# FFEA software: Repeated early innovation

openaccessgovernment.org/article/ffea-software-repeated-early-innovation/168737

23 October 2023

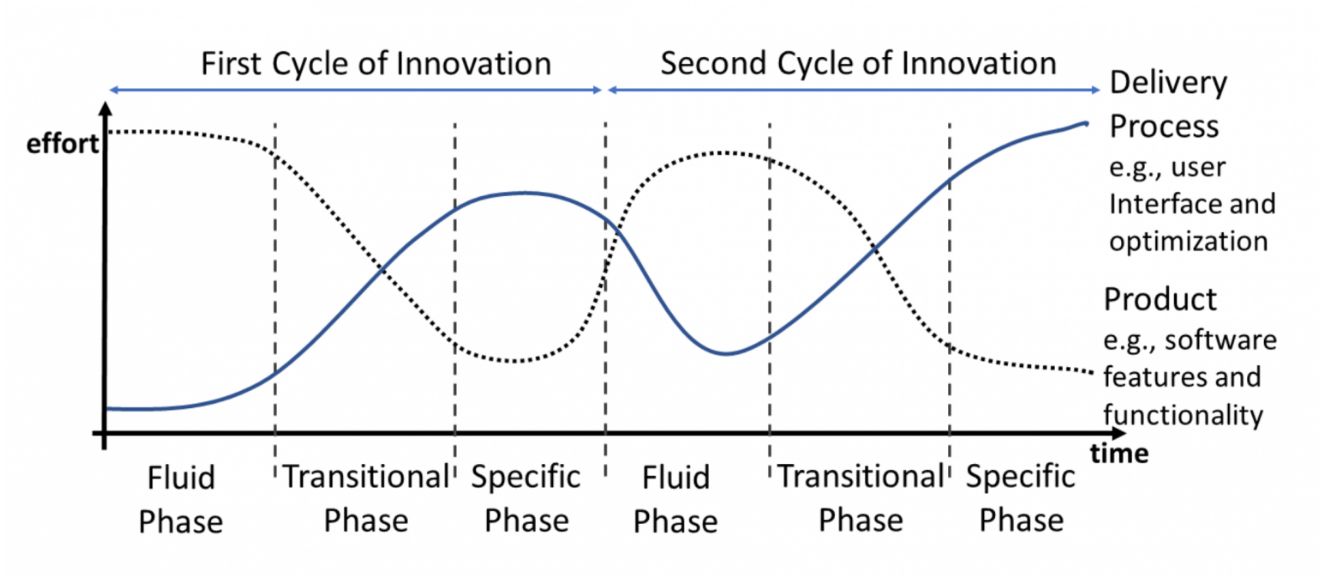


Fig 1: The Abernathy-Utterback curve represents the innovation pipeline in this case as it transitions through the production of two consecutive stable products. It has three phases: 1) the Fluid Phase, where flexibility is needed because of uncertainty in the product idea, the technology and the market and in this case, the needs of the research community; 2) the Transitional Phase, when the technology, the application and the customer's needs are better understood until a 'dominant design' emerges; 3,) the Specific Phase the 'dominant design' shifts from being different to having good performance

## Dr Joanna Leng, Senior Research Software Engineering Fellow at the University of Leeds, discusses software with repeated early innovation using the example of the FFEA software

This article analyses the application of software engineering (SE) tools and practices and research software engineering (RSE) practices to the Fluctuating Finite Element Analysis (FFEA) software, a mesoscale modelling package.

In 2016, I met the FFEA team to discuss improving their software. In 2020, I started working with them to develop a roadmap for the FFEA software by assessing their research interests and software.

### The RSE and repeated innovations

In a previous article, I explained what an RSE is, why the development of that role is vital to the Engineering and Physical Sciences Research Council (EPSRC), and where the PERPL software sat on the Abernathy- Utterback representation of the innovation pipeline.

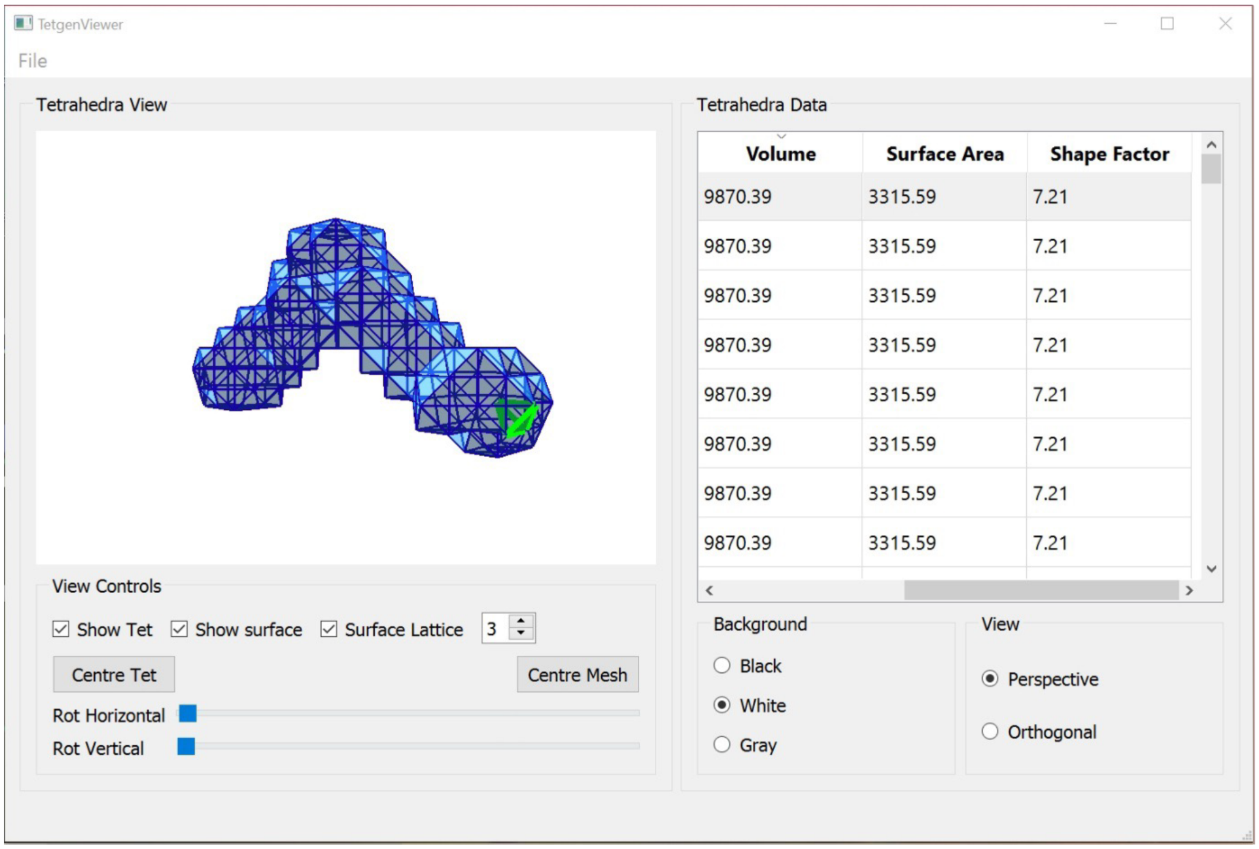
In that case, the pipeline was for a single large innovation, the adoption of super-resolution light microscopy (SRLM); however, the curve makes no distinction between the number or size of innovations, only effort. The pipeline for the FFEA software covers two major software releases. A significant release requires considerable SE effort to make the product stable, while research students mainly add effort for functionality/features.

The software started in 2007 as a PhD project when only scientific results were published. In 2018, after much SE effort (including improving the documentation, adding tests and making it easy to install), the FFEA team published a software paper. This effort is represented in Fig. 1 by the first peak of the blue curve, while the second peak represents my team's effort.

Getting funding at this stage in the pipeline is challenging, particularly for research software, as there is a small user base. There is proof of concept but not yet ease of use, and it must compete with more established innovations/technologies. This is called the Innovation Valley of Death.

All innovations require funding to survive. For academic research software, early investment is in time; academics donate time to explore new research interests and RSE teams as a pump-priming. Funding takes an innovation from proof of concept to profitable new products/innovations the market adopts. The market is a research community for research software, and use/adoption is continual.

The problem is that funding for research software dominates the field of mature software. In academia, funding changes as software moves along the pipeline. The host university funds early innovation software through research student projects or free RSE time as prime-pumping; the costs are added to the university overheads. Research councils fund established software. While in business, there are more ways to fund crossing the Innovation Valley of Death, e.g., venture capital.



| Volume | Surface Area | Shape Factor |
|---|---|---|
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |
| 9870.39 | 3315.59 | 7.21 |

[✓ Show Tet] [✓ Show surface] [✓ Surface Lattice] 3

Background: Black [✓ White] Gray

View: [✓ Perspective] Orthogonal

Fig 2: The new FFEA meshing software showing the visualisation scene on the left displaying a tetrahedral mesh of a myosin molecule with a small element highlighted in green

## The FFEA software

FFEA performs continuum mechanics simulations of proteins and other globular macromolecules. It combines conventional finite element methods with stochastic thermal noise to enable simulations at the mesoscale (length-scales in the range of 5nm to 1μm), where few modelling tools exist. The unknown nature of the mesoscale makes it ripe for exploration and new discoveries.

The C++ core program began in 2007, and Python 2.7 support utilities, FFEAtools, were added in 2010. These were integrated into a single package for the 2018 paper to improve ease of installation. The lead was an EPSRC-funded early career fellow, and while these improvements were critical, they were not an accepted academic output.

In 2020, a combined assessment of software and research interests involving users and developers was performed. While it is recognised that software assessment is an advanced SE skill, combining it with research interests is an unrecognised advanced RSE research method. They wanted backwards compatibility, future flexibility to allow for adaption to their various funding and research interests, and problems to be fixed.

## Improvements to the FFEA software

By 2018, FFEA was a single software package integrating a C++ core and Python 2.7 utilities. It is now a suite of three packages:

- The FFEA core C++ program.
- FFEAtools, a Python 3.9 utility pipy package.
- Ffea_meshing, a Python 3.9 GUI pipy package.

The suite is modular, making it easier to extend.

## Incentives to improve the FFEA software

The FFEA team, including the RSEs, are in faculty, so success is measured in publications. This prioritises improving functionality over stability. The research students have limited time to learn SE/RSE skills. They do not have the advanced SE/RSE experience needed to assess software, re-design and re-engineer in alignment with research needs.

## FFEA software discussion

The FFEA software has unique functionality implementing new physics, which is at the forefront of mesoscale biomolecular simulation globally.

This software could not have been created or improved without research students. However, research students do not have the advanced expertise needed to assess and re-engineer the software; a junior RSE in an anonymised pool does not have the domain knowledge or numerical expertise to be successful; a generalised team of RSE will struggle to supply advanced RSE on an ad hoc basis; there is no funding route so, for now, research students will do their best.

As the software grows and gains complexity with each new addition from a research student, it will need more advanced RSE expertise to maintain and produce stable releases compatible with future research interests. The software is in danger of being lost unless the team can gain access to a permanent advanced RSE who understands the software and the research needs of the group, best achieved if they are part of the research team.

*References can be provided on request.*

**Acknowledgement:**

Thanks to FFEA team (https://ffea.readthedocs.io/en/latest/index.html) based at the University of Leeds. This work was supported by the EPSRC grant EP/R025819/1.

Please Note: This is a Commercial Profile

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

## More About Stakeholder

The School of Computing – Dr Joanna Leng
Dr Joanna Leng discusses her role at the School of Computing at the University of Leeds, with a focus on Research Computing and Imaging.