The Journal of Open Source Software

Heta compiler: a software tool for the development of large-scale QSP models and compilation into simulation formats

Evgeny Metelkin[1]

1 InSysBio LLC

DOI: 10.21105/joss.03708

Software

- Review
- Repository
- Archive

Editor: Frederick Boehm

Reviewers:

- @martinmodrak
- @elimillera

Submitted: 20 August 2021
Published: 11 November 2021

Summary

Today mathematical modeling is becoming more and more popular in biomedicine and drug development. **Quantitative systems pharmacology** (QSP), a relatively new research discipline, is devoted to complex models describing organisms, diseases, and drug dynamics. Designing these models presents a set of challenging methodological problems like managing a huge amount of data, dealing with large-scale models, time-consuming calculations, etc. **Heta compiler** is a small and fast software tool written in JavaScript which manages infrastructure for QSP modeling projects. The purpose of the tool is to build and integrate QSP platform modules, to check their completeness and consistency, and then to compile everything into runnable code that will be executed in simulation software. A user can apply a command-line interface to run the model building process. Alternatively, Heta compiler can be used as a package for developing web-based applications or be integrated with simulation software.

Statement of need

The large and still growing Systems Biology (SB) and Systems Pharmacology modeling communities utilize a variety of software tools for simulation and data analysis (Knight-Schrijver et al., 2016; Mentré et al., 2020; Stéphanou et al., 2018). Usually, the modelers solve the algebraic-differential equations or perform parameters identification or sensitivity analysis. While being useful for tackling specific problems, each software tool often has no user-friendly way for routine operations like step-by-step model creation and maintenance. Furthermore, different tools have their own internal model format which cannot be reused.

This paper presents Heta compiler which provides a convenient and flexible way for the development of dynamic large-scale models based on the **Heta language** code. The compiler translates the source modeling code into a variety of formats to be run in simulation software tools. Heta compiler also provides information on errors in a model which can be used to debug.

This tool is an effort to resolve the typical problems in a QSP project by creating a controllable working environment. The pre-formulated requirements are:

- store QSP models and data in integrated infrastructure,
- support iterative platform updates,
- support of models written in human-readable formats as well as in tables,
- help for model code reuse and sharing,
- provide interface for storing several models in a single platform,
- export models and data to different popular formats so it can be used out-of-the-box.

Heta formats

`Heta compiler` has been evolving alongside the Heta language (Metelkin, 2019) specification. Heta is a series of human-readable and writable formats for QSP and Systems Biology projects: Heta code, table representation, JSON, and YAML notation. Heta describes dynamic models in the process-description format i.e., as interacting components that describe volumes, concentrations, amounts, rates. On the other side, it was designed to be easily transformed into ODEs or other formats.

The standardization of process-description modeling notation was also pursued in formats like SBML, CellML, Antimony. However the Heta standard can be distinguished by the specific features:

- Human-readable/writable code that can be used for model development or modification.
- Easy code parsing.
- Modularity: QSP/SB platform can be subdivided into several files and spaces for better project management.
- Multiple interchangeable representation: human-readable code, tables, JSON, YAML.
- Reusability: modeling platforms should be easily extended for other projects.
- Reach annotation capabilities for better model code revision.
- Simple transformation to popular modeling formats or general-purpose ODEs.
- Support of translation from/to SBML (Hucka et al., 2003).

Example

Code in Figure 1 is an example of the Heta code describing a simple one-compartment model. The metabolic scheme of the model can be found in Figure 2.

```
/*
  Hello World!
*/
comp1 @Compartment .= 1;
A @Species { compartment: comp1 };
B @Species { compartment: comp1 };
r1 @Reaction { actors: A => 2B };

// math
A .= 10;
B .= 0;
r1 := k1*A*comp1;

k1 @Const = 1e-3;

#export { format: SBML, filepath: sbml };
#export { format: Mrgsolve, filepath: mrgsolve };
```

Figure 1: Model code in Heta format: `index.heta` file.

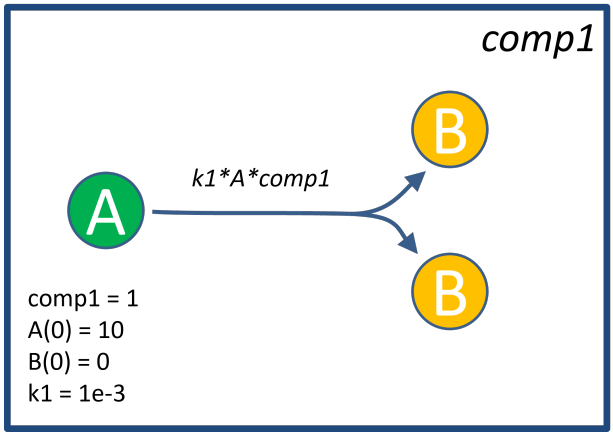

Figure 2: One compartment model with two metabolites and one reaction.

When the size of a model code is large it is recommended to subdivide it into modules but this model is small and can be placed into a single file, e.g `index.heta`. To build the platform with `Heta compiler` one can run the compilation with the following command in a command terminal: `heta build`.

Features overview

`Heta compiler` includes the parser of the Heta formats and supports all features of the Heta specifications of version 0.4.1. It was designed to support exporting to different popular modeling formats. The current version supports the following formats:

- DBSolveOptimum
- SBML of levels 2 and 3
- mrgsolve
- Simbiology
- Matlab describing ODEs file
- Julia language code
- JSON/YAML
- Excel sheets

`Heta compiler` can work in two modes: as a command-line tool for model development or as a library to be incorporated into third-party applications. The source code is written in pure JavaScript and can be run in the Node environment or in a browser. It can be used for both: server-side and front-end applications.

To use `Heta compiler` in a modeling project a user should follow the specific formats and agreements. Project files i.e. model code, datasets, figures, etc. should be stored in the same directory. This directory typically includes an optional `platform.json` declaration file that stores the supplementary information of a platform as well as specific parameters for platform compilation. The alternative way to set the options of the compiler is to use command-line arguments. The list of them can be shown with `heta build -h` command in a shell.

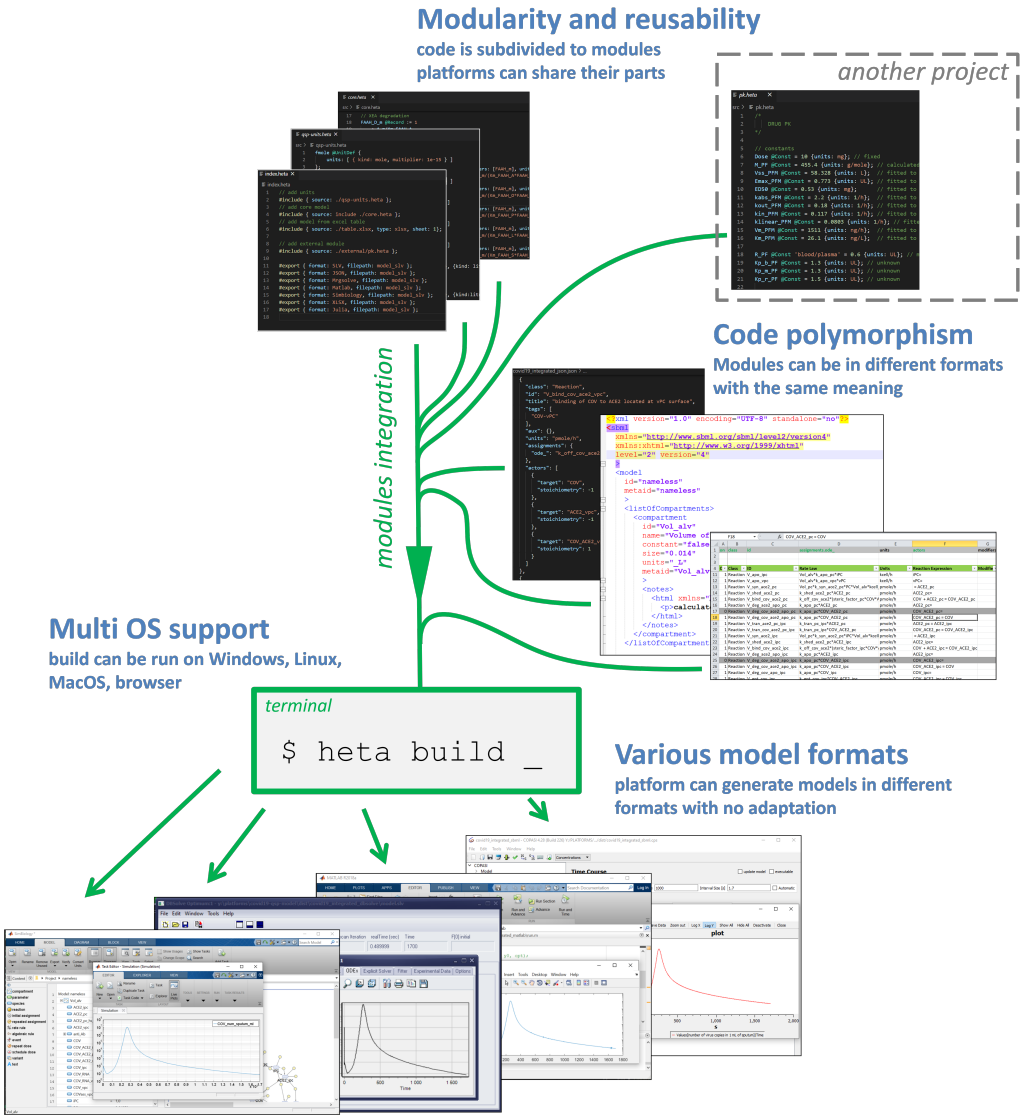

Figure 3: A typical workflow of `heta compiler` in a modeling project.

Results and discussion

`Heta compiler` can be used as the framework for a QSP modeling project of any size and complexity. Currently, it is applied for the development and maintenance of a variety of commercial and open-source modeling projects (Metelkin et al., 2019; Metelkin & Demin, 2020). `Heta compiler` has also been used for the development of web applications like the Immune Response Template navigator and "PK/RO simulator" R-Shiny application (Shchelokov et al., 2019).

The Heta-based formats are friendly for version control systems like Git and SVN because of the modular structure and the text-based representation. Heta compiler can easily be integrated with existing modeling infrastructure, workflows or used as a part of a CI/CD workflow.

`Heta compiler` is a part of the Heta project which is an initiative for the development of full-cycle infrastructure for modeling in pharmacology and biology: https://hetalang.github.io.

References

Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., SBML Forum:, and the rest of the, Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., Goryanin, I. I., Hedley, W. J., Hodgman, T. C., … Wang, J. (2003). The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. *Bioinformatics*, *19*. https://doi.org/10.1093/bioinformatics/btg015

Knight-Schrijver, V. R., Chelliah, V., Cucurull-Sanchez, L., & Novère, N. L. (2016). The promises of quantitative systems pharmacology modelling for drug development. *Computational and Structural Biotechnology Journal*, *14*. https://doi.org/10.1016/j.csbj.2016.09.002

Mentré, F., Friberg, L. E., Duffull, S., French, J., Lauffenburger, D. A., Li, L., Mager, D. E., Sinha, V., Sobie, E., & Zhao, P. (2020). Pharmacometrics and systems pharmacology 2030. *Clinical Pharmacology & Therapeutics*, *107*. https://doi.org/10.1002/cpt.1683

Metelkin, E. (2019, October). "Heta" is a new declarative language to define the large-scale systems pharmacology and systems biology models. *American Conference on Pharmacometrics 10*.

Metelkin, E., Bagrova, N., Benson, N., Demin, O., & Graaf, P. van der. (2019). FAAH inhibitor: QSP modeling platform describing fatty acid amide hydrolase inhibition in human. In *GitHub repository*. GitHub. https://github.com/insysbio/faah-inhibitor

Metelkin, E., & Demin, O. (2020). QSP model of COVID-19: SARS-CoV-2 virus and host cell life cycles, immune response and therapeutic treatments. In *GitHub repository*. GitHub. https://github.com/insysbio/covid19-qsp-model

Shchelokov, D., Metelkin, E., & Jr, O. D. (2019). PK/RO simulator for anti-PD-1 mAbs. In *ShinyApps.io*. InSysBio LLC. https://insysbio.shinyapps.io/mAb-app/

Stéphanou, A., Fanchon, E., Innominato, P. F., & Ballesta, A. (2018). Systems biology, systems medicine, systems pharmacology: The what and the why. *Acta Biotheoretica*, *66*. https://doi.org/10.1007/s10441-018-9330-2