A., Baker, D., Sliz, P., 2012. Shining light into black boxes. Science 336, 159–160.
Müller, B., Balbi, S., Buchmann, C.M., de Sousa, L., Dressler, G., Groeneveld, J., Klassert, C.J., Bao Le, Q., Millington, J.D.A., Nolzen, H., Parker, D.C., Polhill, J.G., Schlüter, M., Schulze, J., Schwarz, N., Sun, Z., Taillandier, P., Weise, H., 2014. Standardised and transparent model descriptions for agent-based models: current status and prospects. Environ. Model. Softw. 55, 156–163.
Parker, D.C., Brown, D.G., Polhill, J.G., Manson, S.M., Deadman, P.J., 2008. Illustrating a new “conceptual design pattern” for agent-based models and land use via five case studies: the MR POTATOHEAD framework. In: Paredes, A.L., Iglesias, C.H. (Eds.), Agent-based Modeling in Natural Resource Management. Universidad de Valladolid, Valladolid, Spain, pp. 29–62.
Peckham, S.D., Hutton, E.W.H., Norris, B., 2012. A component-based approach to integrated modeling in the geosciences: the design of CSDMS. Comput. Geosci. http://dx.doi.org/10.1016/j.cageo.2012.04.002.
Peng, R.D., 2011. Reproducible research in computational science. Science 334, 1226–1227.
Polhill, J.G., Izquierdo, L.R., Gotts, N.M., 2005. The ghost in the model (and other effects of floating point arithmetic). J. Artif. Soc. Soc. Simul. 8, 5.
Polhill, J.G., Parker, D.C., Brown, D.G., Grimm, V., 2008. Using the ODD protocol for describing three agent-based social simulation models of land use change. J. Artif. Soc. Soc. Simul. 11, 3.
Sliz, P., Morin, A., 2013. Optimizing peer review of software code. Science 341 (6143), 236–237. http://dx.doi.org/10.1126/science.341.6143.236-b.
Strube, T., Benz, J., Kardaetz, S., Brüggemann, R., 2008. ECOBAS – A tool to develop ecosystem models exemplified by the shallow lake model EMMO. Ecol. Inform. 3 (2), 154–169.
Wilensky, U., Rand, W., 2007. Making models match: replicating an agent-based model. J. Artif. Soc. Soc. Simul. 10, 2.
Wing, J.M., 2008. Computational thinking and thinking about computing. Philos. Trans. R. Soc. Math. Phys. Eng. Sci. 366 (1881), 3717–3725. http://dx.doi.org/10.1098/rsta.2008.0118.