



Jet Propulsion Laboratory
California Institute of Technology

InnerSource at JPL: Collaboration around software in a science, technology, engineering and research enterprise

David Mittman, Manager
Enterprise and Information Systems Engineering Section

October 3, 2018



AGENDA

- ▶ What is Inner Source and Why Should I Care?
- ▶ How to Get Started with Inner Source
- ▶ Additional Resources and Acknowledgements

WHAT IS INNER SOURCE AND WHY SHOULD I CARE?

- ▶ An introduction to inner source
- ▶ Lessons learned from the open source community
- ▶ Why should the Lab adopt inner source?
- ▶ The anatomy of an inner source project

**“US[ING] OPEN SOURCE DEVELOPMENT
TECHNIQUES WITHIN THE CORPORATION”**

Tim O'Reilly

AN OPEN SOURCE COMMUNITY BEHIND OUR FIREWALL

- ▶ Inner source is not open source
- ▶ Inner source does not imply reduced security or privacy
- ▶ Adopting inner source practices is like starting an open source community within the Lab

WHAT MAKES A PROJECT OPEN SOURCE?

- ▶ The open source community spans the **interests and skillsets of millions of contributors**. Together, they work on programming languages like Apple's Swift, and frameworks like Facebook's React.js – and they don't just write code. Datasets, legal documentation, and more can also be open sourced.
- ▶ **Anyone can use open source software**. They can also view, modify, and distribute a project for any purpose – as enforced by open source licenses. Similarly, anyone can help build open source software. In general, projects are developed by a community of distributed contributors who share code, feedback, bug reports, and more.

LESSONS LEARNED FROM THE OPEN SOURCE COMMUNITY

- ▶ Open collaboration encourages more contributions
- ▶ Developers don't always have to start from scratch
- ▶ Transparent decision-making builds process, trust, and alignment
- ▶ Participation is critical
- ▶ Core development teams strengthen a project's process

OPEN COLLABORATION ENCOURAGES MORE CONTRIBUTIONS

- ▶ More contributions mean more opportunities. Open source projects can accept changes from anybody in the world. That's a lot of brainpower dedicated to advancing a project, meeting user needs, and finding and fixing bugs.
- ▶ Similarly, inner source brings more ideas to the table. Our teams can more easily innovate – and with more people inspecting code for errors and inconsistencies, **we can build more secure, more reliable software**. Problems are found and fixed before the wrong person discovers them.

DEVELOPERS DON'T ALWAYS HAVE TO START FROM SCRATCH

- ▶ Anyone can discover and reuse open source projects for nearly any reason. People can even use them to build other things or modify them to suit their specific needs.
- ▶ Similarly, inner source code helps us discover, customize, and reuse existing internal projects. We can also establish and build on a shared set of documented processes to optimize the way we deploy and use software. **This can lead to lower cost, greater flexibility.**

TRANSPARENT DECISION-MAKING BUILDS PROCESS, TRUST, AND ALIGNMENT

- ▶ Successful open source maintainers document their decisions by default. Because each conversation has its own URL and a history of comments for context, time zones matter less, and **developers can work asynchronously without skipping a beat.**
- ▶ Opening up our projects brings a new level of transparency to the Lab. Not only is the code visible but also the process and decision-making behind it. **Well-documented conversations** help developers on distributed teams **get up to speed and start building.** And with work happening in the open, collaboration can also include **product managers, designers, security teams, and more.**

PARTICIPATION IS CRITICAL

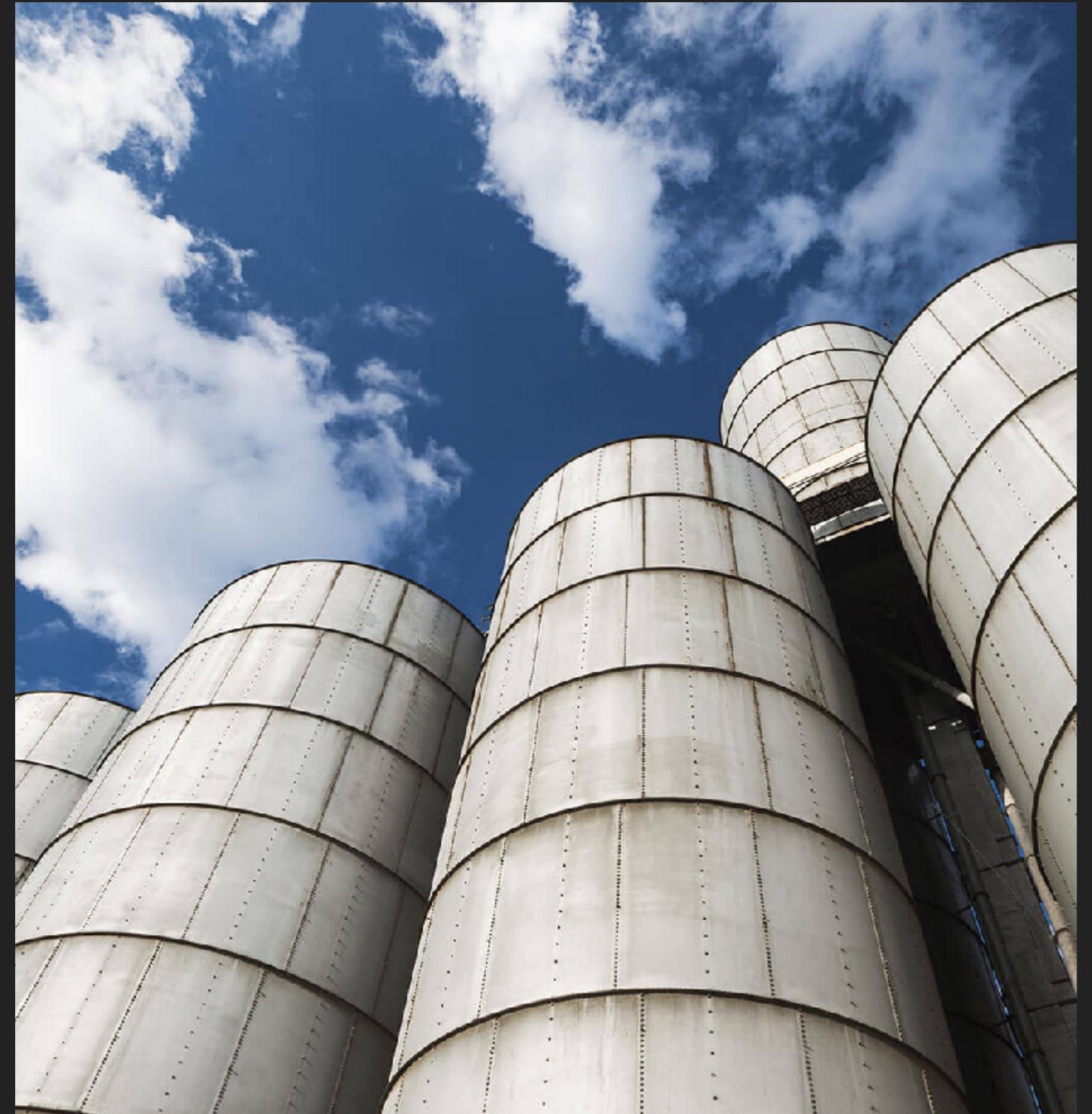
- ▶ The success of any open source project depends on participation. There are many built-in reasons to contribute – to **improve skills, find a mentor, or build a reputation**, for example – but project maintainers also need to **create a community culture that welcomes and encourages participation**.
- ▶ Within the Lab, individual developers can pursue their interests, share ideas on a level playing field, and more easily learn from their peers. However, inner source also requires a cultural shift. Our culture needs to **encourage knowledge sharing and welcome contributions** from across the Lab.

CORE DEVELOPMENT TEAMS STRENGTHEN A PROJECT'S PROCESS

- ▶ Open source projects may have thousands of contributors and community members, but a much smaller team is usually responsible for the project's overall direction. This speeds up decision-making and ensures that someone is always responsible.
- ▶ For inner source projects, distributing control across a smaller group of participants frequently makes approvals and reviews more effective. Creating a small, cross-functional team of decision makers can also help teams stick to quality standards and gain management support.

SILOS!

- ▶ The slow, systematic practice of gathering requirements, holding meetings, and **developing in silos** is not in step with the pace of technology today – or even the pace of customer demands.
- ▶ Our new developers expect to take the **lessons learned from developing open source software** and apply them to the way they develop software internally.
- ▶ Inner source can be a great tool to help **break down silos**, encourage **internal collaboration**, accelerate new **engineer on-boarding**, and identify opportunities to **contribute software back to the open source world**.



WHY SHOULD THE LAB ADOPT INNER SOURCE?

- ▶ Makes it easy to find and reuse code on a broad scale, avoiding wasted resources and duplication
- ▶ Drives rapid development
- ▶ Reduces silos and simplifies collaboration throughout the entire organization – inside and between teams and functions, as well as across teams and business lines
- ▶ Increases clarity between engineers and management, as well as anyone else who's interested
- ▶ Creating a culture of openness, a precursor to open source participation
- ▶ Reinforcing the pride, growth, and job satisfaction felt by team members who can help wherever there is a need

THE ANATOMY OF AN INNER SOURCE PROJECT

- ▶ Many open source projects follow a similar organizational structure – one we may want to consider when setting up cross-functional teams for the Lab's inner source projects.
- ▶ **Maintainers:** Contributors who are responsible for driving the vision and managing the organizational aspects of the project. They are not necessarily the original owners or authors of the code.
- ▶ **Contributors:** Everyone who has contributed something back to the project.
- ▶ **Community members:** People who use the project. They might be active in conversations or express their opinion on the project's direction.

THE ANATOMY OF AN INNER SOURCE PROJECT, CONT.

- ▶ Within the Lab, “contributors” are developers from across all our organizations, while “maintainers” are a project’s leaders and key decision makers.
 - ▶ Maintainers: Developers, product managers ([PDMs](#), [CogEs](#)), and other key decision makers within the Lab who are responsible for driving a project’s vision and for managing day-to-day contributions.
 - ▶ Contributors: Developers, data scientists, product managers ([CAMs](#)), marketers, and **other roles within the Lab that help drive software forward**. Contributors are not necessarily part of the direct project team, but help build software by contributing code, submitting bug fixes, and more.
 - ▶ Community members: JPL software developers who, through **enlightened self-interest**, contribute to the projects they use, or **“not invented here” developers who are converted** by the quality attributes displayed by the projects they would otherwise pass over.

TOOLS OF THE TRADE FOR INNER SOURCE PROJECTS

- ▶ **Issues** are where developers bring up topics and start conversations. If someone finds a bug or has an idea for a new feature, an issue is a great place to start – and anyone with access to it can join in on the discussion.
- ▶ **Pull requests** are living conversations about changes that developers would like to make to a project. They're where people start working on solutions and review changes that are in progress.
- ▶ Sometimes teams need to make quick decisions. Synchronous chat channels like **the JPL Slack team** are complementary to discussions and comments on GitHub Enterprise, and are great for talking through problems in real time.

IS INNER SOURCE RIGHT FOR YOUR TEAM?

- ▶ You have a **vision** for your project that is both realistic and shared across teams. Projects should have clearly defined problems and opportunities to be addressed.
- ▶ Key participants (initiators, catalysts, evangelists) in projects you plan to inner source have **experience working collaboratively**.
- ▶ You have a plan to **onboard and acclimate** new “contributors” and other participants into your process.
- ▶ Your team has the tools and processes to **communicate openly and build consistently**.
- ▶ You’re able to start with an intra-organizational group of people with **defined shared goals**.

IS INNER SOURCE RIGHT FOR YOUR TEAM?

- ▶ Inner source can shift how individuals see themselves and their responsibilities, making organizational structure an important consideration. An effective innersource process should be **informal, mentored, self-selecting, and supportive of its participants.**
- ▶ To effectively adopt inner source practices, contributors need to be able to **work easily across silos and other organizational divisions.** The degree to which an organization supports **knowledge-sharing initiatives** can be a good indication of how well they'll be able to adapt.

IS INNER SOURCE RIGHT FOR YOUR TEAM?

- ▶ Does the Lab have an **open and transparent culture**?
- ▶ Do we develop software on a **single, open platform**?
- ▶ Are our **engineering initiatives well resourced** and supported by leadership teams?
- ▶ Do developers have enough **autonomy to contribute to projects outside** their immediate teams?
- ▶ Does the Lab **participate in the open source community**?
- ▶ Do our engineering teams use **continuous integration tools**?
- ▶ Are there **pre-existing, cross-functional communities** that work across teams at Lab?
 - ▶ If yes, do these communities have **built-in leadership**?

HOW TO GET STARTED WITH INNER SOURCE

- ▶ The strategic landscape: Strengths, weaknesses, opportunities and threats
- ▶ Reflection and action, a retrospective: Start, stop, more and less
- ▶ Future state: Inner source patterns of success

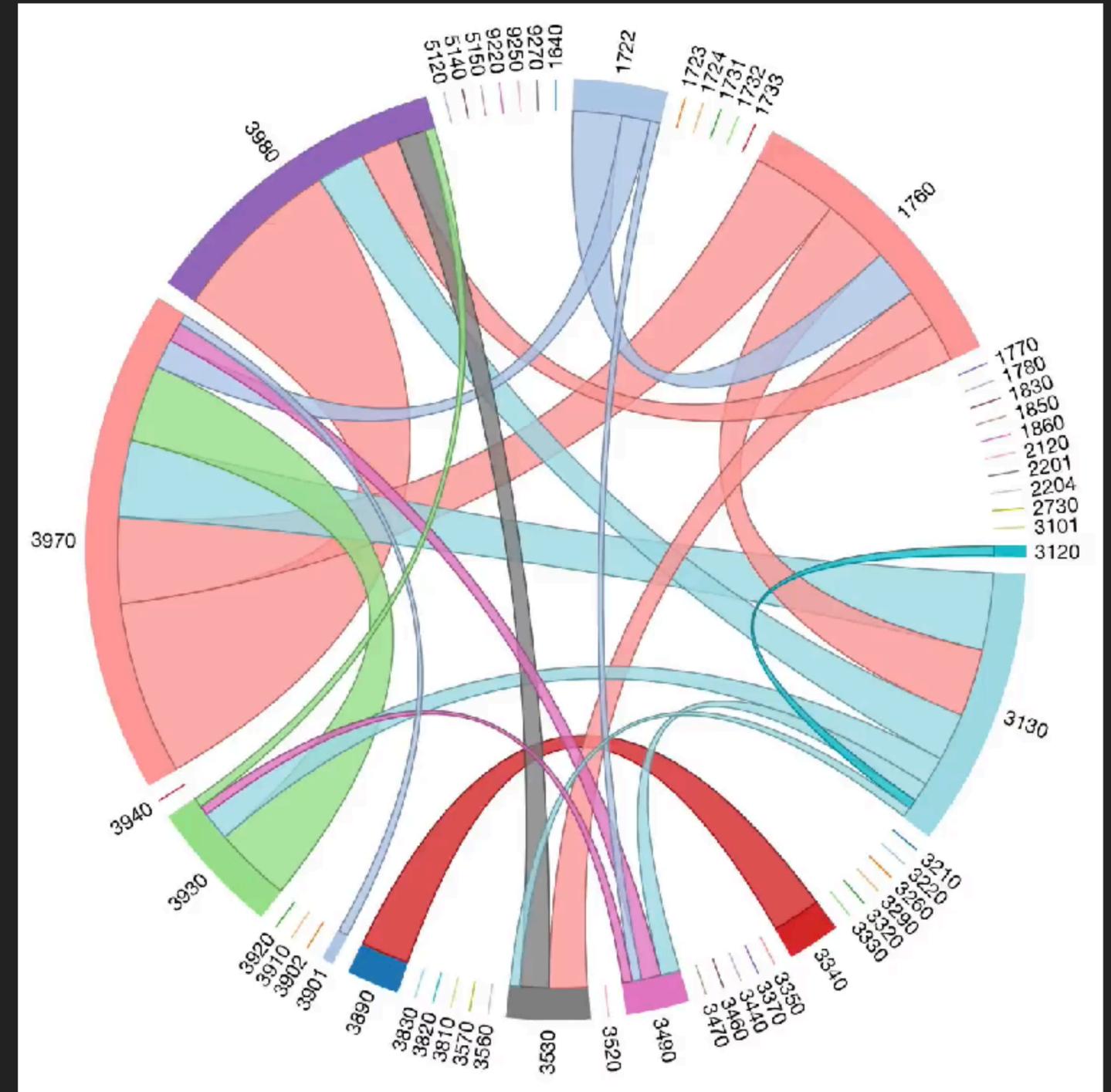
THE OTHER SWOT

- ▶ **Strengths:** characteristics of the business or project that give it an advantage over others
- ▶ **Weaknesses:** characteristics of the business that place the business or project at a disadvantage relative to others
- ▶ **Opportunities:** elements in the environment that the business or project could exploit to its advantage
- ▶ **Threats:** elements in the environment that could cause trouble for the business or project



GITHUB ENTERPRISE

- ▶ Introduced in May 2013
- ▶ Over 1,400 active developer users
- ▶ Over 14,000 repositories
- ▶ 52% of repositories are accessible to US persons
- ▶ Collaboration between JPL orgs has increased 20-fold and now encompasses 40 Sections
- ▶ A second FN server is available for non-export controlled projects



JPLERS OFTEN FIND THAT THEY LACK EASY ACCESS TO THE BASIC KNOWLEDGE THEY NEED TO DO THEIR JOB. THE KNOWLEDGE EXISTS ON SITE, BUT OUR KNOWLEDGE NETWORKS ARE NOT ALWAYS UP TO THE TASK OF SUPPLYING IT.

René Fradet, Director for Business Operations (2x)

10% OF THE WORK YOU DO ON ANY GIVEN DAY AT JPL, HAS ALREADY BEEN DONE BY SOMEONE ELSE AT JPL.

The Duplication Speculation

SIMILARITY TO THE OPEN SOURCE, WE'RE NASA, WE'RE CALTECH

- ▶ Large, diverse community of contributors
- ▶ Community of many small teams
 - ▶ Largest teams tend to be in flight software
- ▶ Ability to leverage “enterprise” versions of tools that have proven their value in the open source community
- ▶ We have a natural affinity with the high tech world as part of NASA and can leverage that affinity to access the investment of high tech in software engineering
- ▶ Our academic roots have contributed to a liberal environment for experimentation

THE COMPETITIVE ENVIRONMENT FOR TALENT AND CAPABILITY

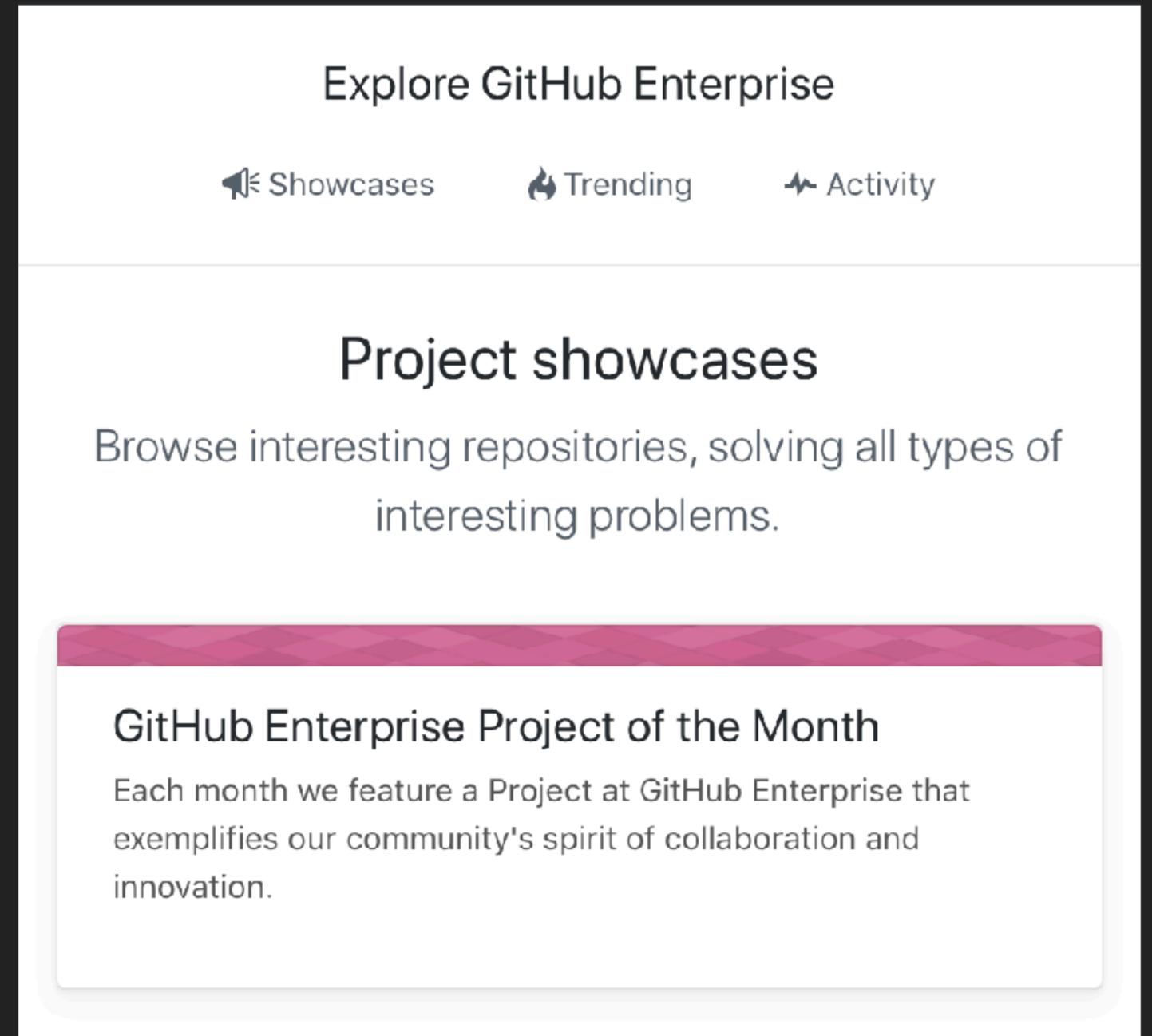
- ▶ Maintaining a highly skilled workforce in an expanding high-tech world
- ▶ Open source software is expanding into the space sector
 - ▶ Hundreds of thousands of Google hits for searches that involve open source CubeSat, ground station, mission operations, and flight software
- ▶ The cost for access to space is decreasing rapidly, and if we do not adapt, we will lose core segments of our business base

START SMALL, WITH A PILOT PROJECT

- ▶ Companies often kick off their inner source programs by **starting small**.
- ▶ Pilot projects can help teams **experiment with more open processes**, democratize access to code, and document best practices before applying inner source more widely.
- ▶ Small successes will help **show the Lab's community** of developers how to make the most of their code and create better software, faster.

GITHUB ENTERPRISE PROJECT OF THE MONTH

- ▶ Is your project in “public” mode?
- ▶ Does your project encourage collaboration with a set of Contributing Guidelines?
- ▶ Is your project actively accepting Pull Requests?
- ▶ Do you encourage users to try your product by including clear working instructions for download and installation?
- ▶ Do you demonstrate your commitment to quality by including badges indicating the status of your builds?

A screenshot of the GitHub Enterprise website's 'Project of the Month' section. The page has a white background with a dark blue header. At the top, it says 'Explore GitHub Enterprise' in a dark blue font. Below this are three navigation links: 'Showcases' with a speaker icon, 'Trending' with a flame icon, and 'Activity' with a lightning bolt icon. The main content area is titled 'Project showcases' in a large, bold, dark blue font. Below the title is a paragraph: 'Browse interesting repositories, solving all types of interesting problems.' At the bottom of the page, there is a white box with a dark blue border and a dark blue header that reads 'GitHub Enterprise Project of the Month'. Below this header is a paragraph: 'Each month we feature a Project at GitHub Enterprise that exemplifies our community's spirit of collaboration and innovation.'

CLONE AND OWN IS SHORT-SELLING THE LABORATORY

- ▶ Devalues the long view
- ▶ Prioritizes short-term gain over long-term value
- ▶ Lessens the value of bug fixes that get applied to one, but not all, copies of the project

MIXED MESSAGES

- ▶ You welcome contributions, but you haven't set expectations for what constitutes a welcomed contribution
- ▶ You welcome contributions, but you haven't established a standard for how long a pull request is allowed to stay open
- ▶ You welcome collaboration, but you don't welcome collaborators with self-learning resources or inclusive messaging



CHALLENGES FOR INNER SOURCE AT JPL

- ▶ Unlike the open source community, inner source is not a volunteer activity
 - ▶ How do we align inner source approaches with our requirement for task-funded work?
 - ▶ You're not working on inner source projects on your own free time
 - ▶ Inner source contributions are made with the full support of your task
 - ▶ What is the role of "incidental time" in improving the general state of the software development community on Lab?

A GREAT README EXAMPLE

- ▶ A bit of bling in the way of a logo
- ▶ Ample badges showing the product's build status – a sign of quality
- ▶ A very direct statement describing the project
- ▶ Quick guide for getting started using the product



Meteor is an ultra-simple environment for building modern web applications.

With Meteor you write apps:

- in modern JavaScript
- that send data over the wire, rather than HTML
- using your choice of popular open-source libraries

Try a getting started tutorial:

- [React](#)
- [Blaze](#)
- [Angular](#)

Next, read the [guide](#) and the [documentation](#).

Quick Start

On Windows, the installer can be found at <https://www.meteor.com/install>.

On Linux/macOS, use this line:

BONUS POINTS FOR DISCOVERABILITY AND BEST PRACTICES

- ▶ Short description
- ▶ Link to product website
- ▶ Use of Issues and Pull requests
- ▶ Use of topics to promote discoverability
- ▶ Code of conduct, contributing guidelines and a license file

meteor / meteor

Watch 1,898 Star 39,284 Fork 4,964

Code Issues 73 Pull requests 23 Projects 0 Wiki Insights

Meteor, the JavaScript App Platform <https://www.meteor.com>

javascript meteor mongodb build-system npm

CODE_OF_CONDUCT.md	Adjust policy doc filenames to use capital snake case	5 months ago
CONTRIBUTING.md	Fix typo in "Documentation" section (#9352)	4 months ago
DEVELOPMENT.md	Merge branch 'devel' into release-1.6	5 months ago
History.md	History.md entry for require("/package.json").meteor.mainModule.	2 days ago
ISSUE_TRIAGE.md	Adjust policy doc filenames to use capital snake case	5 months ago
IssueTriageFlow.png	More work on IssueTriage	2 years ago
LICENSE	Update license range for 2018 🎉 (#9523)	2 months ago

MAKE FEWER “PRIVATE” MODE REPOSITORIES

- ▶ Unless there's a requirement to make your project “private” mode, don't do it.
- ▶ Some examples of requirements
 - ▶ Space Asset Protection (Spacecraft Command Binary Patterns)
 - ▶ Personally Identifiable Information (PII)
 - ▶ Third-Party Proprietary Information
- ▶ Review your repositories assuming you'll make them “public” mode unless there's a requirement for them to be “private” mode

INNERSOURCE PATTERNS

- ▶ **Faster development:** Programmers use unit tests, code coverage, and continuous integration to remove bugs at early stages
- ▶ **Complete documentation:** Code is documented better, both in-code comments and less formally on discussion lists
- ▶ **Code reuse:** Programmers across the organization understand the code and architecture of modules developed by other teams
- ▶ **Cross-team collaboration:** Contributions by members outside of the team are frictionless and rarely have to be rewritten
- ▶ **Development with GitHub Enterprise:** GitHub Enterprise maintains “private” mode repositories only for projects requiring access restrictions, and “public” mode repositories for shared source code

WHAT'S NEXT?

- ▶ Join the InnerSource Commons – Founded by PayPal in 2015 as a forum for discussing and improving the practice of inner source
- ▶ Host an InnerSource Commons meeting at JPL
- ▶ Organize an InnerSource Symposium at JPL
- ▶ Join the [#innersource](#) channel in the JPL Slack team
- ▶ Participate in the monthly Open Developer Meetup
- ▶ Start a JPL Bug Bounty program

ADDITIONAL RESOURCES AND ACKNOWLEDGEMENTS

- ▶ The *InnerSource Commons* at <http://paypal.github.io/InnerSourceCommons/>
- ▶ GitHub's *Introduction to InnerSource* at <https://resources.github.com/articles/introduction-to-innersource/>
- ▶ *InnerSource at JPL* in the JPL Wiki at <https://goto.jpl.nasa.gov/innersource>
- ▶ The *#innersource* channel in the JPL Slack team at <https://jpl.slack.com/>
- ▶ Chris.A.Mattmann@jpl.nasa.gov, Manager, Advanced IT Research and Open Source Projects Office
- ▶ David.S.Mittman@jpl.nasa.gov, Community Organizer, Open Developer Meetup

GETTING STARTED WITH INNERSOURCE

Inspired by the spread of open source software throughout the areas of operating systems, cloud computing, JavaScript frameworks, and elsewhere, a number of companies are mimicking the practices of the powerful open source movement to create an internal company collaboration under the rubric InnerSource. In these pages, you'll read about the experience of the leading Internet commerce facilitator PayPal and see how InnerSource can benefit engineers, management, and marketing/PR departments.

Getting Started with InnerSource

Keys to collaboration and productivity
inside your company



Andy Oram

DO YOU HAVE YOUR OWN INNOVATIVE IDEA?



- ▶ Spark – How We Work Is As Important As What We Do
 - ▶ Find your Champion, Sponsor, Subject Matter Expert and Facilitator
 - ▶ Visit spark.jpl.nasa.gov
 - ▶ Email jplspark@jpl.nasa.gov
 - ▶ Contact Laura.B.Fisher@jpl.nasa.gov
 - ▶ Contact Tomas.J.Soderstrom@jpl.nasa.gov
 - ▶ Contact David.S.Mittman@jpl.nasa.gov



ACKNOWLEDGEMENTS

- ▶ Tom Soderstrom
- ▶ Carlos Balacuit
- ▶ Gary Richmond
- ▶ Evan Chan
- ▶ Ricky Ma
- ▶ Laura Fisher
- ▶ Bill Seixas
- ▶ Tara Estlin
- ▶ Rebecca Castano
- ▶ Arthur Amador
- ▶ Douglas Hughes
- ▶ Caleb Sweeney
- ▶ Virinder Dhillon
- ▶ Ian Roundhill
- ▶ Lyman Lam
- ▶ Chris Mattmann
- ▶ Minh Le
- ▶ Chris Delp
- ▶ Charles White

Portions of this presentation were adapted from [An introduction to innersource](#),
GitHub Resources Library.



Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov