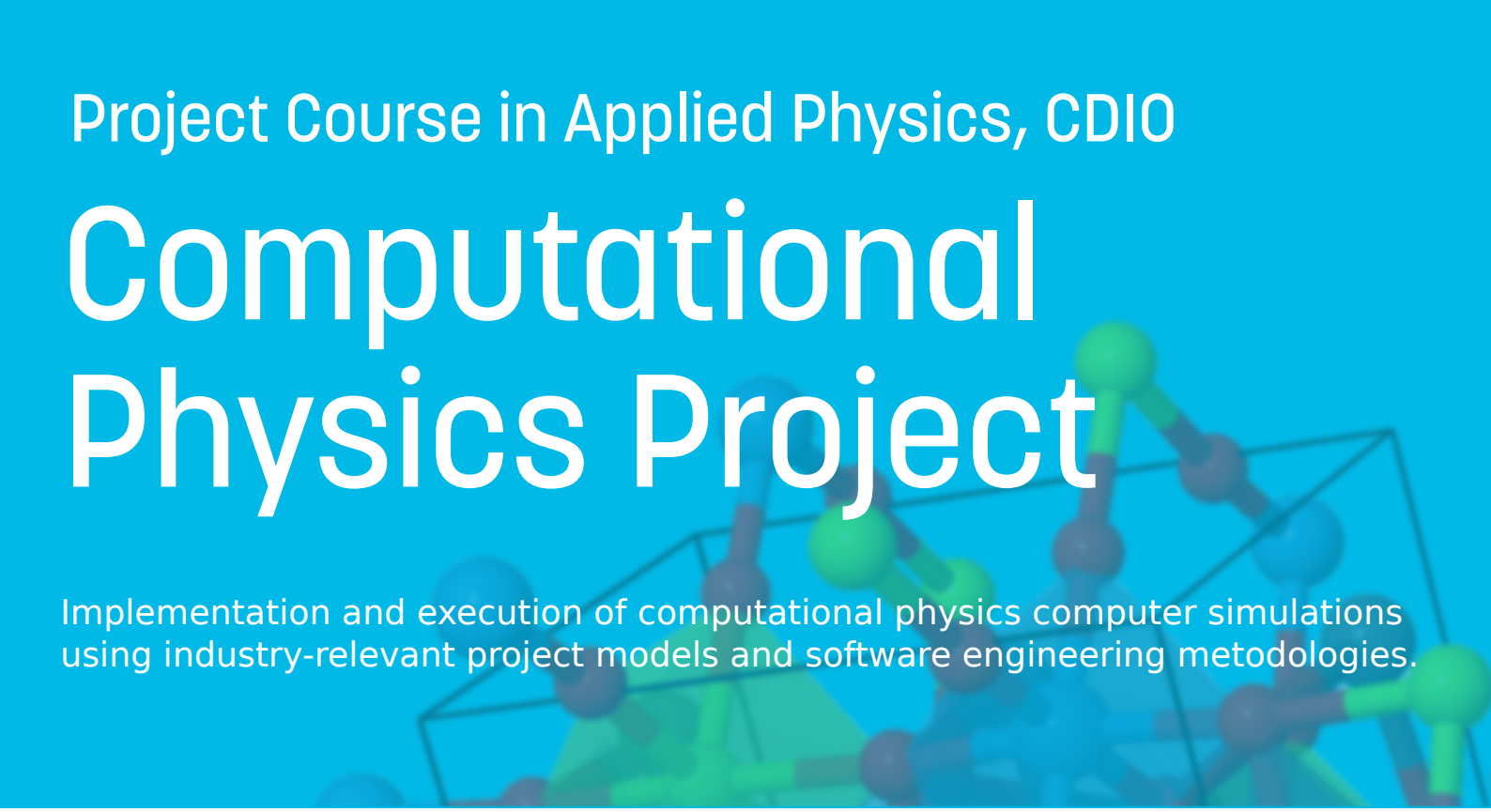
Project Course in Applied Physics, CDIO

Computational Physics Project



Implementation and execution of computational physics computer simulations using industry-relevant project models and software engineering metodologies.

The project at a glance

• Co-develop software for state-of-the-art molecular dynamics simulations, run it in high-throughput on the NSC supercomputers to generate big data, analyze it, visualize it, and make the results available to the world in a database accessible by an open API.

Skills

- Evolve your skills from programming→ software development→ ***software engineering.***
- Get experience working in an industry-relevant ***agile project model.***
- Work on your ***portfolio*** for showing prospective employers.
- Lectures will cover the ***relevant physics***, background theory, analysis, ***visualization,*** + a wide range of ***software-related topics:***

  (e.g., version control, databases, visualization, test-driven development / continuous integration, optimization / profiling, security aspects, and advanced programming topics: patterns / paradigms / concurrency.)
- Train your ***presentation skills*** in written and oral presentations.

Four hands-on-sessions

- Version control with git, collaborative development on GitHub and GitLab.
- Advanced programming techniques and visualization.
- Molecular dynamics with ASE and ASAP.
- Running high-throughput computations on NSC supercomputers.

Contact & Info: **Rickard Armiento**, rickard.armiento@liu.se, Room G307

Project Course in Applied Physics, CDIO

Computational Physics Project

Preliminary Course Outline

Lecture 1
- Introduction: History, background, course plan - Your portfolio, project models: CDIO, LIPs, Agile/Scrum.

Lecture 2
- Project models cont.: Scrum - version control systems, collaborative development (git, svn, GitHub, ...).

Hands-on exercise 1
- git: commits, branches, merging, etc.
- GitHub: collaborative coding via pull-requests and approvals; issues, projects, milestones, labels, releases.

Lecture 3
- The project planning phase - Documentation, software licensing.
- Avanced programming techniques: paradigms, patterns, concurrency, algorithm complexity.
- Visualization.

Lecture 4
- Avanced programming techniques cntd. - Computational physics: basics, solid state physics.

Lecture 5
- Software engineering in industry *(guest lecture)* - Computational Physics: simulations (ASE).

Hands-on exercise 2
- Advanced programming techniques; computer languages and paradigms.
- Visualization in Python (matplotlib, and more.)

Lecture 6
- Test driven development / continuous integration, optimization / profiling, debugging.
- Computational Physics: molecular dynamics simulations (ASAP, and more.)

Lecture 7
- Databases: SQL and noSQL.
- Computational Physics: high-throughput simulations.

Lecture 8
- Parallell comutation (processes, threads, Open-MP, MPI), supercomputers.
- Computational physics: molecular dynamics cont.

Hands-on exercise 3
- Molecular dynamics with ASE and ASAP tutorial.
- Unit tests and continous integration on GitHub (GitHub actions).

Lecture 9
- Project execution phase - Computer security aspects.
- Computational Physics: molecular dynamics cont., more about the project.

Hands-on excerise 4
- NSC Super computer usage, queue scripts, high-throughput computation, etc,
- Running ASAP-simulations on NSC supercomputers.

Lecture 10:
- Computational Physics: final remarks about the project execution and final phases.

Project work: final report due at end of december.

Contact & Info: **Rickard Armiento**, rickard.armiento@liu.se, Room G307