SANPC 2024

MUSES collaboration

Claudia Ratti

University of Houston
Department of Physics

T. Andrew Manning

University of Illinois Urbana-Champaign
National Center for Supercomputing Applications

Motivation

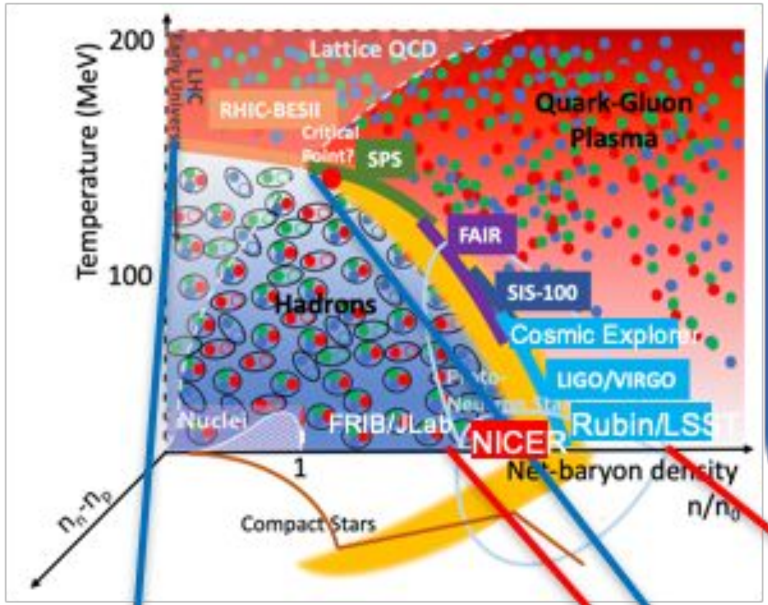
Is there a critical point on
the QCD phase diagram?

What are the degrees of freedom
in the vicinity of
the phase transition?

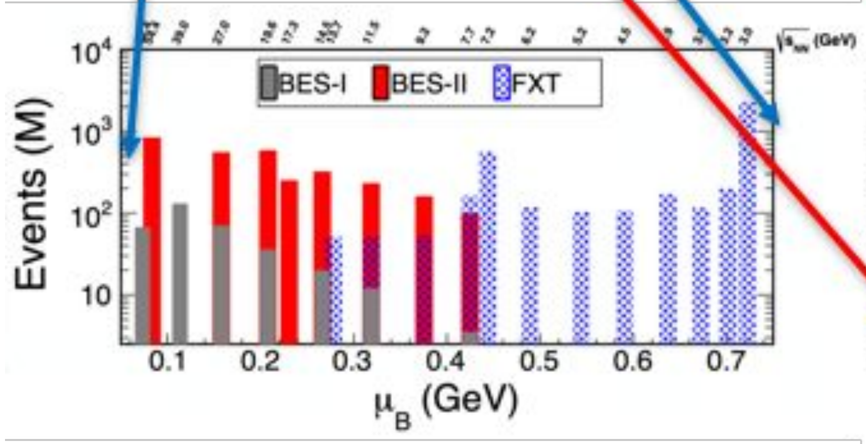
Where is the transition line
at high density?

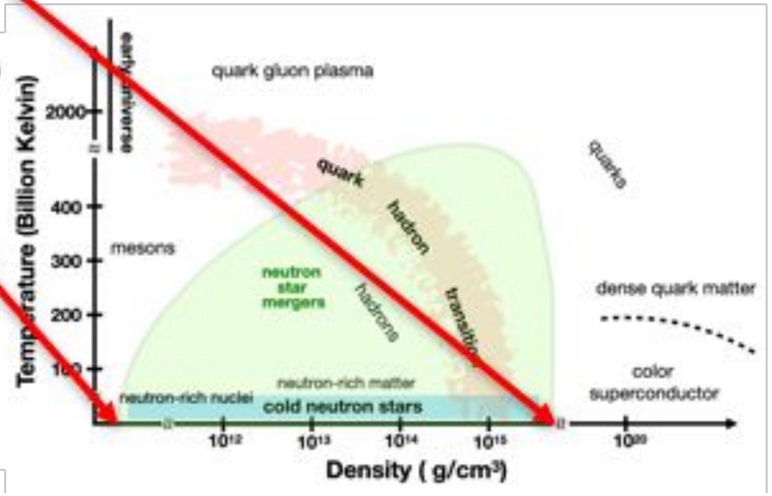
What are the phases of QCD
at high density?

What is the nature of matter
in the core of neutron stars?



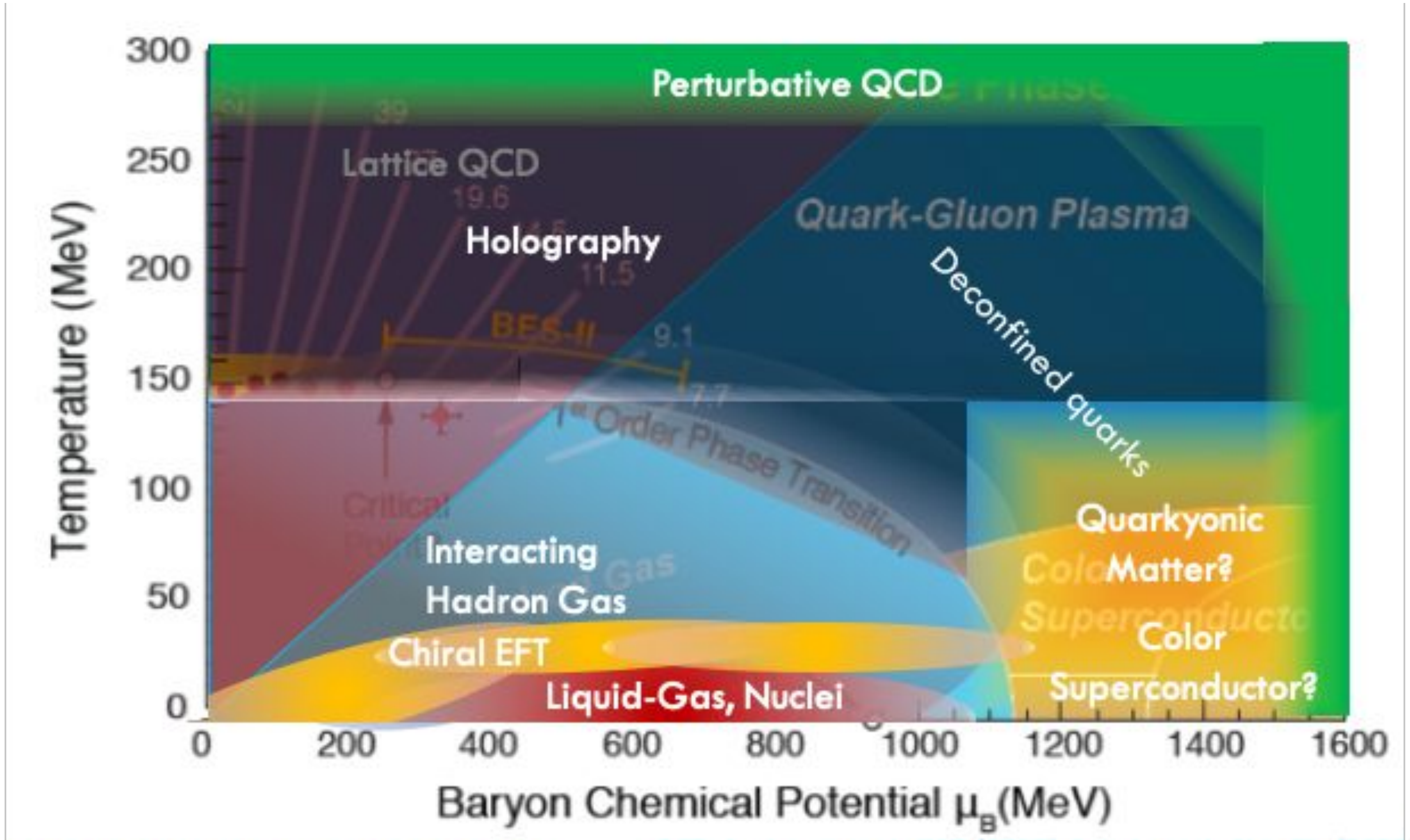
- Run 2019:
  - Collider: $\sqrt{s_{NN}}$=14.6, 19.6, 200 GeV
  - Fixed target: $\sqrt{s_{NN}}$=3.2 GeV
- Run 2020:
  - Collider: $\sqrt{s_{NN}}$=9.2, 11.5 GeV
  - Fixed target: $\sqrt{s_{NN}}$=3.5, 3.9, 4.5, 5.2, 6.2, 7.2, 7.7 GeV
- Run 2021:
  - Collider: $\sqrt{s_{NN}}$=7.7, 17.3 GeV
  - Fixed target: $\sqrt{s_{NN}}$=3.0, 9.2, 11.5, 13.7 GeV





What happens at large densities?

- We need to merge the lattice QCD equation of state with other effective theories
- Careful study of their respective range of validity
- Constrain the parameters to reproduce known limits
- Test different possibilities and validate/exclude them



Lattice QCD: S. Borsanyi, C. R. et al, PRL (2021)
Interacting HRG: V. Vovchenko et al., PRL (2017)
Liquid-gas, Nuclei: see e.g. Du et al. PRC (2019)
Chiral EFT: see e.g. Holt, Kaiser, PRD (2017)
Holography: R. Critelli, C. R. et al., PRD (2017)

pQCD: Andersen et al., PRD (2002); Annala et al., Nat. Ph. (2020)
quarks: C. R. et al., PRD (2006), Dexheimer et al., PRC (2009); Baym et al., Astr. J. (2019)
quarkyonic: McLerran, Pisarski NPA (2007)
CSC: Alford et al., PLB (1998); Rapp et al., PRL (1998), S. Rossner, C. R. et al, PRD (2007), .

MUSES – Modular Unified Solver of the Equation of State

"An open-source cyberinfrastructure fostering a community-driven ecosystem that provides key computational tools to promote, transform and support groundbreaking research in nuclear physics and astrophysics, computational relativistic fluid dynamics, gravitational-wave and computational astrophysics."

- Modular: while at low densities the equation of state is known from 1^{st} principles, at high μ_B we will implement different models ("modules") that the user will be able to pick

- Unified: the different modules will be smoothly merged together to ensure maximal coverage of the phase diagram, while respecting established limiting cases (lattice, perturbative QCD, ChEFT…)

First MUSES-wide publication

muses

Theoretical and Experimental Constraints for the Equation of State of Dense and Hot Matter

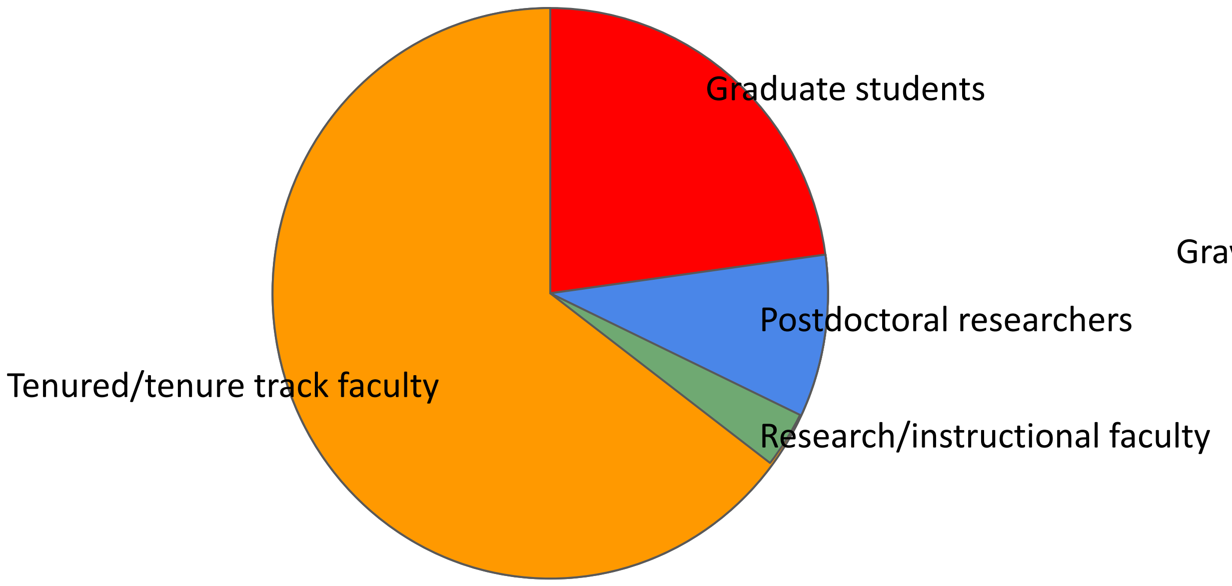
Rajesh Kumar,[1,*] Veronica Dexheimer,[1,†] Johannes Jahan,[2] Jorge Noronha,[3] Jacquelyn Noronha-Hostler,[3] Claudia Ratti,[2] Nico Yunes,[3] Angel Rodrigo Nava Acuna,[2] Mark Alford,[4] Mahmudul Hasan Anik,[5] Katerina Chatziioannou,[6,7] Hsin-Yu Chen,[8,9] Alexander Clevinger,[1] Carlos Conde,[3] Nikolas Cruz Camacho,[3] Travis Dore,[10] Christian Drischler,[11] Hannah Elfner,[12] Reed Essick,[13] David Friedenberg,[14] Suprovo Ghosh,[15] Joaquin Grefa,[2] Roland Haas,[3] Jan Hammelmann,[16] Steven Harris,[17] Carl-Johan Haster,[18,19] Tetsuo Hatsuda,[20] Mauricio Hippert,[3] Renan Hirayama,[16] Jeremy W. Holt,[14] Micheal Kahangirwe,[2] Jamie Karthein,[21] Toru Kojo,[22] Philippe Landry,[23] Zidu Lin,[5] Matthew Luzum,[24] T. Andrew Manning,[3] Jordi Salinas San Martin,[3] Cole Miller,[25] Elias Roland Most,[26,27,28] Debora Mroczek,[3] Azwinndini Muronga,[29] Nicolas Patino,[3] Jeffrey Peterson,[1] Christopher Plumberg,[30] Damien Price,[2] Constanca Providencia,[31] Romulo Rougemont,[32] Satyajit Roy,[5] Hitansh Shah,[2] Stuart Shapiro,[3] Andrew W. Steiner,[5,33] Michael Strickland,[1] Hung Tan,[3] Hajime Togashi,[22] Israel Portillo Vazquez,[2] Pengsheng Wen,[14] and Ziyuan Zhang[4]

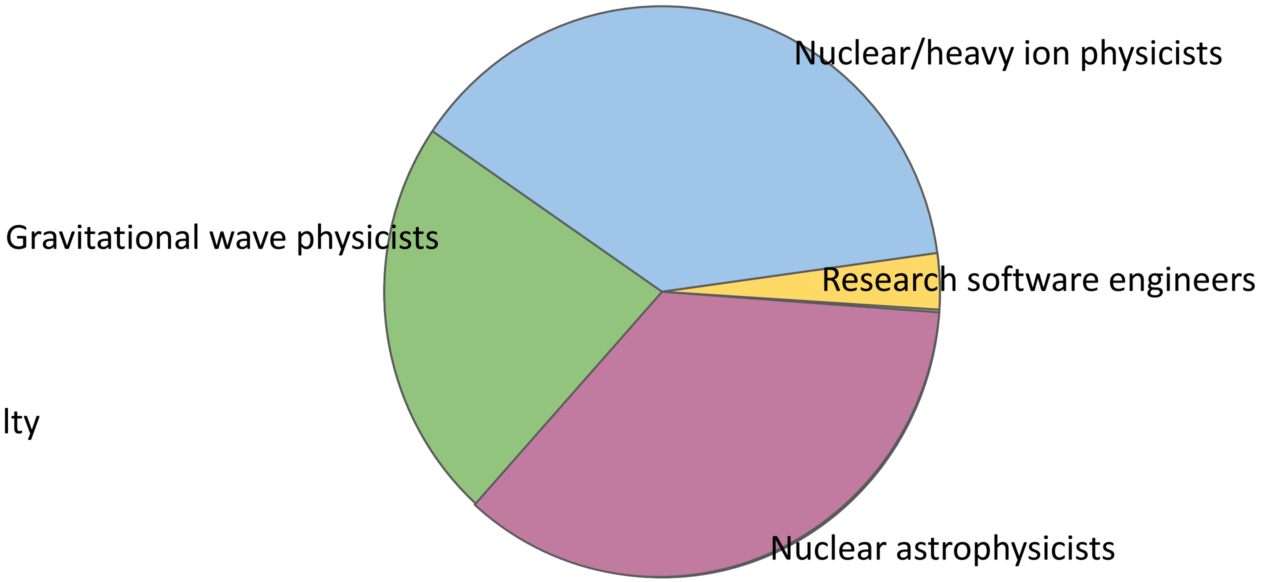
(MUSES Collaboration)

Living Reviews in Relativity 27 (2024)

MUSES – Collaboration at a glance

Collaborators: - 25 at the moment of application
- 85 today!



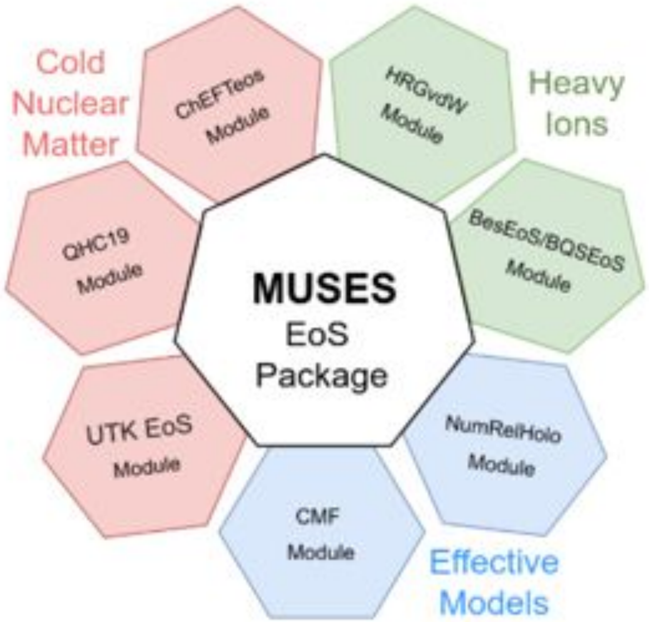


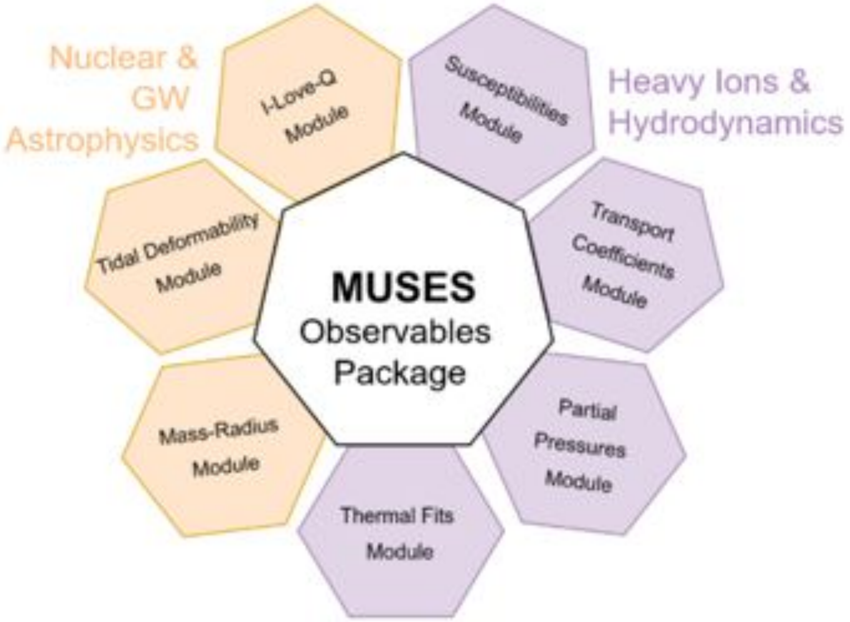
About 25% of collaboration is contributing towards software development at different levels

MUSES – Goals

Cyberinfrastructure of interoperating tools and services within a replicable and flexible deployment system:

- Create both **new calculation modules and upgrade existing libraries** to modern programming languages, improve algorithms, and adopt standards conducive to interoperability.
- Build **web-based tools and services** that provide interactive interfaces to the Calculation Engine, our **job management system** that executes processing workflows requested by researchers.
- Design a scalable, high-availability **deployment system** that can be reproduced in other computing environments.





MUSES Framework & Calculation Engine (CE)

We are defining a software **framework** in which scientists can develop and contribute scientific calculation **modules** that generate and transform data in composable **processing workflows**.

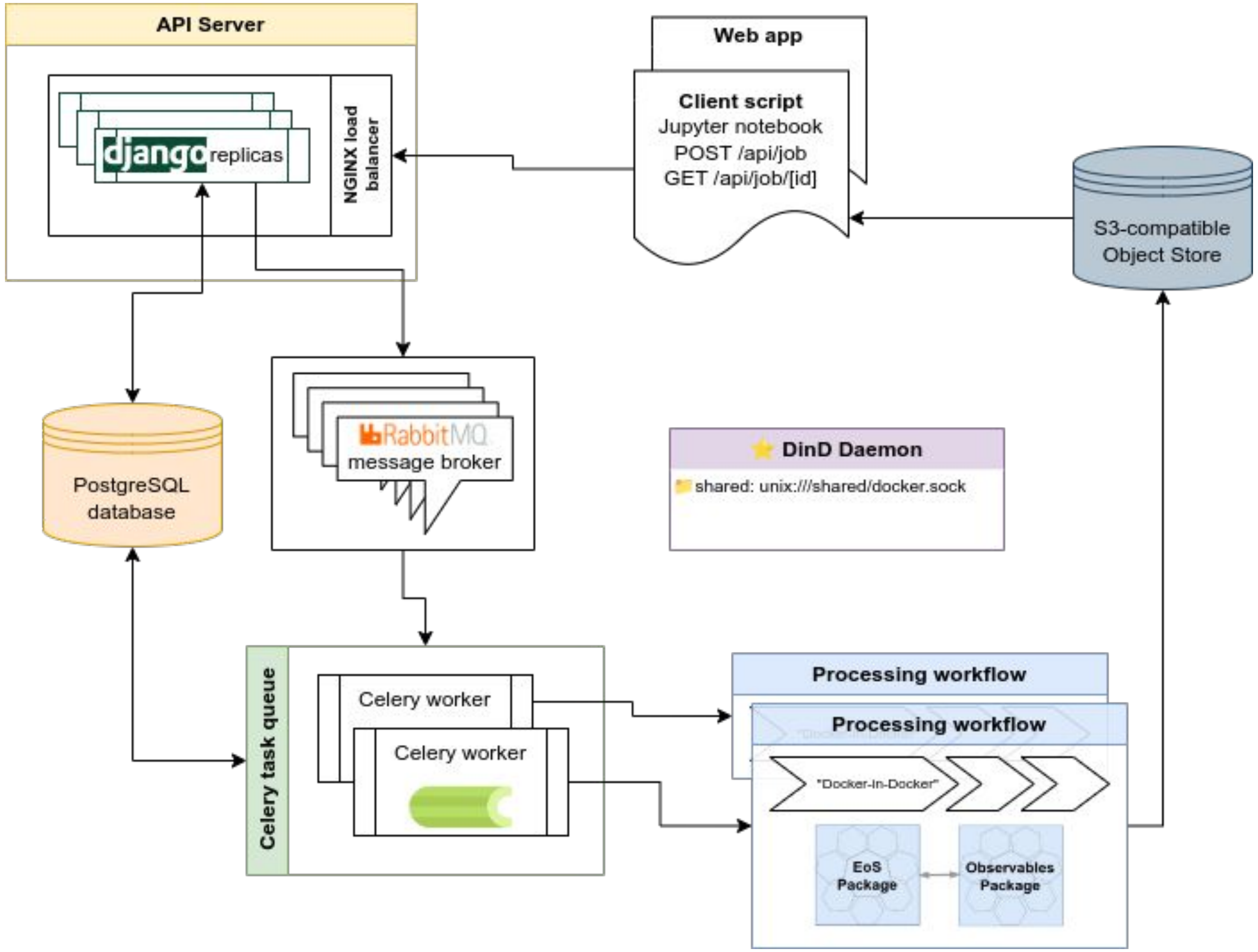
We are also building and hosting a scalable **web application** called the Calculation Engine that executes these workflows.

- **Containerization**
  - Modules must build and push container images that the CE can download and execute via Docker.
- **Manifest**
  - Each module declares input and output file paths, image URLs, and other information required by the CE in a standard format.
- **API specification**
  - The data structure schema for inputs and outputs must be declared in machine-level detail using the OpenAPI standard format.

What is the Calculation Engine?

The Calculation Engine is a software application that

- manages user-submitted **jobs**; a job is the execution of one or more modules in a **data processing workflow**.
- manages the **movement of data** between subsystems
- serves data for download
- enforces **access control**
- organizes and stores information in a structured database
- tracks **data provenance**

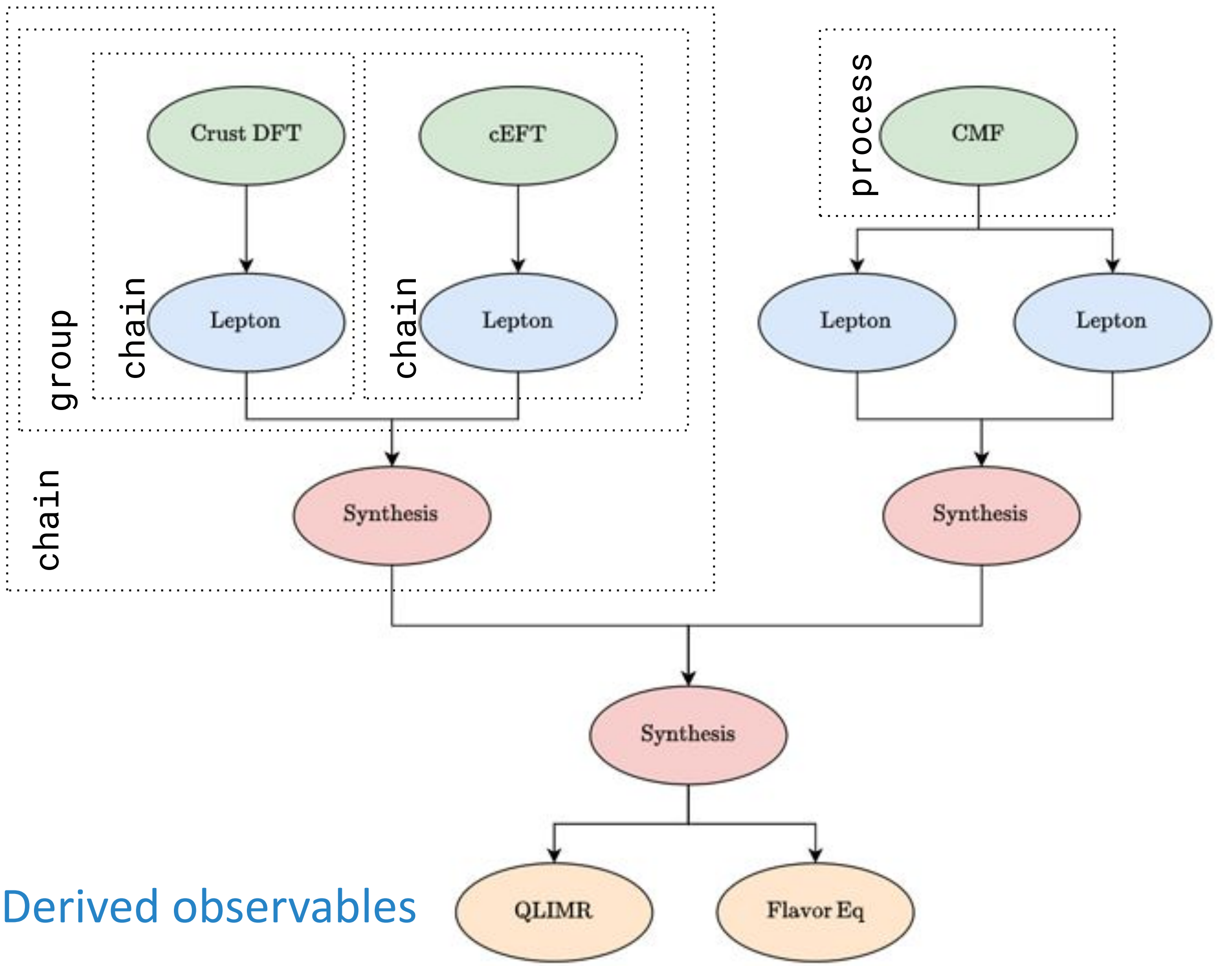


System architecture schematic

Workflows

Data processing workflows are composable structures of arbitrary complexity that are built from a few primitive component types.

Equation of state generators



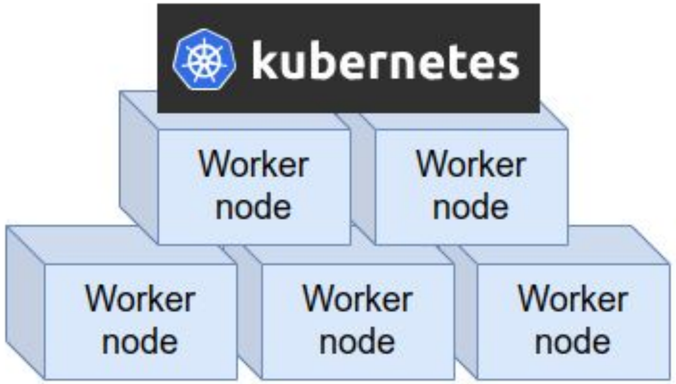
Derived observables

Workflow composed of hierarchically defined components

Deployment platforms

To facilitate rapid code interation, the “alpha release” of our Calculation Engine is deployed via **Docker Compose** on a single VM running on **Jetstream2** at Indiana University, free through the **NSF ACCESS program**.

Once stable, we will migrate to a **Kubernetes**-based deployment on our cluster at NCSA for **scalability** and **resilience**. We plan to integrate additional HPC clusters like NCSA’s Delta as optional execution targets.



Build software communities

muses

- Context
  - Short-term grants fund academic groups to develop software that **functions as the scientific instrument to conduct their research**.
- Goals
  - Reduce wasteful redundant effort and “software decay”.
  - Produce higher quality software products for better science.
- Strategies
  - Make software **free and open source** (FOSS). Use FOSS licenses. Host code publicly.
  - Build upon **existing software ecosystems**. Establish or use standard file formats for common data structures. Collaboratively develop and publish libraries that process these formats.
  - **Publish packages** on multiple package repos for discoverability & easy installation (e.g. PyPI, npm).
  - Write **documentation** targeting both “end-user” researchers as well as developers.
  - Leverage available **code collaboration-ware**: integrated tools for bugs/issues, wiki, CI/CD for transparent reproducibility.
  - Choose **decentralized solutions** where possible to support continuity and freedom for communities.

Embrace portable cyberinfrastructure (CI)

muses

- Context
  - Short-term grants include **funding for RSEs** with the expertise to design, deploy, operate, and maintain the **CI underpinning services** hosted for the target research community.
- Goals
  - Reduce wasteful redundant effort related to designing a deployment system, provisioning machines, installing dependencies, and operating services. Use force multiplying techniques.
  - Lower barriers to migration of services and data between institutions and hosting providers.
- Strategies
  - Follow the **GitOps paradigm** and **Infrastructure-as-Code** patterns, specifying CI declaratively and using industry-standard FOSS solutions to bootstrap and provision computing resources (e.g. Terraform, Helm, ArgoCD)
  - Design deployment systems that are **reusable and composable**.
  - Use FOSS solutions and platforms **based on open standards** throughout the stack: OpenStack, Docker, Kubernetes, MariaDB/PostgreSQL/Cassandra/MongoDB.
  - Choose software architectures **conducive to migration**: Keycloak in front of identity providers, S3-compatible object storage instead of filesystems

Long-term data storage

muses

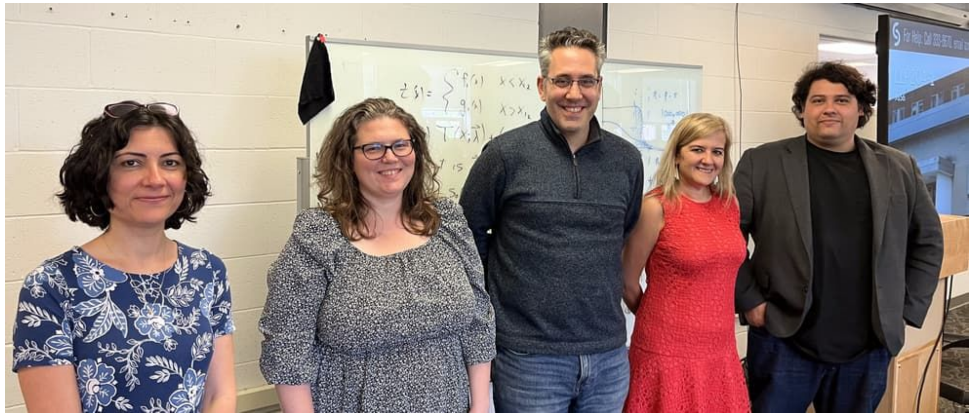
- Context
  - The use of advanced computing will generate increasing amounts of **data products that will be scientifically useful** beyond the scope of funded projects. Current funding models do not support long-term storage.
- Goals
  - Follow the FAIR principles: Findability, Accessibility, Interoperability, and Reuse
  - Maintain the **availability** of published data as long as possible
  - Ensure that data objects have **persistent URLs** that can survive beyond the funding period.
- Strategies
  - Leverage **existing data repositories** where feasible (Zenodo, Data Dryad, Illinois Data Bank, etc).
  - Design around **economical storage options** like S3-compatible object storage where possible
  - **Construct "nomadic" URLs** to data objects. Instead of using the domain of an institution in the funded phase of a project, select a domain name that can be transfered to a new custodian such that data can be migrated transparently to researchers.
  - **Start investing** in truly decentralized, peer-to-peer storage networks like IPFS.

Workforce development

muses

- Context
  - **Research Software Engineers (RSEs) are critical partners** in advancing science that requires advanced computing. They are professionals with expertise beyond what a physics grad student or postdoc has time and motivation to acquire.
- Goals
  - **Integrate** RSEs into funded research projects.
  - Encourage the **growth** of the RSE professional field.
- Strategies
  - **Identify students and postdocs who exhibit interest** in Research Software Engineering and pair them with RSEs participating in their research projects so that they can learn about **alternative career paths**.
  - **Include grad students and postdocs** from departments like computer science **outside of physics**, whose research is aligned the project and who can focus on the software and CI development.

Connect with us!



https://musesframework.io/connect

Extra slides

MUSES – Participants

muses

PI and co-PIs

1. Nicolas Yunes; University of Illinois at Urbana-Champaign; **PI**
2. Jacquelyn Noronha-Hostler; University of Illinois at Urbana-Champaign; co-PI
3. Jorge Noronha; University of Illinois at Urbana-Champaign; co-PI
4. Claudia Ratti; University of Houston; co-PI and **spokesperson**
5. Veronica Dexheimer; Kent State University; co-PI

Senior investigators

1. Roland Haas; National Center for Supercomputing Applications
2. Timothy Andrew Manning; National Center for Supercomputing Applications
3. Andrew Steiner; University of Tennessee, Knoxville
4. Jeremy Holt; Texas A&M University
5. Gordon Baym; University of Illinois at Urbana-Champaign
6. Mark Alford; Washington University in Saint Louis
7. Elias Most; Princeton University

Current and past postdoctoral researchers

1. Mauricio Hippert; University of Illinois at Urbana-Champaign
2. Johannes Jahan; University of Houston
3. Mateus Reinke Pelicer; Kent State University
4. Reed Essick; Perimeter Institute
5. Rajesh Kumar, KSU
6. Alex Haber, Wash U Saint Louis
7. Zidu Lin, UT Knoxville
8. Agnieszka Sorensen, University of Washington

External collaborators

1. Helvi Witek; UIUC
2. Stuart Shapiro; UIUC
3. Katerina Chatziioannou; Caltech
4. Phillip Landry; California State University Fullerton
5. Volodymyr Vovchenko, University of Houston
6. Rene Bellwied; University of Houston
7. David Curtin; University of Toronto
8. Michael Strickland; Kent State University
9. Matthew Luzum; University of Sao Paulo
10. Hajime Togashi; Kyushu University
11. Toru Kojo; Central China Normal University
12. Hannah Elfner; GSI/Goethe University Frankfurt
13. Debarati Chatterjee, IUCAA India
14. Hsin-Yu Chen, MIT
15. Tetsuo Hatsuda, RIKEN
16. Cole Miller, University of Mariland
17. Israel Portillo, University of Houston

https://musesframework.io/connect

Current and past graduate students

1. Mahmudul Hasan Anik, UT Knoxville
2. Suprovo Ghosh, IUCAA India
3. Alexander Clevinger, KSU
4. Nikolas Cruz Camacho, UIUC
5. Joaquin Grefa, University of Houston
6. Jiaxi Wu, Caltech
7. Jamie Karthein, University of Houston
8. Micheal Kahangirwe, University of Houston
9. Angel Nava, University of Houston
10. Hung Tan, UIUC
11. Pengsheng Wen, Texas A&M University
12. Ziyuan Zhang, WUSL
13. Carlos Conde, UIUC
14. Yumu Yang, UIUC
15. Jordi Salinas San Martin, UIUC
16. David Friedenberg, Texas A&M University
17. Hitansh Shah, University of Houston
18. Satyajit Roy, UT Knoxville
19. Ahmed Abuali, University of Houston
20. Prachi Garella, University of Houston
21. Musa Rahim Khan, University of Houston

Sustainability concerns

<table>
<tr><th>Challenge</th><th>Potential solution / stopgap</th></tr>
<tr><td>Computing clusters cost money; the virtualized resources we are using for the MUSES Calculation Engine service that we are operating for the community will be destroyed when funding ends.</td><td>NSF ACCESS is an amazing new program that may provide sufficient resources at no cost for a relatively short period of time depending on the scale of usage.</td></tr>
<tr><td>Without funded personnel, software development is at the mercy of typically uncoordinated, independent groups who have very specific motivations.</td><td rowspan="2">As free and open source software products, the CE and contributed modules should strive to foster user/developer communities that will extend the longevity of the software and ideally improve its utility.</td></tr>
<tr><td>Our containerized approach insulates us against a certain level of “bit rot” and “dependency hell”, yet all software must be updated and maintained over time due to the complex interdependencies between continually evolving software components, including the operating system and runtime environment (e.g. Docker).</td></tr>
</table>

Sustainability concerns

muses

Challenge	Potential solution / stopgap
Cluster nodes require continuous system administration to update the OS with security patches, troubleshoot operational issues like unresponsive machines, monitor data storage utilization, etc. If there remains a hosted service, someone must respond to user help requests and monitor the system for fair use.	Write thorough documentation and train a dedicated team of graduate students & postdocs to handle routine maintenance and basic troubleshooting, relying on robust GitOps methodology to repair and restore broken cyberinfrastructure. Establish a consulting relationship with a Research Software Engineering group who can help occasionally with more complex troubleshooting.
Amount of time it takes to train new students in programming languages and techniques required for the physics research can be very costly.	Leverage existing collaborations or institutions who offer resources like experienced mentors and tutorial/training sessions.