




# BioGears: A C++ library for whole body physiology simulations

Austin Baird<sup>\*1</sup>, Matthew McDaniel<sup>1</sup>, Steven A. White<sup>1</sup>, Nathan Tatum<sup>1</sup>, and Lucas Marin<sup>1</sup>

<sup>1</sup> Applied Research Associates, Inc. Advanced Modeling and Simulation Systems Directorate

DOI: [10.21105/joss.02645](https://doi.org/10.21105/joss.02645)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Marie E. Rognes](#) 

## Reviewers:

- [@fcooper8472](#)
- [@MiroK](#)

Submitted: 28 August 2020

Published: 09 December 2020

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

## Summary

BioGears is an open source, extensible human physiology computational engine that is designed to enhance medical education, research, and training technologies. BioGears is primarily written in C++ and uses an electric circuit analog to characterize the fluid dynamics of the cardiopulmonary system. As medical training requirements become more complex, there is a need to supplement traditional simulators with physiology simulations. To this end, BioGears provides an extensive number of validated injury models and related interventions that may be applied to the simulated patient. In addition, BioGears compiled libraries may be used for computational medical research to construct *in-silico* clinical trials related to patient treatment and outcomes. Variable patient inputs support diversity and specification in a given application. The engine can be used standalone or integrated with simulators, sensor interfaces, and models of all fidelities. The Library, and all associated projects, are published under the Apache 2.0 license and are made available through the public GitHub repository. BioGears aims to lower the barrier to create complex physiological simulations for a variety of uses and requirements.

Physiological models have been used for healthcare simulation for many years but generally, complex models of the human physiology are not implemented and the patient state is pre-programmed by the instructor. The most prominent commercial implementation of similar software is Maestro, developed by Canadian Aviation Electronics, Inc. This product is proprietary and it is not clear to the authors how models are developed, implemented, and validated. Other similar open source projects include: [pk-sim](#) the pharmacological modeling framework ([Willmann et al., 2003](#)), CellML a generic biological modeling markup language with applications spanning biological applications ([Lloyd et al., 2004](#)), and [Pulse](#) a maintained BioGears fork lacking some of the recent models but instead focusing on Unity VR integration. Each of these either has submodels that are integrated into BioGears (like a pkpd pharmacological model), or is a more generic implementation of biological modeling, like CellML. BioGears is unique in the depth of integrated models, while being free and open-source for the research community.

## Statement of need

The fields of Medical simulation and computational medicine are growing in application diversity and complexity ([Sweet, 2017](#)). Simple CPR manikins are now being replaced with complex robotic systems that can simulate breathing and react to the performance of the trainee. As these systems use cases grow, there is a requirement that they be supplemented

---

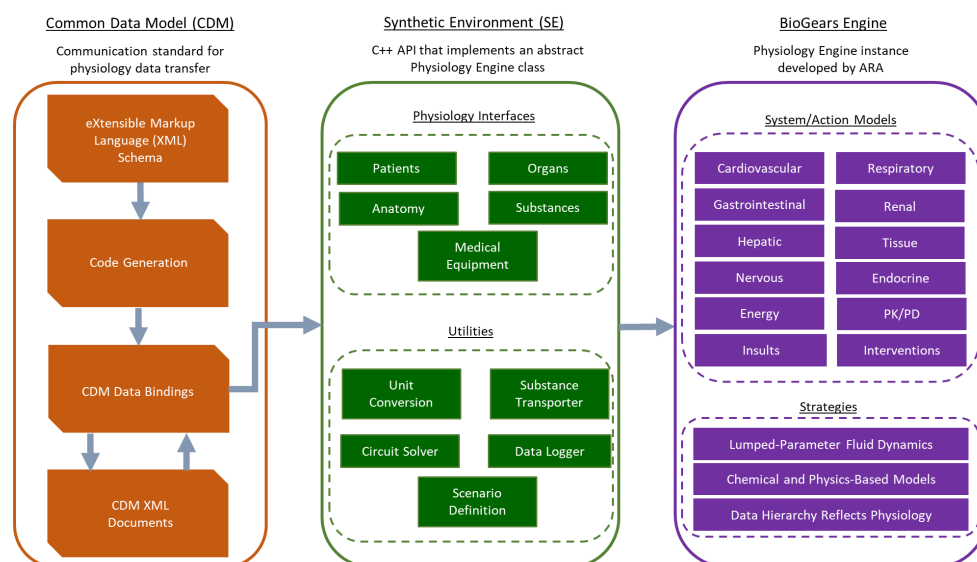
<sup>\*</sup>Corresponding author.

with physiology modeling. BioGears fills this need by providing a free computational framework to use as a backbone to many of these robotic training manikins and may support other computational medicine research applications. The BioGears project aims to better democratize the construction of high-fidelity medical training by providing a sophisticated, complex physiology engine to developers that is easy to integrate and free to use.

BioGears uses a lumped circuit model to describe the circulatory and respiratory systems. This approximation of the simulated cardiopulmonary system has been studied in the past and shown to accurately represent the hemodynamics of the arterial system by using resistance and compliance elements (Otto, 1899; Westerhof et al., 2009). This approximation creates a system that can be solved for rapidly, decreasing the simulation run-time and computational requirements. In addition, BioGears implements models of diffusion and substance transport to properly simulate the gas/blood interface in the lungs. To handle more complex models of physiology, such as pharmacological models, BioGears constructs a set of hierarchical compartments built on top of the circuit analogs. Top-most compartments represents the system level data, such as the liver, with sub-compartments representing more granular biology of the patient such as the nephron, extravascular tissue, and even intracellular spaces. A generic data request framework, leveraging XML, is used to access various substance, fluid, thermal, and gas information for a specific compartment of the body.

The BioGears engine has been used in numerous computational medical research applications including: models of sepsis (McDaniel, Keller, et al., 2019), burn (McDaniel & Baird, 2019), surgical planning (Potter et al., 2017), and pharmacological kinetics and clearance (McDaniel, Carter, et al., 2019). For each application, the patient physiology and traditional interventions used to treat each injury are validated. Full documentation and validation for every action available to the user is provided through our [website](#).

New drugs can be implemented in BioGears by filling in the appropriate physiochemical properties in the provided eXtensible Markup Language (XML) format. The BioGears engine handles computation of clearance, tissue diffusion, and patient responses based on this file and does not require additional C++ programming by the user. The software architecture of BioGears is implemented in three layers of abstraction to encourage easy integration and provide a robust application programming interface (API), see [Figure 1](#).



**Figure 1:** Overview of the BioGears engine software structure. The SE layer provides a generic physiology API and may be leveraged for other physiology engine implementations and/or integration with other computational biology applications.

## Features

The BioGears engine, once compiled, provides a set of libraries that may be included in any application that wishes to leverage a physiological simulation of a patient. In addition, BioGears provides build support and testing for all major user platforms (MacOS, Windows, Linux, and ARM). An instance of a BioGears engine models a single patient's physiology and can be edited at the start of runtime or during the simulation, in the following ways:

- The patient is defined by parameters, such as height, weight, systolic and diastolic pressure.
- Initialize the patient with specific chronic and/or disease states via conditions.
- Modify the patients external environmental conditions (weather, submerge in water, etc.)
- Apply various actions: acute insults/injuries, interventions, conscious breathing, exercise, etc.
- Interact with equipment models, such as an Anesthesia and/or an ECG Machine as well as an Inhaler via the action framework.

Constructing a pointer to an engine, loading a patient, creating data requests (published to a given .csv file), and applying actions to the patient is easy and can be done in only a few lines of code:

```
using namespace biogears;
void HowToFasciculation()
{
    // Create the engine and load the patient
    std::unique_ptr<PhysiologyEngine> bg =
    CreateBioGearsEngine("HowToFasciculation.log");
    bg->GetLogger()->Info("HowToFasciculation");

    if (!bg->LoadState("./states/StandardMale@0s.xml")) {
        bg->GetLogger()->Error("Could not load state, check the error");
        return;
    }

    // Set data requests and create output file
    bg->GetEngineTrack()->GetDataRequestManager().
    CreateLiquidCompartmentDataRequest().
    Set("VenaCava", *Na, "Molarity", AmountPerVolumeUnit::mmol_Per_L);
    bg->GetEngineTrack()->GetDataRequestManager().
    SetResultsFilename("HowToFasciculation.csv");

    // Advance Simulation time by 1 minute
    bg->AdvanceModelTime(1.0, TimeUnit::min);

    // Create an AirwayObstruction action with a severity of .6
    obstruction.GetSeverity().SetValue(0.6);
    bg->ProcessAction(obstruction);

    bg->GetLogger()->Info("Giving the patient an airway obstruction.");
    for( auto i = 0; i < 60*60; --i){
        //Advance time for 1 hour and log EngineTracks to ouput
        bg->AdvanceModelTime(1.0, TimeUnit::s);
        m_Engine.GetEngineTrack()->
```

```
TrackData(m_Engine.GetSimulationTime(TimeUnit::s),m_append_data);
}
```

The Biogears project includes 36 complete howto's which demonstrate various uses of the API and a versatile command line utility for running XML defined Scenarios using the CDM. Additional support can be found by going to <https://biogearsengine.com/>, <https://www.biogears.dev/>, or by joining our IRC [slack channel](#).

## Acknowledgments

This work was made possible by cooperative agreements that were awarded and administered by the US Army Medical Research & Materiel Command (USAMRMC) and the Telemedicine and Advanced Technology Research Center (TATRC), at Fort Detrick, MD, under Contract Number W81XWH-13-2-0068 and W81XWH-17-C-0172. We'd like to acknowledge the support and guidance of Hugh Connacher, Harvey Magee, and Geoff Miller.

## References

- Lloyd, C. M., Halstead, M. D., & Nielsen, P. F. (2004). CellML: Its future, present and past. *Progress in Biophysics and Molecular Biology*, 85(2-3), 433–450. <https://doi.org/10.1016/j.pbiomolbio.2004.01.004>
- McDaniel, M., & Baird, A. (2019). A full-body model of burn pathophysiology and treatment using the BioGears engine. *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 261–264. <https://doi.org/10.1109/EMBC.2019.8857686>
- McDaniel, M., Carter, J., Keller, J. M., White, S. A., & Baird, A. (2019). Open source pharmacokinetic/pharmacodynamic framework: Tutorial on the BioGears engine. *CPT: Pharmacometrics & Systems Pharmacology*, 8(1), 12–25. <https://doi.org/10.1002/psp4.12371>
- McDaniel, M., Keller, J., White, S., & Baird, A. (2019). A whole-body mathematical model of sepsis progression and treatment designed in the BioGears physiology engine. *Frontiers in Physiology*, 10, 1321. <https://doi.org/10.3389/fphys.2019.01321>
- Otto, F. (1899). Die grundform des arteriellen pulses. *Zeitung Fur Biologie*, 37, 483–586. [https://doi.org/10.1016/0022-2828\(90\)91460-o](https://doi.org/10.1016/0022-2828(90)91460-o)
- Potter, L., Arikatla, S., Bray, A., Webb, J., & Enquobahrie, A. (2017). Physiology informed virtual surgical planning: A case study with a virtual airway surgical planner and BioGears. *Medical Imaging 2017: Image-Guided Procedures, Robotic Interventions, and Modeling*, 10135, 101351T. <https://doi.org/10.1117/12.2252510>
- Sweet, R. M. (2017). The CREST simulation development process: Training the next generation. *Journal of Endourology*, 31(S1), S–69. <https://doi.org/10.1089/end.2016.0613>
- Westerhof, N., Lankhaar, J.-W., & Westerhof, B. E. (2009). The arterial windkessel. *Medical & Biological Engineering & Computing*, 47(2), 131–141. <https://doi.org/10.1007/s11517-008-0359-2>
- Willmann, S., Lippert, J., Sevestre, M., Solodenko, J., Fois, F., & Schmitt, W. (2003). PK-sim (r): A physiologically based pharmacokinetic 'whole-body' model. *Biosilico*, 4(1), 121–124.