2021 Computational Infrastructure for Geodynamics Developers Workshop

Brad T. Aagaard, U.S. Geological Survey
Jed Brown, University of Colorado, Boulder
Catherine M. Cooper, Washington State University
Rene Gassmoeller, University of Florida
Lorraine Hwang, University of California, Davis
Marc Spiegelman, Lamont-Doherty Earth Observatory, Columbia University

The Computational Infrastructure for Geodynamics (CIG) held its first Developers Workshop February 23 and 25, 2021. The workshop themes included:

- Expanding the CIG software developer community;
- Making CIG software more accessible to new users;
- Identifying and leveraging common infrastructure; and
- Leveraging collective wisdom to make CIG software better and easier to develop and maintain.

The 43 workshop participants represented developers of CIG community codes, those interested in contributing to the community codes, and researchers interested in improving their own codes. The workshop was held online and consisted of six discussion sessions with short presentations introducing each set of topics. The discussions were driven by responses from a detailed developer survey and a shorter user survey. Thirteen developer teams associated with 15 different computer modeling codes submitted responses; this provided information from the developers of all the major CIG-related modeling codes as well as a few other codes. Thirty-one users submitted responses; this represents a small fraction of the total users of the codes (we obtain a much better response rate when surveys are associated with user-related workshops).

Most of the developer teams indicated that there is a much greater demand for developers to implement and test new features and update documentation than they have time for. Users indicated that troubleshooting simulation setup was the most time-consuming step and the most challenging step in their modeling workflow; this demonstrates the need for both good documentation and error messages that help users resolve problems. The workshop discussions were designed to address these challenges. It was clear throughout the workshop that there is much to be gained by fostering a community of developers that share ideas and collaborate.

Expanding the CIG software developer community

The discussion centered on how to increase the number of developers to meet the demand for new features and documentation and how to properly acknowledge contributions from the community. A developer triangle (or developer pyramid) is often used to describe distribution of users and developers and the progression along the way. New users (usually the largest group) form the base of the triangle, with experienced users above them, followed by contributors, developers, and maintainers, the smallest group, at the top (Figure 1). For simplicity we describe the transitions as discrete steps, but, in reality, the transitions are more gradual. For new users, good support through documentation, cookbook examples, tutorials, and responding to questions are critical to maintaining a strong user base and facilitating users climbing the ladder to become experienced users. Hackathons provide experienced users exposure to

making small contributions and becoming familiar with the source code and software development workflow. Developer documentation, clean plug-and-play interfaces, encouragement, and mentoring are all important for obtaining useful contributions. Progressing from contributor to developer or from developer to maintainer usually involves extended duration and close collaboration with other developers and maintainers. The primary takeaway from the discussion is that support for all levels of users is critical for developing a pipeline of users that generates a sufficient number of developers and maintainers. Postdoctoral programs have a particularly important role in this process, because they provide close collaboration over 1-2 years at the stage when users have developed sufficient expertise as graduate students to transition from contributors to developers.

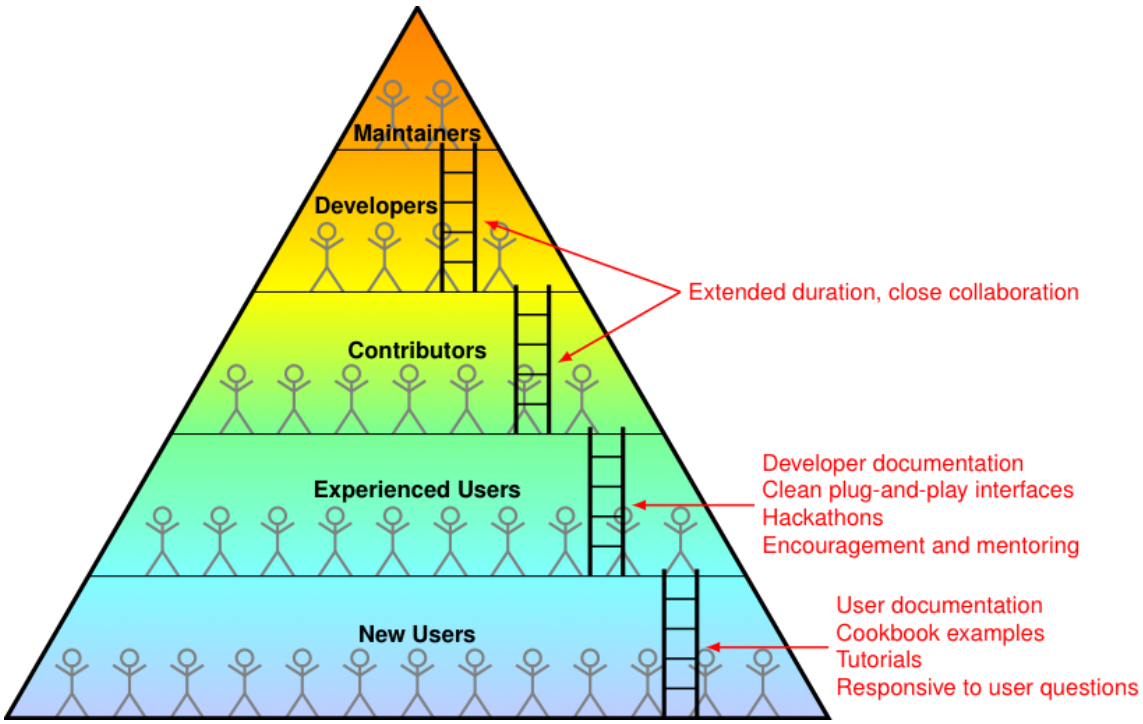


Figure 1: Developer triangle (sometimes called a developer pyramid) showing the progression from new users to maintainers along with key conditions for people to climb the ladder to the next level.

The workshop also included significant discussion on how to best recognize contributions to community software to help scientists advance in their careers. Students and postdocs gain exposure when they are listed as contributors in release notes, software manuals, and community forums or mailing lists; this helps them obtain faculty and research positions. Demonstrating the impact of contributions becomes more important in faculty and research scientist advancement. Coauthorship on journal articles is the most direct way to demonstrate impact, but in many cases, collaboration may be insufficient to warrant coauthorship. To better address these cases, developer teams need to make it easy for users to cite contributions associated with features they use. Likewise, promotion committees in academia, research organizations, and industry need to reward significant contributions to community codes.

Some additional actions that CIG and related modeling communities can take to facilitate a strong suite of users include

- Maintain a catalog of recommended building blocks, e.g., libraries and packages, for potential use in developing new codes;
- Maintain a list of teaching, coding, and numerical modeling resources;

- Forge partnerships with organizations running software carpentry workshops and helping develop ones with a focus on geodynamics; and
- Provide Research Experiences to Undergraduates (REUs) in geodynamics modeling with an emphasis on increasing participation from underrepresented minority groups.

Identifying and leveraging common software infrastructure

One of the workshop objectives was to identify the potential for developing common infrastructure to reduce duplicative development efforts. We focused on the potential for standardizing output and developing a common application programming interface (API) for specifying values for material properties, initial conditions, or boundary conditions. Adopting standards for these two areas would greatly facilitate connecting processes at different scales modeled by different codes.

One session focused on standardizing simulation output. The responses to the developers survey indicated that most of the modeling codes output similar information using similar standard formats. The consensus from the discussion was that CIG should

- Form a working group to develop standard layouts for VTK, HDF5+Xdmf, and netCDF files;
- Leverage standards that have already been developed by other organizations where possible;
- Consider providing web interfaces for inspection, simple processing, and simple visualization; and
- Identify possible common post-processing algorithms and scripts.

The discussion also covered challenges associated with storing results from higher order discretizations, which are rarely supported in visualization tools, and reducing storage requirements of output from massively parallel simulations using compression, including lossy compression. The push towards exascale computing has generated several projects, such as Alpine/ZFP and Conduit, that address these issues and are worthy of investigation to assess their utility.

A second session focused on the feasibility of adopting a standard interface for specifying values for material properties, initial conditions, or boundary conditions. This session included presentations on three existing libraries within the CIG community. World Builder, used by ASPECT, targets specification of material properties by defining volumes within a domain using simple spatial features and assigning parameterizations and properties to the features. SpatialData, used by PyLith, provides an API for querying values in space along with several implementations; it supports georeferencing and unit conversion and is used for specifying values for material properties, initial conditions, and boundary conditions. The library easi, used by SeisSol, provides an API for mapping values from R^n to R^m; it supports several model types, which can be composed together to provide values for material properties and initial conditions. SpatialData and easi have similar APIs and functionality and would not be difficult to merge.

Developing a standard API for specifying material property values, initial conditions, or boundary conditions that could be used across most CIG modeling codes may be feasible, but it will be much more difficult than standardizing simulation output. Specifying values for initial conditions or boundary conditions often only involves simple point queries. Specifying values for material properties is often much more complicated and involves additional information, such as composition and state variables. The discussion suggested that development of an API and implementations would likely involve multiple stages, with use of common infrastructure for a few codes as a first step. The recommendation is for CIG to form a working group to assess use cases, scope, and outcomes of an API and a corresponding library.

Making CIG software more accessible to new users

The user survey showed that many CIG codes are used within a larger modeling workflow or are run many times for sensitivity analyses, uncertainty quantification, or inversions. Additionally, most users struggle with troubleshooting simulation setup, preparing simulation inputs, and post-processing results. Most of the discussion dovetailed from a presentation on the Pangeo project that illustrated the use of Jupyter notebooks and related software for integrating data, modeling, and visualization into computational narratives (interactive description of the modeling setup, simulation, and results). About one-third of the session participants indicated that they currently use Jupyter notebooks. There was broad interest in forming a working group to assess how to best use Jupyter notebooks to improve modeling workflows across all levels of users within the CIG community.

Leveraging collective wisdom to make CIG software better and easier to develop and maintain

Two sessions focused on this theme of improving developer productivity and efficiency. The first session was devoted to developer tools and the software development process. The primarily takeaways from the discussion included the following:

- CIG should encourage use of integrated development environments (IDEs), including providing resources showing people how to set up an IDE and CIG should use IDEs in tutorials and hackathons;
- CIG should provide test runners and consider using a repository-based continuous integration tool, such as GitHub Actions or GitLab Pipelines;
- Singularity containers are likely the best path forward for helping people get CIG modeling codes installed on clusters, which requires parallel-processing libraries configured for the cluster hardware;
- An intermediate step to Singularity containers would be to make available existing Docker containers used in continuous integration tests;
- Spack is another method of installing code that might be useful for resolving difficult dependencies;
- There are ways to make binaries that provide optimized code for multiple architectures. CIG should consider looking into the feasibility of doing this; and
- CIG should encourage codes to provide online documentation, which has several advantages over PDF files, in addition to a PDF version.

A second session focused on best practices for software development. The survey results showed that most of the developer teams follow many of the CIG Software Development Best Practices. As a result, the discussion focused largely on two areas where current practices could be improved.

Most of the developer teams who responded to the survey do not provide the community with a regularly updated document that outlines priorities for software development with a rough timeline of when features might be implemented. However, several developer teams do use GitHub Issues to track feature requests. In fact, GitHub Issues tagged with milestones and GitHub Projects provide users with a more complete picture of the status of new features, because they often include a discussion of design decisions. The discussion identified the importance of developer teams engaging the community at least once a year to update development priorities. In an ideal world, following community engagement developer teams would provide both a developer plan document and track progress using tools, such as GitHub Issues and Projects.

Checkpointing, which allows a simulation to restart from a previous state, was the second best practice that received considerable discussion. Although most codes implement checkpointing, they use their own implementations and they could be more efficient. The conclusion was that several general checkpointing tools are available that CIG should investigate to determine if they are useful for CIG modeling codes.

Summary

The CIG Developers Workshop resulted in a number of recommendations that we think will help expand the CIG developer community, make software more accessible to new users, and increase developer productivity through use of common infrastructure and best practices for software development. This includes building a broad user base with sufficient support through documentation, tutorials, user forums, hackathons, scientific workshops, and mentoring to maintain a healthy suite of software developers and maintainers. Communities also need to offer opportunities, like this workshop, for developer teams to interact with each other to exchange ideas, identify common infrastructure, and interact with users to discuss modeling workflows and development priorities.

2021 CIG Developers Workshop webpage, highlights and links on the CIG Community Forum.

Disclaimer

Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

Agenda

All times are Pacific Time (UTC-8)

Day 1: Tuesday February 23

8:00-8:15 Opening Remarks and Workshop Objectives

SESSION I: DEVELOPER TOOLS

8:15-8:30 Results of Developer Survey: Developer Tools, Presenter: Brad Aagaard

8:30-9:30 Discussion topics:

- What tools/techniques can we use to improve our software development workflow?
- What can CIG do to help the software development process?

Moderators: Jed Brown and Rene Gassmoeller, Scribe: Juliane Dannberg

9:30-10:00 Break

SESSION II: CIG SOFTWARE DEVELOPMENT BEST PRACTICES

10:00-10:05 Results of Developer Survey: Best Practices, Presenter: Brad Aagaard

- What best practices do most codes follow?
- Which best practices do most codes not follow?
- Which best practices do most codes want to work towards and/or need help with?

10:05-11:00 Discussion topics:

- Which best practices need to be updated?
- Are there additional best practices we need to consider?
- Which best practices need more detail (e.g., examples/templates)?

Moderators: Rene Gassmoeller and Timo Heister, Scribe: Lorraine Hwang

SESSION III: SHOULD CIG SPECIFY A STANDARD OUTPUT FORMAT?

11:00- 11:10 Results of Developer Survey: Current output formats, Presenter: Brad Aagaard

11:10-12:00 Discussion topics:

- Is it desirable for CIG to specify/adopt a standard output format?
- How feasible is it for CIG to adopt a standard output format?
- Are there alternatives to a standard output format that provide similar outcomes?

Moderators: Wolfgang Bangerth and Brad Aagaard, Scribe: Kali Allison

---

Day 2: Thursday February 25

8:00-8:15 Recap of Day 1

SESSION IV: SHOULD CIG ADOPT A STANDARD INTERFACE FOR SPECIFYING VALUES FOR BOUNDARY CONDITIONS AND MATERIAL PROPERTIES?

8:15:8:25 Results of Developer Survey: Boundary condition and material property values, Presenter: Brad Aagaard

8:25-8:45 Existing Libraries

- World Builder (ASPECT), Presenter: Menno Fraters
- Spatialdata (PyLith), Presenter: Brad Aagaard
- easi (SeisSol), Presenter: Carsten Uphoff

8:45-9:15 Discussion Topics:

- What are the critical features for specifying values for boundary conditions and material properties?
- What would be the key benefits of developing common infrastructure (API and library) for boundary conditions and material properties?
- How feasible is it for CIG to develop common infrastructure (API and library) for boundary conditions and material properties?

Moderators: Menno Fraters and Brad Aagaard, Scribe: Rene Gassmoeller

SESSION V: IMPROVING MODELING WORKFLOW

9:15-9:20 Results of Developer and User Surveys: user obstacles, pre- and post-processing, modeling workflow

9:20-9:30 Overview of Pangeo, Presenter: Lindsey Heagy

9:30-10:00 Discussion topics:

- How can we make it easier for users to troubleshoot simulations?
- How do we provide better data/model integration?
- How do we facilitate reproducibility and/or replicating someone else's results?
- How do we facilitate transparency (understanding someone else's results and modifying their model to make new models)?

Moderators: Marc Spiegelman and John Naliboff, Scribe: Lorraine Hwang

10:00-10:30 Break

SESSION VI: GROWING THE CIG DEVELOPER COMMUNITY

10:30-10:40 Results of Developer and User Surveys, Presenter: Brad Aagaard

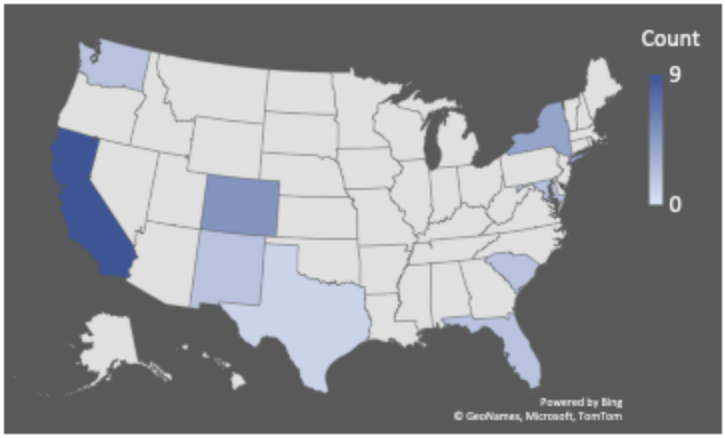
10:30-12:00 Discussion topics:

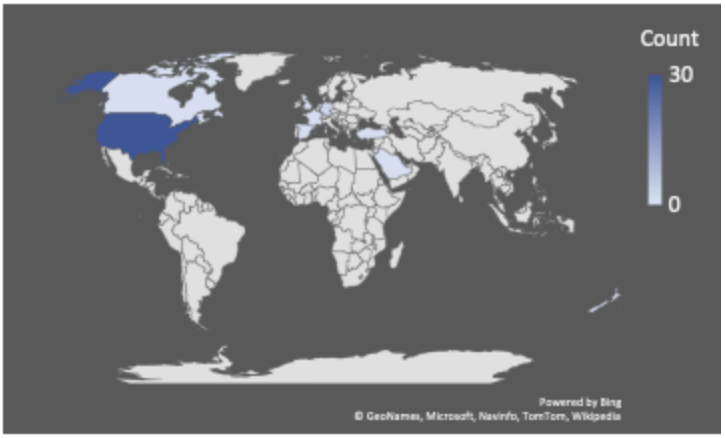
- How do we facilitate the transition from users to user developers?
- How do we recruit developers from diverse backgrounds?
- What is the best way to acknowledge contributions to codes?
- How much time should developers spend on providing support to users?
- How should we organize code-related workshops?
- Should CIG develop a coding bootcamp?

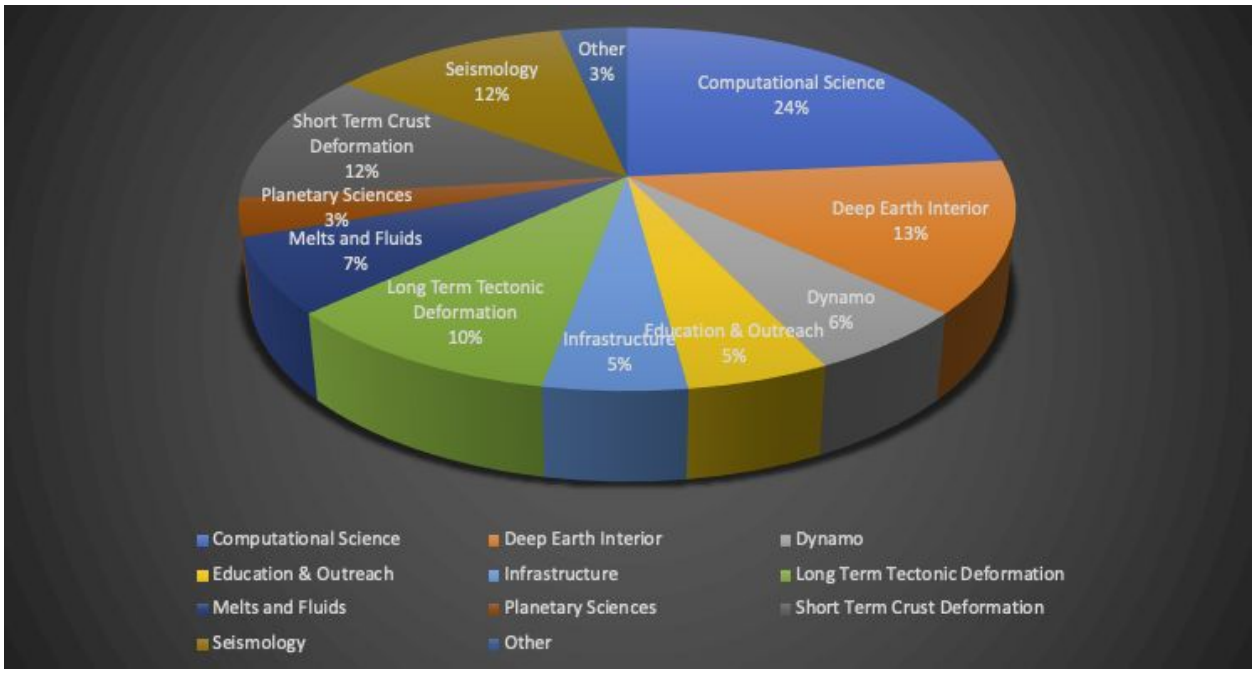
Moderators: Katie Cooper and Juliane Dannberg, Scribe: Lorraine Hwang

Participants

Total Attendees: 43







Brad Aagaard, USGS
Kali Allison, University of Maryland
Jean-Paul Ampuero, GEOAZUR - IRD
Wolfgang Bangerth, Colorado State University
Sylvain Barbot, University of Southern California
Ankit Barik, Johns Hopkins University
Thorsten Becker, UT Austin
Jed Brown, CU Boulder
Bruce Buffet, University of California, Berkeley
Recep Cakir, Washington Geological Survey
Catherine Cooper, WSU

Katherine Cosburn, University of New Mexico
Juliane Dannberg, University of Florida
Nick Featherstone, Southwest Research Institute
Menno Fraters, UC Davis
Alice Gabriel, Ludwig-Maximilians-Universität München
Rene Gassmoeller, University of Florida
Hom Nath Gharti, Queen's University
Lindsey Heagy, UC Berkeley
Timo Heister, Clemson University
Lorraine Hwang, UC Davis
Matt Knepley, University of Buffalo
Shengduo Liu, Caltech
Loren Matilsky, JILA, University of Colorado, Boulder
Hiroaki Matsui, University of California, Davis
Robert Moucha, Syracuse University
John Naliboff, New Mexico Institute of Mining and Technology
Tarje Nissen-Meyer, University of Oxford
Ryan Overdahl, University of Colorado
Fernando Perez, UC Berkeley
Daniel Peter, KAUST
Patrick Sanan, ETH Zurich
Ebru Şengül Uluocak, Canakkale University, Turkey
Marc Speigelman, LDEO
Cedric Thieulot, Utrecht University
Carsten Uphoff, Ludwig-Maximilians-Universität München
Jorge Nicolas Hayek Valencia, Ludwig-Maximilians-Universität
Ylona van Dinther, Utrecht University
Robert Walker, University at Buffalo
Charles Williams, GNS Science
Cian Wilson, Carnegie DTM
Jiaqi Zhang, Clemson University
Sergio Zlotnik, Universitat Politècnica de Catalunya