Sprint Documentation #22

Couple FABM and REcoM3 (COFARE)

Wilton Jaciel Loch[1], **Özgür Gürses**[2], **Iris Ehlert**[1], **and Judith Hauck**[2]

[1] Deutsches Klimarechenzentrum GmbH, Hamburg, Germany
[2] Alfred Wegener Institute, Helmholtz Centre for Polar and Marine Research, Bremerhaven, Germany

Contact: info@nat-esm.de

Published on 20.09.2026 on https://www.nat-esm.de/services/support-through-sprints/documentation

1 Summary

The sprint focused on implementing the Regulated Ecosystem Model (REcoM) within the Framework for Aquatic Biogeochemical Models (FABM), while simultaneously improving the software engineering and modularity of the original model. The sprint resulted in a first working prototype of a standalone REcoM implementation in FABM, supported by substantial refactoring to decouple REcoM from FESOM, the introduction of a standalone build system, continuous integration workflows and improved code quality tooling. During the implementation, we also gained practical insights into the advantages and limitations of the FABM architecture, particularly regarding code reuse and long-term maintenance of existing biogeochemical models. Although scientific validation of the implementation remains future work, the sprint provides a solid technical foundation for further development and contributes valuable experience for ongoing discussions on ocean biogeochemistry coupling within natESM.

2 General Information

Start and end date:	October 2025 – June 2026
Intended period:	06 months
Responsible RSE:	Wilton Jaciel Loch, DKRZ
Responsible scientist:	Judith Hauck, AWI

The Regulated Ecosystem Model (REcoM - version 3, Gürses et al. 2023) is a water column biogeochemistry and ecosystem model. The model combines cycles of carbon and nutrients (nitrogen, iron, and silicon) and oxygen with varying intracellular stoichiometry in phytoplankton, zooplankton, and detritus. It has been developed at the Alfred Wegener Institute, Helmholtz Centre for Polar and Marine Research (AWI), in Bremerhaven, Germany for about two decades (since Schartau et al., 2007). REcoM has been successfully used in many studies on the ocean carbon cycle, the biogeochemical environment and the marine ecosystem (Hauck et al., 2013, 2020; Schourup-Kristensen et al., 2014; Völker and Tagliabue, 2015; Nissen et al., 2022; Seifert et al., 2022, 2023; Karakus et al., 2021, 2022; Butzin et al. 2024; Oziel et al. 2022, 2025).

REcoM can be run in configurations of varying complexity with currently up to four phytoplankton functional types (PFTs), namely diatoms, coccolithophores, small phytoplankton and phaeocystis, up to three zooplankton types (micro-

meso and polar macrozooplankton) and up to two detritus classes (slow- and fast-sinking). REcoM can also simulate carbon and iron isotopes, and can be used coupled to the sediment model Medusa (Ye et al., 2025). REcoM is coupled to FESOM and MITgcm and is the ocean biogeochemistry module of the AWI Earth System Model. It is used for hindcasts, CMIP-type future projections (e.g., Mongwe et al., 2024) and paleo applications.

The Framework for Aquatic Biogeochemical Models (FABM, Bruggeman and Bolding, 2014) is an open-source coupler that provides an interface between the biogeochemical/ecological model and the ocean circulation model. Information is exchanged between the models at runtime. FABM is also used as a platform to develop complex biogeochemical models in an isolated form and to assess them coupled to different physical models or forced by prescribed circulation fields. It also provides a 1D framework, and is used in both 1D and 3D versions for multi biogeochemical model comparisons with the same ocean physics.

Some more technical details: REcoM3 is a water column model. It computes the sink and source terms for the tracers at every timestep and updates the concentration of the biogeochemical state variables (e.g., nutrients, dissolved and particulate organic matter). Updated concentrations are passed to the general ocean circulation model (Finite-volumE Sea ice-Ocean Model, FESOM2, Scholz et al., 2019) where computationally expensive advection and diffusion of the state variables take place.

3 Sprint Objectives

The sprint goal was to implement REcoM as a biogeochemical model within the FABM framework, and if time allowed, replace the calls to REcoM inside of the FESOM ocean model with calls to the FABM framework. This would allow a more general coupling between FESOM and REcoM, while at the same time allowing REcoM to be run with other general circulation models, and compared with other biogeochemical models from the community.

4 Procedure and Insights

4.1 Technical Approach / Procedure

Already from the proposal, a number of steps have been identified as required for the implementation. These included rearranging the REcoM code to fit the FABM “do” subroutine, adapting the REcoM variables to use the FABM standard naming, detaching processes that rely on FESOM infrastructure and more.

From a high-level point of view, the first requirement to be able to implement REcoM as a FABM model is the ability to use REcoM code inside of FABM. Whilst there are parts of REcoM which would prompt a restructuring to be usable inside of FABM, there are also parts which could be directly used in the framework, if included there. These include certain subroutines, but also variables publicly defined inside of REcoM modules. The problem here is that although REcoM is a model of its own, it has been historically developed as a part of FESOM, and has resided in the same Git repository, sharing the same infrastructure and having explicit and implicit dependencies on its code. In preparation/parallel to this sprint, REcoM was moved into its own GitHub repository. So, in order to include REcoM code in FABM, we can either bring the dependencies from FESOM together with it, or remove these dependencies so that REcoM can be built and included as a standalone. We chose to move with the second option as it also brings benefits to REcoM outside of the FABM implementation, like the possibility of having a standalone driver in the future, as well as improved testing opportunities. So the first part of the technical work consisted in increasing the separation between REcoM and FESOM, so that it could be built as a library to be linked against the FABM implementation. Since in the beginning of the sprint a copy of the REcoM code was already placed in a separate repository, we decided to do the development there, as having it separate was in any case the final goal. This version would later replace the inline REcoM in FESOM as an external submodule.

As this work would involve changes to the REcoM calls in FESOM, which now lived in a separate repository and had gatekeepers who were not directly involved in the sprint, we divided the approach in two parts. The first focused only on the REcoM internal parts of the code, that could be individually and more directly changed without effects on production runs. The second part would then involve changes to the public subroutines of REcoM, that would have to

be coordinated with changes in FESOM. The diagram in **Figure 1** shows the file organization of REcoM and the dependencies between its modules. It also shows which subroutines in REcoM are being used by FESOM, and in which files they are called.

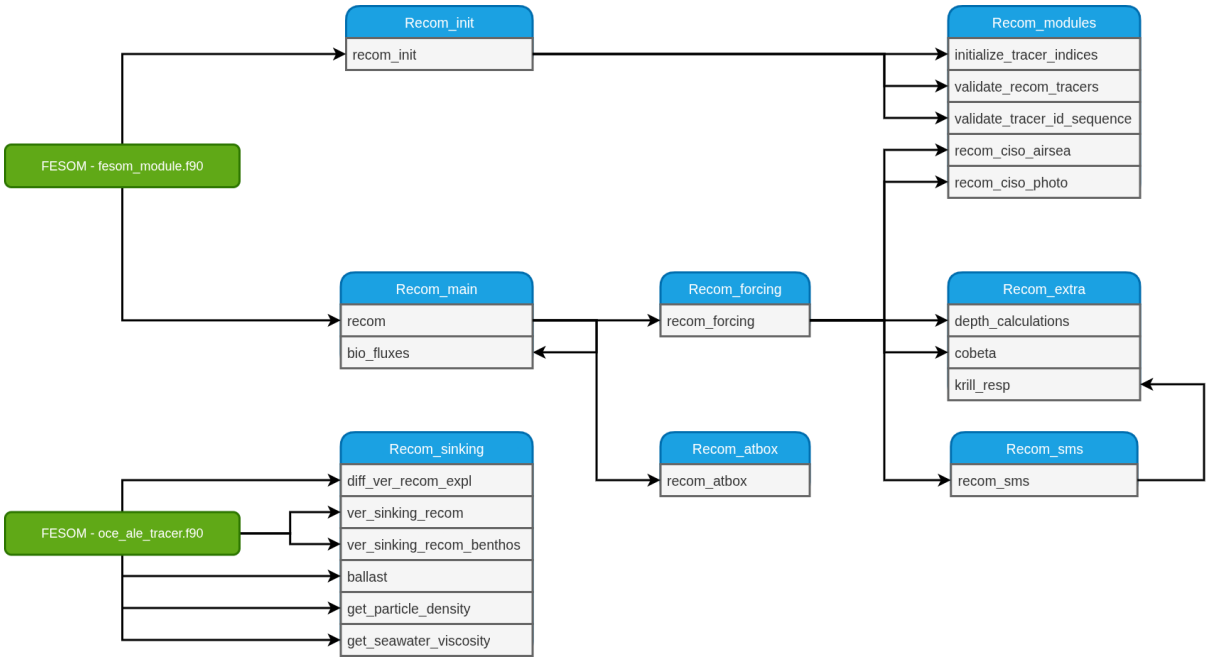

Figure 1. REcoM files, subroutines and call graph in relation to FESOM.

The dependencies of REcoM on FESOM consisted of three main factors. The first were data structures that hold information of the simulation, like the mesh, partitioning, state variables and others, that were passed from FESOM to REcoM calls as parameters. The second were FESOM modules with parameters and constants that were directly included in the REcoM code. The third were utility subroutines for data exchange, aggregation and others, also directly called from the REcoM code. The strategy for dealing with data dependencies in general is to move them from includes, either types or actual data, to arguments passed to the subroutines that need them. So, for scalars and constants from FESOM, they were simply added as new arguments to all parent REcoM subroutines up to the public one, which is externally called. A similar strategy was used for derived data types, but we first extracted all of its members and turned them into arguments. For the subroutines the strategy was either to replace them with simpler native alternatives, like custom finalization subroutines replaced by the MPI abort, or to create copies to exist in the REcoM code. After performing all of these changes we also added CMake build support, which ultimately allowed REcoM to be built as a library, and its public data structures and subroutines to be usable from other models.

The next stages focused on improvements to organization, development workflows, software quality and other ancillary aspects, which are not fundamental to the goal of the sprint, but hugely beneficial for the model. This included adding CI infrastructure, with two workflows, one for building the newly created REcoM library, assuring that future changes to the model will keep it building correctly; and one for checking a number of software quality issues and prevent their inclusion into the main trunk. For the latter we have introduced the pre-commit tool – already used in ICON – that checks smaller issues like avoiding large files to be added to Git, YAML file syntax, preventing extraneous whitespaces and blank lines, etc. On top of that we introduced two additional tools to the CI checks. The first was the Codee Fortran formatter – also used in ICON – that can automatically reformat the code and keep a consistent coding style across all files. The second was Fortitude, a Fortran linter capable of checking deeper problems related to code syntax and semantics. After the introduction of such tools the code also had to be modified to adhere to their standards.

To continue with the focus on the FABM integration, parts of the REcoM code would have to be refactored. This would not only serve the integration purpose but would also in many cases simplify the code and improve its usage in the

standalone case. The changes focused mainly on the REcoM initialization and on the "sources minus sinks" subroutine, which is the bulk of the model's logic. This stage included division and simplification of subroutines, changes from vector- to scalar-based functions, fusion of code blocks, simplification of conditionals and more.

After this refactoring round, the actual implementation of REcoM code in FABM could then begin, starting with adapting the basic biogeochemical model template from FABM for REcoM and structuring the build workflow. In order to run FABM, a general circulation model is needed to drive the framework. We chose GOTM for our tests – a single column model – since it seemed like the simplest available option and had fairly good documentation. The build setup then works as follows: first, we build REcoM as a static library, creating *librecom.a*. Then, we build GOTM, but with a flag pointing to an external location of FABM, to be built together with the model. Also at this stage, we pass additional flags that will be used in the FABM compilation, that point to the location of the *fabm_recom* implementation code and to the location of *librecom.a*, tying everything together to produce the final model executable.

To run this setup, we had to generate configuration files for GOTM and FABM, which could also be done following their respective templates. After being able to build and run everything together, we could then incrementally move the code from REcoM to the FABM implementation.

During a significant part of the sprint and especially during the refactoring portion, the goal was to try to have a single version of the code, that could be used both when calling REcoM through its public interface – currently done only by FESOM, but potentially also by other models in the future – and when using REcoM in FABM. This is challenging mostly due to the fact that FABM prescribes its biogeochemical models to operate as dimensionless kernels, which are then applied over the spatial domain of the host model by the FABM infrastructure. Meanwhile in REcoM, like many other biogeochemical models which were not born directly into FABM, most of the logic relies on spatial information retrieved directly from the host general circulation model, with explicit loops and usage of spatial variables. Unifying these two conflicting expressions of the same logic quickly showed to require large engineering efforts, which would still likely not result in full reusability of the same code. This point is further explored in the discussion of section 4.2, and served as one of the basis for choosing a different strategy: developing the FABM implementation as a dedicated standalone version.

From this point onwards, the development progressed first with the import of critical infrastructure from REcoM, such as parameters, variables, and namelist reading functionality. Then, the code was brought part by part from the "source minus sinks" module of REcoM to the main "do" loop of the FABM implementation, with modifications introduced as needed to operate within the FABM restrictions. Most of the code brought from REcoM could be directly implemented in FABM in this stage. The only exception was the light attenuation calculations, which had to be implemented as a dedicated FABM model due to the vertical dependencies of its calculations.

4.2 General Insights

The sprint has provided us with valuable information into how FABM can be used, how to implement models into it and some of its limitations. The first important point to discuss is the nature in which FABM requires its biogeochemical models to be implemented within the framework.

All biogeochemical modelling logic should be implemented as scalar kernels that in principle operate on a single point in space and have no access to global spatial or geometric information. In other words, the spatial domain should be transparent to the biogeochemical processes implemented in the code, and the FABM runtime then takes care of running these kernels over the domain, be it 0D up to a full 3D domain. The main advantage of this design is that the code can be more focused directly on the logic of the processes, and require less scaffolding for dealing with spatial information. This also shields the biogeochemical application code from having to deal with multiple different ways of encoding spatial data, direction and size of dimensions and other implementation-specific details of ocean models. The main disadvantage of this approach is that the code has to be written in a specific way, usually directly into the FABM implementation, that is often incompatible with how most biogeochemical models are written. This disadvantage has several consequences that we will discuss further.

Biogeochemical models depend heavily on information coming from ocean models, and often they are developed within the framework of a "parent" ocean model. This is the case for REcoM with FESOM and for HAMOCC with ICON, for example. In this environment, these codes are often tightly coupled with their parent ocean models, and often depend on their variables, subroutines and infrastructure. This means that they are often relying on access to tracers defined over the full domain, using spatial variables as dependencies for calculations and managing loops with explicit spatial information. As a consequence, when such a model is implemented within FABM, most of the code has to be refactored or "translated" to work in a scalar-based fashion, without explicit spatial loops, and without reusing dependencies from the parent ocean models. This creates a complex scenario in terms of code maintenance, and there are two main options on how to deal with it.

The first is to completely translate all of your code into the FABM implementation, and rely on it for all the simulation needs, abandoning the original version. This is easier to do with models which are being written from scratch that do not yet have a user base relying on the original model for running experiments and to produce science. For existing models this takes a lot of effort and becomes even more difficult if there is already an established ocean model which relies on them as providers of biogeochemical processes and data. It also raises questions related to the independence and autonomy of the models, because even if the FABM development is open and responsive, the ownership and governance become external to the model, and its individual needs might not be tackled with the same freedom as if the model was a standalone.

The second option is to somehow maintain two different versions of the code, and there are multiple ways of doing this. This approach is probably the most cost-ineffective one, as one has to solve bugs, implement features, add comments and documentation twice for every new development. Of course, it is possible to *strive towards* a single unified version of both codes, trying to create subroutines that can be used on both sides, and data structures that can be fed either directly in standalone operation or through FABM. However, getting to a single reusable version both for FABM and standalone runs is likely not possible or not practical.

Since most existing biogeochemical models cannot rely uniquely on FABM due to multiple reasons, they usually go with the two versions approach. But, as maintaining two updated versions of the same software is normally very costly, a snapshot of the model is taken and translated into FABM, being crystallized in time and not necessarily receiving further updates. This allows this version of the software to be part of the FABM ecosystem, but hinders its evolution in terms of new features, bug fixes, etc. This is potentially the path of least resistance for models and that's why many biogeochemical models listed as implemented in FABM were done in this way (such as ECOSMO, iHAMOCC, MEDUSA, MIZER, MOPS, PISCES and potentially others).

In the end, this was also the approach selected for this sprint, with the creation of a standalone version of REcoM specifically for FABM, which will not be so easy to be maintained with current developments in the original version, but will allow the model's participation in the ecosystem, and its execution with both other ocean and biogeochemical models.

From a maintenance perspective this approach is still not satisfying, and from this point of view, it is recommendable for a model to make a choice between either completely relying on FABM or not using it at all. However, from a scientific perspective there is of course value in having a version of the software implemented within the FABM framework, allowing its usage and comparison with other models. The decision should then be made weighing the scientific benefits of the implementation, against the upfront work of translating the model and the potential costs for maintaining it – or losses due to missing updates. Perhaps models that evolve more slowly have less of a penalty for using such an approach compared to models that change more quickly. Other similar aspects should be taken into account when considering a FABM implementation.

Another point to be evaluated is the readiness of implementations listed on the FABM website as part of the framework. ICON-O for example, is listed there as available, but the only known FABM version of ICON-O is in a very experimental state not fit for larger production experiments. It is also based on an upstream version from 3+ years ago, being therefore quite an outdated snapshot of it. As a consequence, this naturally also raises questions about the readiness of other models listed as part of the framework, and the overall reach of the tool.

5 Results

The sprint has been successful from the technical point of view in producing a first prototype of an implementation of REcoM in FABM. The results have not yet been properly validated, since from the technical side there was no possibility to have comparisons between the original FESOM/REcoM experiment and the GOTM/FABM/REcoM one. Naturally, the scientific validation of the implemented setup has not been performed, and is a task for the responsible scientists in the future.

Apart from the main sprint goal, the work has also provided a number of ancillary improvements to REcoM, including the removal of explicit dependencies on FESOM, the addition of a standalone build system with corresponding library build, addition of build and linter CI to the REcoM standalone repository, refactoring and simplification of multiple parts of the code and in parallel the migration of REcoM from an inline folder to a submodule in FESOM.

6 Conclusions and Outlook

Apart from producing a first implementation of REcoM in FABM and improving the REcoM code, we have gathered knowledge on FABM and how it works, what it can provide to the community and some of the limitations of its usage. This gathered experience will be valuable for the ongoing discussions on the consolidation of ocean biogeochemistry within natESM. Options like FABM should be evaluated, but also other general approaches, like a shared interface for biogeochemical models, similar to what ComIn does for ICON-A. The main advantage of such an interface in comparison to FABM would be its flexibility. All models could likely continue to work in mostly the same way they do now, especially in terms of data handling, only with minor modifications required.

In REcoM there are still many possibilities of further work, including improving the technical standards, organization and engineering. Especially for the FABM implementation, additional work could be done on pursuing higher unification between the FABM and the original version, so as to minimize code duplication.

More generally, a decision should be made in natESM regarding supporting FABM or not for future sprints related to biogeochemistry and ocean coupling. The information provided here should serve as a base for starting this discussion and for defining the path forward.

7 References

[0] FABM website: fabm.net

[1] FABM REcoM implementation: https://github.com/RECOM-Regulated-Ecosystem-Model/fabm_recom

[2] REcoM standalone repository: https://github.com/RECOM-Regulated-Ecosystem-Model/REcoM