EUROPEAN COMMISSION
Community research

is-enes
INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

IS-ENES2 DELIVERABLE (D -N°: D2.5)

ENES governance report on software sharing

File name: {IS-ENES2_D2_5.pdf}

Author(s): **Mick Carter**

Reviewer(s): **Sylvie Joussaume**
Reinhard Budich
Francesca Guglielmo
Sophie Valcke

Reporting period: ***01/04/2016 – 31/03/2017***

Release date for review: ***31/03/2017*** Final date of issue: ***19/04/2017***

Revision table			
Version	**Date**	**Name**	**Comments**
0.1	20/03/17	Mick Carter	Initial Draft
0.2	21/03/17	Mick Carter	Added NEMO example from document provided by Claire LEVY
0.3	21/03/17	Mick Carter	Responded to Claire's comments and changes.
0.4	22/03/17	Mick Carer	Clarify "community governance"
0.5	23/03/17	Mick Carter	Sophie's input
0.6	23/03/17	Mick Carter	Clean up and COSP example added
0.7	24/03/17	Mick Carter	Minor editorial changes plus add a section on standards governance. Sophie's final comments added. Dave's input on the Cylc section.
0.8	07/04/17	Mick Carter	Most of Sylvie and Francesca's changes
0.9	07/04/17	Mick Carter	Minor edits on re-read.
0.10	10/04/17	Mick Carter	Inputs from Sophie on interaction between OASIS governance "layers" and Reinhard on CDO section.

Abstract

All software needs governance and this paper defines and considers *community governance*. That is, the process by which a wider community can influence the decision making and quality control. A number of examples relevant to the IS-ENES2 project are analysed and the paper recognises that a range of approaches is required depending on the software functionality, how it is used, how it is developed and institutional and cultures. Recommendations are made on approaches and things to consider when setting up community governance. This paper also notes that a new approach to start with a governance of standards in advance of software governance may encourage more software sharing.

Project co-funded by the European Commission's Seventh Framework Programme (FP7; 2007-2013) under the grant agreement n°312979		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants including the Commission Services	
RE	Restricted to a group specified by the partners of the **IS-ENES2** project	
CO	Confidential, only for partners of the **IS-ENES2** project	

Table of contents

Executive Summary

Software governance is the process of decision making and quality control for software development. This paper looks at *community governance*. That is, the process by which a wider community can influence the decision making and quality control. In this context, the wider community is the users of the software outside the institutions developing the software.

Within the context of IS-ENES2, the aims of community governance are to increase the likelihood, sustainability and efficiency of software sharing.

This paper looks at a number of examples of the sorts of *community governance* that are either in place or proposed for software that is shared within the ENES or wider community.

By analysing these examples, we can conclude that the existence of community governance is not always a major factor when people choose to adopt software developed by another site. That is, there are examples of software being adopted by non-developer sites with no governance in place and early adoption before governance common. However, the paper argues that community governance can bring benefits to projects through both challenge and support and there are examples where community governance is welcomed despite it not being linked to funding.

The following governance structure, proposed for OASIS, is considered a useful template for mature software aimed at community use:

- The Developer Team covering decision making for day-to-day development.
- The User Group covering user requirements, priorities and experiences.
- The Advisory Board covering strategic direction.

But it is not always possible for the community to mandate a governance structure, especially when there is no sustained funding. Hence, this document gives advice on aspects to consider when setting up community governance structures.

1. Objectives

The objectives of this document were initially envisaged in the IS-ENES description of work as follows:

> *Existing and previous EU funded projects have started the process of encouraging the sharing of software within the ENES community (e.g. the OASIS coupler and the CDO*

processing tool). However, in preparing this bid, it became clear that other opportunities exist and that there is a growing appetite to achieve code sharing as a result of such initiatives as program access to PRACE supercomputing. The Foresight paper provided by IS-ENES has highlighted the benefits of improved collaboration on software. The document also notes that bottom-up approaches have tended to be more successful than top-down approaches. But there is good evidence that bottom-up approaches need substantial support by institutions.

Since then, we have gained a great deal of experience of the challenges of software sharing and the role of community governance in that aim.

This paper analyses a number of examples of software sharing and attempted software sharing looking at the role that community governance has played. In doing this, we need to think what sort of community governance is practical outside the IS-ENES2 project and the funding it has provided. Because of this, the aims of the paper are to provide advice to software owners in a number of different scenarios, as it is not possible to be prescriptive about how software is delivered to a wider community based on a local funding model or expect that a single community governance model will be effective for all situations.

2. The aims of Community Governance

Within the context of IS-ENES2, the aims of community governance are to increase the likelihood, sustainability and efficiency of software sharing. A reasonable definition of governance in this respect is the structure in which decision are made within an activity that will develop and potentially support software. Good governance also implies good quality assurance, but even this comes down to decision making.

For any activity, the governance will aim to ensure the software meets the quality needs of the intended user base in the following respects, both in the short and long terms: functionality, performance, ease of use, platform portability, reliability (from defects and failures), supportability, security and conformance to applicable standards.

The level of control applied to each of these should be based on the following criteria:

- The nature of the software. In particular how large and complex it is.
- How it is used. For example is it a standalone application, a library, a service, a toolbox, a framework etc.
- The criticality of its use, which could range from a “nice-to-have extra tool” within a research project to a vital part of a real-time mission critical system.
- The culture of the organisation(s) and team(s) developing and supporting the software.

Large, complex software that sits at the heart of mission critical systems will require significantly more governance than small, simple software that is considered useful or optional rather than essential.

If we consider an individual planning to use software provided by a third party, the following additional key questions apply:

- What is the nature of the license both from the point of use and, if necessary, the ability to amend the code?
- How is the ongoing development (and support) for the software funded?
- How much effort will be required to evaluate it?
- How much effort will be required to integrate it?
- Do I need support? Can I develop my own support capability? Can I understand the software?
- How long do I intend to use this software for?
- If something goes wrong, is there an alternative approach I can take?
- Can I influence its development?
- Can I contribute to the development of the software, and get my code onto the trunk?

No single approach to governance within IS-ENES2 is possible or would be appropriate and effective because software functionality, how it is used, how it is developed and institutional and cultures vary. The only thing that can be concluded is that it is more likely that people would be prepared to rely on third-party software if the governance of that software is well defined, understood and appropriate for the software and its use. Where good governance exists, we should advertise it well as a key selling point of the software.

3. Examples

3.1 OASIS example

The OASIS coupler library is widely used within the climate community and is part of IS-ENES service on tools. OASIS has been supported by funding from a number of EU funded projects starting with the PRISM[1] project and further optimised in IS-ENES. It also has benefitted from a community-spirited management at CERFACS and stable technical leadership. OASIS is used for both research and mission critical operations and has been designed to facilitate easy integration into existing software systems. It has reached a level of maturity where a 3 tier governance structure is seen to be beneficial after information gathered within IS-ENES2 (see deliverable D4.3). The structure, now being put in place (see Discussion section below), includes a Developer Team for the day-to-day development and quality control, a User Group to feed back on user requirements and experiences with the software and an Advisory Board to consider more strategic issues such as long terms

[1] PRISM "Partnership for Integrated earth System Modelling" December 2001-November 2004

requirements and external drivers. The role of the Developer Team is to regularly collect the User Group's feedback on current needs, sketch a roadmap for the coupler developments, present the roadmap to the Advisory Board and readjust it based on the feedback from the advisory board. Full details are available in the IS-ENES2 website[2].

It should be noted that OASIS became a popular tool well before any community governance structure was in place and community governance has not been a driver or an inhibitor to its uptake although community engagement through various EU funded projects has been useful. Within IS-ENES2 WP4-NA3 there has been an exercise to understand the interest around the community governance needs for OASIS and the level of feedback to that exercise was highly variable, even among OASIS users. This probably says as much about the role of those people involved in responding to the exercise and the culture they work in as it does about the importance of the quality of OASIS. It is possible for good internal governance to be in existence, leading to a good experience, but yet be completely invisible. For people at the working level, a good experience may be enough, but for senior managers responsible for delivery of critical services, there is likely to be a very different attitude to the risks associated with a heavy reliance on software with no visible governance if this is brought to their attention. For example, at the Met Office, OASIS is a key part of mission critical operations and the Met Office had senior level involvement in the response to the questionnaire. It is therefore not surprising that the Met Office is keen to be actively engage in OASIS community governance.

In the case of OASIS, external influence on the governance is seen, by those funding OASIS development, as beneficial for software quality, leading to a stronger offering. This belief is common in a number of successful open source projects.

3.2 Cylc example

The Cylc meta-scheduler is an example of a software system initially designed to be used at the heart of mission critical operational weather forecasting but also used for research activities and has been extended to meet the needs of the climate community. It has received a good degree of interest within the ENES community and presented at IS-ENES2 workshops[3]. However, the majority of its adoption within the climate community relates to use of the Met Office's Unified model. Examples of other climate groups adopting Cylc include NCAR and also GFDL where Cylc is used to automate GFDL's post processing for climate model data flows in a project called Chaco. The high level of interest within the ENES community has led to investment in Cylc within the ESiWACE centre of excellence.

The investment required to use Cylc at a basic level is very low. However, users often prefer to get by incrementally writing scripts of ever increasing complexity rather than learn a new

[2] IS-ENES2 D4.3 https://verc.enes.org/ISENES2/documents/deliverables/is-enes2_d4-3_coupler-governance-model-document/view
[3] https://is.enes.org/events/final-is-enes2-workshop-on-workflow-solutions-1 and https://is.enes.org/events/isenes2-workshop-on-workflows

tool (even if this is much less efficient in the long run). Therefore Cylc is most successful where it is adopted and supported as a strategic decision at the institutional level. However, once institutional knowledge has been gained, it has been demonstrated that the power of Cylc is such that significant overall efficiencies can be made.

Opportunities to make a significant investment such as Cylc adoption are relatively infrequent within an organisation and are likely to need both working level interest and senior level sponsorship. Senior sponsors are likely to seek confidence in the governance of Cylc and the ESiWACE Centre of Excellence provides the advantage of resources to allow this to be explored if institutions come forward within the CoE's period of existence.

Cylc is developed by two main sites, NIWA (New Zealand) and the Met Office. Both set a high bar in terms of the reliability requirements of Cylc because of their operational NWP commitments and hence people can be confident of a robust QA process. Plans and problems reports are all managed in the public domain and there are various ways in which end users can interact with the project. Again, ESiWACE funding can currently ensure proactive engagement with the ENES and partner community.

Because of the robust development, the extension of Cylc use across all Unified Model partners has been relatively straight forward thus providing a strong basis for meeting portability requirements. Further, the range of workflows developed across the current Cylc user base, including ancillary file production and data processing proves Cylc's flexibility for workflows in general.

Although external (to the UM community) influence on long-term priorities is not strong, there is robust user input via these groups and the priorities presented to a wider community by The Met Office at the Lisbon workflow workshop in 2016[4] were accepted as appropriate. However, the teams delivering Cylc are very busy and without the funding that ESiWACE provides it would be hard to see how priority requirements not common to the core community could be met from the existing Cylc team. The Cylc core user base use Cylc across a very wide range of applications and, as a result it is well tested across a very broad range of functionality. Hence, it is unlikely that significant new requirements would emerge as a result of a wider adoption. Further, Cylc development from outside the Met Office/NIWA teams is possible.

3.3 CDO example

CDO is a collection of command-line operators used to manipulate and analyse climate and numerical weather prediction data. It includes support for netCDF-3, netCDF-4 and GRIB1, GRIB2, and other formats. CDO has been part of IS-ENES service since its first phase with an increasing usage. The final IS-ENES2 general assembly concluded that the CDO are an

[4] https://is.enes.org/events/final-is-enes2-workshop-on-workflow-solutions/slides/26Matthewsesiwace.pdf

excellent example of *heroic development,* and of community driven support. That is, a single-handed developer successfully identifies the (evolving) user requirements and delivers solutions with no formal user engagement or visible community governance, relying however on occasional user feedback (see below).
The nature of the tool and its application within workflows lead to the perception that the risk of this open-source/self-propelled approach be low, thus not currently limiting CDO's adoption nor its use in any way. This whole successful model –collecting satisfactory community feedback - could be however at stake if ownership were to change hands or should the developer of CDO decide to cease their activities.
Moreover, if on one hand the investment required to use CDO at a basic level is very low, on the other hand a future, possible institutional reliance on it could be not sustainable if it would not be accompanied by adequate support. Support for use and maintenance as well as community involvement via the user forum for bug fixing, maintenance, answering FAQs, raising issues and feature demands, and further help-desk-like activities were solely put in place thanks to various ENES-initiated projects.

3.4 COSP example

COSP is a diagnostic tool that is used in climate models to facilitate the comparison of model simulations with satellite observations. COSP was developed by an international collaboration of scientists and recommended for use in CMIP5[5]. IS-ENES2 supported its optimisation for use in CMIP6. The use of COSP grew in advance of a formal governance model. However, as the number of users grew and COSP became an important tool used by modelling centres, it became clear that a more formal management process was needed. A meritocratic model was adopted to provide a management structure in the project and to clarify the decision making process. This defines different roles based on a willingness and capability of people to contribute to the project. Roles include users, contributors, committers, and members of the management committee. The Project Management Committee (PMC) was created to provide direction to the project, incorporating all the main contributors to the project (8 people). Soon after the implementation of this governance model, the PMC worked on the following tasks:

- The code was moved from an internal, private repository, to a public, version-controlled repository. This provides clear traceability of code changes.
- A more open license was adopted (BSD license). A restrictive license is a big hurdle that reduces the usability of the code.
- A users' group was created. This group facilitates the interaction between the community and the provision of technical support.

In recent times, the user base has expanded, and more importantly, some of the community members (not only the PMC members) are becoming active developers. During this process, we have learnt the following lessons and are working towards their implementation:

[5] http://cmip-pcmdi.llnl.gov/cmip5/output_req.html#cosp

- Developers need clear working practices. The working practices provide clear guidance and facilitate and implementation of new developments. The working practices describe the development, testing and review process. This avoids surprises and frustration from developers.
- A clear decision-making process is needed. The governance model and working practices define the decision-making process.
- The regression tests need to be simple and robust. Current regression tests require a substantial amount of human intervention, which is not practical. Robust software development techniques, along with publicly available tools can greatly improve this aspect.

The COSP community has recognised that adopting standard Quality Assurance processes will be beneficial for the success of the project in the long term. Although they may be seen as 'red tape', the main aim is to streamline the development process, making it more efficient and reducing the current bottlenecks.

This should reduce the involvement of the PMC members. It is worth noting that COSP doesn't have secure, long-term funding, so the level of involvement of PMC members is very variable, and therefore reducing the effort required by PMC members in any new development is a priority of the project.

3.5 NEMO example

NEMO is a numerical platform for ocean modelling (dynamics, biogeochemistry and sea-ice) which is part of the service on models within IS-ENES2. Its development is led by investment of a consortium of 6 institutions (CNRS, NOC, CMCC, Mercator, Met-Office and INGV) located in three countries. The partners of the consortium cover a range of focus from ocean research all the way through to operational delivery. This has led to a very well respected ocean model used across a wide range of spatial and temporal scales and used in CMIP models by 9 institutions. The development of NEMO is governed by the NEMO consortium and each consortium member provides at least one full time person of effort to the NEMO System Team.

The governance structure includes a Steering Committee which is the NEMO executive board. The NEMO System Team includes computational and natural scientists and their job is the sustainable development of NEMO including all quality assurance aspects. The Scientific Advisory board (NEMO Developer's Committee) gives advice on the ongoing work and the yearly work-plan. The Advisory board is also in charge of writing and updating the NEMO Development Strategy document describing the development strategy for the next five years. Working groups are formed to deal with specific scientific and technical challenges and ongoing functions (such as configuration manager), and to answer questions or provide expertise for some development choices.

There is a culture of continuous improvement of the development process that has benefitted from the experience of 8 years since the creation of the consortium and supported by strong and stable technical and scientific leadership. There are well defined and published working practices.

For the consortium members, there is a strong driver to ensure that the Consortium continues to deliver a world-class ocean model able to be used in a variety of configurations. This is a major strategic decision for the consortium members and leads to a high level support at the Consortium institutions leading to an appropriate level of investment overseen by senior managers to meet this strategic aim.

The focus of NEMO development is a consortium focus rather than a community focus, but the consortium has broad enough aims for NEMO to be used by a much wider community with more than 1000 registered users and 200 projects world-wide. There is a user group, occasional surveys and a science forum that provide the opportunity for the wider community to give input to consortium decisions, but decisions are made by consortium structures.

4. Discussion and Conclusion

The best established community governance in our examples is the NEMO ocean model. Here, the consortium members have a very large stake in the direction of development of the model and it is in the interest of the consortium to work as a single body. The governance for NEMO is best described as consortium governance with community influence. The executive make consortium-based decision but there is opportunity for wider influence through a user group, community surveys and a science forum.

4.1 Networking as a precursor to governance

Networking activities do encourage software sharing and can provide influence, but do not in themselves constitute governance. Networking is often the most effective way to initiate code sharing and can lead to community engagement and possibly community governance.

4.2 How important is community governance in encouraging software sharing?

Experience within the IS-ENES2 community (see CDO and the early stages of OASIS and COSP) suggests that community governance is not necessary for software sharing to take off. Other factors will have a greater impact, such as the quality and usefulness of the software, the lack of local solutions, the ease of adoption, the drivers for change that encourage people to look for externally developed software, individual interests and institutional cultures. All these attributes suggest effective governance, but not necessarily community governance.

The fact that ENES has seen success with no community governance in place does not mean there is no benefit in community governance. Any management process has the potential to be enhanced by being both supported and challenged to improve decision making or working practices. Community governance is one way to achieve that by bringing ideas, drivers and interest from a wider community. Embracing these *outside* influences can be viewed as an act of enlightened self interest – the project is using external resources to benefit its own aims as well as the aims of the wider community. The engagement will hopefully lead to better testing, portability and future proofing which can reduce the risks for the primary developers.

If strong relationships between institutions are built through this sort of engagement, it is possible that reciprocal benefit could result - "I provide some software and receive other software in exchange". There is little direct evidence that people make decision based on such reciprocal arrangements but mutually beneficial relationships help build trust.

4.3 Striking the right balance for community governance

Getting effective engagement to support a community governance structure is not a trivial matter and a balance needs to be struck to as not to discourage engagement by asking for too much input. However, there needs to be enough engagement to have influence.

When there is no option for community governance, this paper recommends making the existing project governance and quality assurance processes visible to the wider community to build confidence in the quality processes.

The gold standard proposed within IS-ENES2 for community governance is that about to be put in place for OASIS. This has three tiers:

- The Developer Team covering working practices.
- The User Group covering user requirements, priorities and experiences.
- The Advisory Board covering strategic direction.

This separation of roles is considered useful to gain the support and challenge from different angles. For less mature community interactions, a two layer structure is probably more likely to be successful.

4.4 Community governance check-list

The following issues should be considered when implementing a community governance structure:

- User groups and Advisory Boards are best chaired by a member of the user community rather than someone from the developer community. But any chair is better than none where an external chair is not an option.

- Face to face meetings are a significant overhead but have advantages if you want to build relationships or seek more engagement for activities such as testing. Test the community appetite for face-to-face meetings and use workshops and the like as an opportunity to meet.
- Surveys can be useful tools and should get input from outside the community in terms of their design.
- Development priorities and problems management "tickets" should be as open to the community as possible.
- Methods of contribution.
- License conditions and their implications for community users.
- It is important to remember to get input on non-functional requirements such as:
    - Performance
    - Ease of use (including documentation)
    - Interface design (APIs)
    - Reliability
    - Robustness

4.5 Community governance of standards - a possible complementary way forward

Within IS-ENES, a discussion that was started at the 4th ENES HPC workshop[6], in particular the presentation by Mike Rezny[7], and continued at the "Crossing the Chasm" workshop[8] concluded that a new approach that could more effectively lead to shared codes would be to set up community governance structures around standards. Such standards would be shared across codes out of which dominant shared codes could emerge, as it would be easy to move between solutions working to a common standard. This provides a different type of top-down influence to support the bottom-up approach that is already recognized as successfully leading to software sharing. Such initiatives have been successful in the past, for example in the development of MPI. For example, all climate models need to understand calendars and a common standard could be defined for both the interface and the functionality of a calendar library. If people adopted such standards, it would be trivial to move to a single shared version of the code. The effort required to define standards is not trivial and could be a new focus for successor projects following IS-ENES2.

[6] https://is.enes.org/events/4th-enes-hpc-workshop
[7] https://www.esiwace.eu/events/4th-enes-hpc-workshop/abstracts/on-paths-to-exascale-will-we-be-hungry
[8] https://is.enes.org/events/save-the-date-crossing-the-chasm-towards-common-software-infrastructure-for-earth-system-model-development