OLF-ENG-2021-02

Open source, open standards, and collective invention in the space industry

Juno Woods, Ph.D.
Open Lunar Foundation
San Francisco, California

Chelsea McMahon
Open Lunar Foundation
San Francisco, California

July 1, 2021

Abstract

Collective invention occurs when free exchange of information enables rapid technological advance, and differs from individual invention and commercial invention (e.g. research and development). In an academic context, one example of collective invention is the open science movement. In the for-profit world, nominal competitors may work together on key infrastructure, called pre-competitive collaboration. Normally, the products of these efforts are eventually privately recaptured, but the free/libre and open source software (FLOSS) movement has created a legal mechanism to prevent that recapture. Moreover, collective invention is often a product of specific engineering cultures or participant ideologies. Silicon Valley engages in pre-competitive collaboration by producing open source infrastructure as a foundation for proprietary, closed source innovations; however, space industry collaborations are much rarer. We briefly review advantages and disadvantages of FLOSS, open hardware, and open standards. We discuss key barriers in the aerospace industry, as well as potential motivators for renewed participation, and make recommendations based on interviews conducted with anonymous space industry executives and several years of experience running open source projects.

1 Background

Until the 1980s, it was widely accepted that intellectual property protectionism (in the form of secrecy and patents) was the main economic driver of technological advancement, providing individuals and corporations with incentives to continue to invent. In 1983, however, Allen discovered circumstances in which "the free exchange of information about new techniques and plant designs" in the nineteenth century British coke furnace industry led to rapid technological evolution 1983. Subsequently, Nuvolari described a case — the Boulton and Watt steam engine — where a broadly written patent retarded innovation for decades before ultimately birthing an engineering culture that eschewed patents in favor of competitors publishing designs and data in an industry journal 2004. This, too, led to a period of rapid technological advance, and Nuvolari argued that collective invention was a key component in the Industrial Revolution. The *shanzhai* movement, which has played a key role in the Chinese electronics manufacturing revolution, is another clear example of collective invention [3].

In commercial settings, collective invention arrangements are termed pre-, pro-, or non-competitive collaboration. Competitors work together, usually on basic infrastructure, so that they can focus on aspects that differentiate their businesses. These infrastructure elements may take the form of standards development, training programs, jointly funded academic research, and others [4].

In academia and elsewhere, collective invention takes the form of 'open science.' It arose out of the European post-Renaissance patronage system, and developed into a reputational system as these early scientists' discoveries began to exceed their sponsors' capabilities to fully understand the products of the research [5]. In open science, the sharing of information needed to reproduce scientific experiments by potential competitors, as well as the sharing of the results of such explorations, enables rapid innovative iteration. The risks that academic faculty take in disclosing their work are balanced by rewards in the form of access to funding, increased credibility, greater job security (tenure), and so on. In the academy, David [5] writes, "the norm of openness is incentive-compatible with a collegiate reputational reward system based on accepted claims to priority; it is also conducive to individual strategy choices whose collective outcome reduces excess duplication of research efforts and enlarges the domain of informational complementaries."

In the late 1970s, collective invention found a place in the development of the UNIX operating system and other computer programs (several of which notably originated in or were nurtured by the academic patronage system). Although UNIX was owned by Bell Labs, now AT&T, a grey market for bug fixes grew up around it, with users developing patches and encoding these on magnetic tapes, which they shared widely. In this environment, Richard Stallman first began writing the GNU (GNU's Not UNIX) operating system at MIT in 1983, and following that, issued the first release of GNU Emacs in 1988 — along with a copyright notice requiring that the software always be accompanied by its source code and the original notice. This notice, the predecessor of the General Public License (GPL), marked the beginning of the Free (as in libre) Software movement. Schrape [6] noted that the GPL represented a major shift from prior cases of collective invention: while other technologies were subject to private appropriation, the GPL provided a legal mechanism that retained such public inventions as part of a permanent intellectual commons. This legal mechanism, founded in copyright law, has also come to be known as *copyleft*.

Concerns about the linguistic ambiguity of the term 'free software,' and consequent alienation of business interests, led to the adoption by many of the 'open source' label, first suggested by Christine Peterson in 1998 [7]. We use the terms free/libre and open source software, or FLOSS, interchangeably in this document, a common convention. The open source label is also often used to refer to hardware designs released under various licenses, such as TAPR and CERN, which are governed by somewhat different economics than open source software. For simplicity, we refer collectively to software and hardware released under open source or free software licenses as open source.

Not all FLOSS (nor open hardware) is copyleft, however. In other words, the culture has existed independently of the legal mechanism, having inherited much from its origins in the academic open science movement. These origins are visible in the names of open source licenses such as the Berkeley Software Distribution (BSD) License and the MIT License, which require attribution but are not copyleft; that is, these projects may be incorporated into commercial software distributions which do not include the original source code, so long as the copyright notice is retained. While coders could repackage and improve such applications and libraries, and then sell these programs for personal benefit, one can see that many projects so licensed nevertheless continue to accrue new contributors and evolve. There exist substantial social and economic forces which support the maintenance of these commons [8]. For example, Nuvolari [2] and Huang [3] both note the use of ostracism to deter private appropriation.

The segment of the technology industry based in Silicon Valley is known for its adoption of FLOSS, with the two being products of the same countercultural movement. The partnership formally began when Netscape released its Communicator web browser under an open source license in 1998. Red Hat, a company built upon the open source operating system Linux, was founded in the early nineties, went public in 1999, and was purchased by IBM for $34 billion in 2018. Even Microsoft, which once eschewed open source, has since embraced it [9]. Linux, Git, and other projects are ubiquitous in the modern technology ecosystem. The second order effects of these innovations make it difficult to measure the full economic impact of collective invention on the sector.

While the commercial space industry relies extensively on collective invention (e.g. academic and NASA research and development, as well as substantial tech industry infrastructure), it contributes much more rarely than the tech industry. While there exist a large number of successful FLOSS projects in the space industry, most are primarily government or academic in origin and support. We identified OreKit and OpenStack as rare public-private partnerships (with the caveat that OpenStack is today largely supported by non-aerospace companies); Ball Aerospace's COSMOS as a successful privately funded project; and Asterank as a FLOSS project acquired and supported for some years by Planetary Resources (while it existed). Encouragingly, Blue Origin received a 2020 NASA award to support development on a Robot Operating System (ROS) variant optimized for use in outer space [10], with the aim to reduce software costs and increase interoperability. Smaller commercial FLOSS projects exist as well, such as Intuitive Machines' ThinVPU (funded by a NASA Tipping Point award in early 2020), but most of these lack the staffing and finances needed to build a community or development consortium.

A number of factors may be at play, from cultural to economic. We explore these factors, as well as the potential benefits of pre-competitive collaborations for space industry actors, in this manuscript. While previous works have examined the role of open source specifically, we focus on the culture of innovation in the space industry and attend to the benefits of shifting that culture toward greater openness and collaboration.

2 Economics of collective invention

Economic motivations for collective invention appear at both the individual and the group level. Groups might be companies, governments, or academia, and accomplish their goals by motivating other groups or individuals to participate.

There are a number of common motivations for collective invention by organizations, reviewed by Schrape [6]. In the case of the coke furnaces, the technology was both not patentable and arose out of the normal course of doing business rather than investment in research and development [1]. Collective invention reduced the generation time of incremental innovations. With the Cornish pumping engines, the advances were instigated by a backlash against a patent viewed as overly broad [2].

In the flat panel display industry, collective invention improved innovative performance by allowing participants to control the direction of technological development [11]. Microsoft has also used this strategy [9]. In mobile phones, "Because it fully controls the development of the OS, Google can determine the technological specifications to which Android partners must abide" [12]. Dahlander and Magnusson [13] suggested participation in collective invention or standards creation may occasionally be a marketing tool. Sometimes, Lerner [14] points out, companies use collective invention as a loss-leader, "comparable to giving away the razor (the code) to sell more razor blades (the related consulting services [...])," including IBM, Intel, HP, SAP, Oracle, and Adobe. Patents may also play a role, given the challenges with patenting software.

Open science has shifted from a peerage patronage system to a public patronage system in many nations, with nobles being replaced by government agencies as the sponsors. Beaudry and Allaoui [15] reviewed the economics of academic research, which is largely publicly funded. The goal of such funding is to further those innovations viewed as important at the national level. Such advances may be adopted by industry and taught to students who will later work in said industry, fueling economic growth. Academic research, since the Renaissance, has also been a form of " 'common agency contracting,' involving the competition of incompletely informed rival principals for the dedicated services of an expert agent" [5]. That is, having access to multiple sources of funding enables academics to cultivate expertise independently of, or at least above and beyond, what a single funding agency might be seeking.

Individual motivations in the FLOSS movement have also been studied extensively, and may mirror those of individual academic researchers. Where the so-called permissive open source licenses failed to provide the commons protection Schrape observed in the GPL, a variety of individual and economic motivations stepped in to take their place. Von Krogh et al. [8] reviewed motivations of individual developers, identifying three categories:

1. **intrinsic**: ideology, altruism, kinship amity, and enjoyment;
2. **internalized extrinsic**: reputation, reciprocity (e.g. gift economy), learning, and personal use; and
3. **extrinsic**: career (e.g. open source contributions as a work portfolio) and for a paycheck.

Notably, FLOSS differs from open hardware in a few key ways. Firstly, open source software has near-zero marginal cost; that is, once programmed, it costs almost nothing to make copies. While hardware designs have near-zero marginal cost, the marginal cost of iterating through prototypes or manufacturing the physical hardware is larger. Whereas the ubiquity of modern computers equips many individuals with the tools to download, compile, and run open source software; or to download, view, and modify hardware designs; fewer individuals have access to the tooling needed to prototype hardware. This may be a temporary state, however, as demonstrated by do-it-yourself semiconductors [16], 3D printer technology, and most especially the recent Chinese *shanzhai* maker movement [3, 17, 18].

Finally, we refer readers to Swann [19] for a review of the economics of standardization. Briefly, there are four models of standards in the literature: compatibility or interface, minimum quality and safety, variety reduction or focusing, and information and measurement. Each has different positive and negative effects. Interface standards provide for network effects (e.g. the WiFi Alliance's standards enable wireless networking cards to work anywhere in the world, and also ensure interchangeability of components). They can also promote monopoly, particularly in the case of standards with patent protection or other closed intellectual property. Quality standards prevent bad products from driving good products out of a market (Gresham's law), but also raise the cost of entry into a market (known as regulatory capture). Variety reduction enables economies of scale but limits the available options. Information standards — such as fuel octane ratings — behave like quality standards in that they reduce transaction costs by helping consumers make informed choices. They may also facilitate regulatory capture, particularly when governments refer to standards definitions in law or regulations. There are also de facto standards — or customs — which are descriptive of the design norms of a field rather than being prescriptive.

2.1 Pre-competitive collaboration

Collective invention often differs between the public and commercial worlds. In academia — and sometimes also in the tech industry, with certain exceptionally talented individuals — the coder or researcher is in-demand and therefore has the agency to control the direction of inventive pursuits. In most commercial settings, as in much other industrial research and development, the work is toward a specific goal and is usually highly directed (often called applied research).

Explicit in the term 'pre-competitive' is that such collaborations revolve around non-differentiators (those technologies that help businesses compete against others in the economic niche are known as differentiators). The line between differentiators and non-differentiators is somewhat fuzzy. A car's steering wheel is common between all street-legal cars and probably would not be a differentiator — right up until one competitor starts producing cars with steering wheels which are markedly different in quality from other available choices.

Legally mandated engineering requirements — such as steering wheel form factor — are often non-differentiating for businesses and a good target for pre-competitive collaborations. Demonstrating compliance to the relevant government agency is often expensive, often serving as the ticket to ride but not always as useful as a marketing tool. One example is the Automotive Open System Architecture (AUTOSAR), a global development partnership between car manufacturers and the companies that build hardware and software to support them. Founded in 2003, the cooperative — whose over 280 members include BMW, Ford, GM, Toyota, and Volkswagen — aims to standardize software architectures of automotive electronic control units (ECUs, or the vehicles' computers) [20]. Participating organizations join working groups which lay out specific engineering requirements and then develop a baseline software that can be shared openly within the industry.

One example of pre-competitive collaboration is the open consortium model, as demonstrated in the GENIVI Alliance, which produced a Linux-based platform for in-vehicle entertainment, and publishes a variety of open standards [21]. GENIVI was founded in 2009 between auto industry competitors including OEMs such as BMW, Honda, and Hyundai, as well as other supply chain participants like Clarion, Bosch, LG, Garmin and Nvidia.

Frequently, pre-competitive collaborations outside of the tech industry also utilize open source methodologies. Automotive Grade Linux (AGL) is a Linux Foundation project, launched in 2012 by a variety of competing automotive and tech entities, including Jaguar, Land Rover, Nissan, Toyota, Nvidia, Samsung, and Texas Instruments. It satisfied the need for an appropriate implementation of the Linux kernel for automotive purposes [22].

In advocating for greater pre-competitive collaboration in the space industry, it's important to understand what makes such collaborations succeed or fail. We turn next to this topic and try to separate out the ideology from the facts.

3 Relevant factors in space industry collaboration

3.1 Open source and open hardware

In making software quality comparisons, identifying a useful sample of programming projects — whether open or closed — is a challenging task. While GitHub and SourceForge host many FLOSS projects, the low bar for creating such repositories ensures that more are inactive and abandoned or nearly dead than those that aren't. In measuring quality, we are most interested in active projects, but the threshold for 'active' can be placed at a variety of levels. Moreover, there are perhaps fewer things in common between individual open source projects than are different, in terms of organizational structure, process for accepting patches, design philosophy, licensing, and community membership, among other factors. Beyond that, 'software quality' is largely subjective, and those studying FLOSS have struggled to offer robust metrics, as reviewed by Ruiz and Robinson [23]. Approaches include structural quality, the structure of the code, including commenting or modularity, among other things; process quality, such as defect fixing, tool usage, testing methodology, group consensus, and project management; and community, such as adoption and usage, member activity, social network analysis, culture, and documentation. Yet, as with code quality, most of these are difficult to define uniformly and challenging to measure [23].

A key challenge to the quality comparison is that the processes and innovations — if not the source code — employed in a well-regarded open source project may always be appropriated for closed source work. The FLOSS movement has provided one transparent experiment after another on project management strategies, some successful and others not; many of these have been borrowed by the agile software development methodology, now widely adopted in many industries that write code.

To be adopted broadly, FLOSS projects must nearly always be coded and well documented with reuse in mind. Of course, there is no guarantee that a given open source project has been so written. For example, a not-uncommon problem with aerospace graduate student projects is that, while licensed as open source, the code was pushed out the door quickly with graduation rather than community-building in mind. Such projects might not be ideal sources for mission-ready products.

In studies on closed source software, building in reusability entails an upfront cost ($\sim$2–5$\times$), but produces a positive return-on-investment within a few years; moreover, the cost of integrating components written with reuse in mind is a fraction of the cost of writing new components [24, 25, 26, 27, 28]. In other words, single-use code is usually cheaper than well-commented, well-designed code — code that makes sense under the eye of another experienced developer or even by the same developer several years later. The same single-use code might be substantially more expensive to dust off and reuse later than it would have been to make reusable in the first place.

The upfront cost estimates also factor in the challenge of publishing software in such a way that it is seen as less time-consuming on average for others within an organization to be able to find it than for them to write their own solutions, a matching problem which can scale quickly as the number of coders increases. Today, this problem is often solved using services like GitHub (first released in 2008) or GitLab (2011), which enable the average coder to quickly grok the (subjective) quality of code and documentation in a project, an order of magnitude on the number of users, how involved its maintainers are and how many they are in number, how quickly bugs are addressed, the size of the code-base and its linguistic make-up, licensing (where appropriate), basic usage instructions, philosophy on unit testing, verification and validation practices (e.g. continuous integration services), and so on. In open source — or potentially even in closed source projects at larger organizations — these same tools serve not only potential users, but also the project, by simplifying the on-boarding of new developers and volunteers.

We might call this property accessibility.

Sometimes language choice also contributes to accessibility. MathWorks' MATLAB, popular in the space industry, lends itself to sharing, but not to accessibility or collaboration — at least, not in Mathworks' public repository, MATLAB Central, which doesn't readily support outside contributions to existing libraries. While many MATLAB scripts can be run in GNU Octave, MATLAB coders are less likely overall to make use of services like GitHub to build projects designed for community participation than are Python or Ruby coders. Moreover, many FLOSS enthusiasts have strong reservations around the use of MATLAB, since use of the language and interpreter requires the purchase of a license. While Python has seen greater adoption over the last few years as a replacement, Julia has experienced a recent surge in popularity due to its speed [e.g. 29, 30, 31, 32].

In general, these arguments support the contention that many if not most of the practices involved in making good open source software are also practices that would have a positive return on investment for aerospace companies, whether they release it or not. It is next useful to ask in what ways releasing source code or hardware designs may be beneficial or harmful to space industry companies.

Successful FLOSS business models are well-documented. The Commercial Open Source Software Index (COSSI) is a list of the fifty-one companies that have achieved revenues of $100 million or more primarily through business models relying on open source [33]. By far the most common business model is *open core*, pioneered with Netscape Navigator, which comes in a variety of flavors. Some companies offer a product with dual licensing, such as a paid license for commercial use, but with free non-commercial use. Others contribute to an open source core product but sell proprietary extensions. One company, VA Linux, made use of hardware sales; Mozilla utilizes advertising revenues and royalties; and SUSE and Red Hat both rely on support services contracts. Red Hat (which distributes Red Hat Linux) is the most successful of these in the index, and Rackspace (OpenStack) the runner up [33], each with annual revenue in the billions of dollars.

Pearce [34] provides a convenient review of open hardware business models and examples of companies using them, among which are kit suppliers (Adafruit, RepRap), suppliers of specialty components of parts needed for open hardware (Shapeways, OpenBeam), calibration and validation services (Ocean Optics), open hardware assembly and resale (particularly for those who would prefer not to build open hardware themselves; Aleph Objects, MatterHackers, Adafruit, Snootlabs, Makershed), market creation through the driving of open standards (Tesla), subscription to services surrounding an open hardware product (as RedHat does with open source software), support and training (Open Source Ecology), consulting to make customized or more sophisticated variants of an open hardware item (sans example), servicing a pre-competitive collaboration consortium to solve a specific problem (an unnamed Canadian photovoltaic study), and providing business support services to other open hardware firms (Seeed Studios). The authors also note that open hardware companies save on legal fees for intellectual property protection — with some companies building proprietary software spending more on such legal bills than on engineering.

We conducted a series of anonymous interviews with space industry executives, all at small companies, and identified a number of key areas where open hardware might be especially useful. We offered an open source star tracker as an example product in these conversations. First and foremost, the specialized nature of many spacecraft components would make it difficult for all but the largest companies to download a design and spin up hardware manufacturing; interviewees indicated they would generally prefer to pay a reasonable price for hardware built by an experienced manufacturer over hiring or reassigning personnel to produce a limited run of devices for a project. A key factor was mission schedule risk, which many would pay a premium to avoid. Another issue raised related to the need for verification and validation of hardware devices to be flown on spacecraft, and that companies would usually prefer to pay the manufacturer for such services over conducting them in-house. All interviewees expressed a desire to support open source and open hardware projects. We also observe that missions tend to source custom components to accommodate varying engineering requirements; so the customization services business model may be especially applicable in the space industry.

There are several indirect benefits of the use of open source and open hardware in the space industry. Such code and design examples can be used for pedagogy in universities, leaving students better prepared to work on these platforms when they enter the workforce. A benefit which accrues to the community is the preservation of useful intellectual property in the public domain if the creator goes out of business. Planetary Resources, now Consenys Space, released its patents, but not its source code or designs (due mainly to concerns about export controls), when it shut down.

In the market interviews we conducted on open source and open hardware, several interviewees also expressed the belief that it wasn't possible to make money with open source broadly, or specifically within the space industry, due to free-riders. This objection has been explored extensively in both open source and economic literature. For example, Johnson [35] posits a game theoretic model which touches on free-riders; the model points out that development of a piece of open source software lowers the value proposition for someone else to create a competing or redundant piece of software. The game theoretic model does suggest, however, that free-riding becomes more of a problem as the number of user-developers increases [35]. Nobel laureate Elinor Ostrom [36] has extensively studied management of common pool resources; she identifies eight design principles present in successful resource management schemes and suggests a framework for creating new ones and managing the problem of free riders. Perens [37] reminds us that a free-rider on a bus uses a scarce resource, but for non-rival goods like software, the resource remains available. Provided "there are a sufficient number of individuals who do not free ride," Weber [38] goes further, arguing that open source is anti-rival rather than non-rival "in the sense that *the system positively benefits from free riders.* Some small percentage of free riders will provide something of value to the system, even if it is just reporting a bug out of frustration. The more free riders in this setting, the better."

The most frequent concern with releasing such products, however, related to export controls, which we discuss next.

3.2 Export controls

Hesitance regarding international collaborations, open standards, and FLOSS in the space industry has been driven by caution around export control laws — particularly in the US, which is by far the largest contributor to open source. Woods [39], in their history of multilateral and US export controls and the space industry, discuss the origins of these attitudes. A key factor was the congressional investigation of Hughes and Space Systems/Loral in the 1990s over export control violations [40]; these companies were fined a total of $52 million [41, 42], and Congress subsequently tightened controls. A Defense Department study found that US commercial satellite manufacturers had lost $2.35 billion in sales as a result of the new policies [43], however, and policymakers initiated a series of reforms in 2012 in an attempt to rescue the industry.

Today, the export control landscape is dramatically different than it was ten years ago, but this information has been distributed unevenly. Confusion reigns over, for example, whether technical standards may provide technical data as regulated by ITAR. Several export control attorneys informed us in personal communication that the State Department is concerned about hardware, and that this concern generally does not extend to standards, open source software, or open hardware designs. One attorney involved in the 2012 reforms indicated that the State Department has First Amendment concerns around enforcement of the 'deemed export' provisions on technical data as applied to FLOSS, and so "regulations diverge from practice." Both the State Department and Commerce Department regulations (ITAR and EAR respectively) already include exemptions for items that have been published, particularly in an open science or academic context, or in libraries. [39]

While many companies are understandably more concerned about the State Department's regulations than its practices, others have posted open source software and even hardware designs accompanied by export notices. For example, Swift Navigation, in San Francisco, includes an export notice in its open source GPS receiver code [44]. The US-based Open Research Institute posts on its website, "Our ITAR strategy does not apply to physical objects such as space satellites, but to their designs and the software which is part of them... ORI does not provide defense services." ORI volunteers also do all of their design work in public, on GitHub, so that no export occurs [45].

The Commerce Department's EAR includes a clear public domain exemption which requires no export license for Internet postings. In July 2020, the Linux Foundation published a helpful report on FLOSS and EAR which suggests best practices for open source communities concerned about export controls [46]. Just as some open hardware companies may save money on legal fees relating to intellectual property protection, we expect savings to be found in export control compliance.

Notably, launch companies are less able to avoid export controls, with greater scrutiny given to technologies that might be used to build or improve ballistic missiles. Unfortunately, the cat may already be out of the bag, as North Korea — one of the key states targeted by US export controls — has been exporting ballistic missile technology since the 1980s [47, 48, 49].

The US government imposes a significant regulatory burden on space companies — especially small businesses — with ITAR and EAR, despite the public domain exemptions. Given the hundreds of dollars per hour that export control attorneys charge for their services, the government ought to provide legal services grants for space startups oriented toward enabling innovative activities in the gray areas of export control. The State Department could also choose to clearly exempt open source software and open hardware from export licensing requirements.

3.3 Infrastructure versus business differentiators in the space industry

The idea of infrastructure products as compared to those that enhance market value for an individual entity is perhaps embodied best by either the open core or hardware business models for open source companies.

As an example, consider Astrobotic's lunar terrain-relative navigation (TRN) solution, OPAL. The software appears to be constructed primarily using libraries that are already open source: a Xilinx implementation of OpenCV atop NASA Goddard's Core Flight Software (CFS) [50]. It works by detecting visible features in terrain images and mapping them to known features in the lander's on-board database [51]. Astrobotic produces the database through a multi-scale renderer, which ray-traces using a variety of publicly available digital elevation models and albedo maps produced from the Lunar Reconnaissance Orbiter and Clementine [52]. Both the renderer and the feature detection pipeline are proprietary.

Astrobotic's risk in keeping its TRN solution closed was realized when Intuitive Machines published its own solution, ThinVPU, under the hardware business model. The mere existence of an open source option reduces the market value of Astrobotic's solution. Even without IM's solution, the market is likely to eventually produce an open version which competes with Astrobotic's. Johnson [35] suggests that "the optimal response is to invest in development with probability one...if the value-to-cost ratio is sufficiently high." In this case, the value would be determined by many factors: the risk of using a black box in a mission-critical context, the number of missions (the addition of which bears a marginal cost), the 'not invented here' mentality, the mission lead-time, and the lead-time for Astrobotic's hardware, the cost of flight-qualifying new hardware, *etc.*

Taking Astrobotic's TRN as a case study, the following are possible methods for participation in open source:

- Open core: Dual-license the renderer.
- Open core: Contribute functionality from the recognition pipeline back to the Xilinx OpenCV implementation without divulging the full pipeline.

- Open core: Dual-license the recognition pipeline.
- Hardware: Release the full source code for the renderer and the pipeline, but sell the hardware.

Each of these is largely independent of the others, and each contributes to the goal of decreasing the value for other companies to develop competing products.

Commercial entities considering open source business models ought to consider not only the existing market but how to facilitate the existence of a future market. Organizations working outside of low-Earth orbit, such as IM and Astrobotic, would benefit more from actions that increase the market size than from finding customers among the currently extremely limited market. Smaller companies should consider the possibility that larger, more vertically-integrated aerospace companies — several of which don't participate in any form of collective invention and have no incentives for change — are their greatest competition in a much longer game.

3.4 Government incentives for collaboration

NASA currently provides add-on incentives to certain grants for activities that involve primary, secondary, or undergraduate education in space industry research and development. For example, there is up to 30–50% more money available atop the NASA Flight Opportunities awards for "meaningful involvement of students in the design and development of the proposed technology" or "a secondary educational payload with direct K–12, collegiate student, or K–12 educator involvement in its design and development" [53]. If NASA wishes to promote the kind of collaboration we have explored in this paper, it ought to also provide incentives for the development of open source technologies.

Under the 1980 Bayh–Dole Act, government contractors may elect to retain ownership of government-funded inventions through their disclosure to the government. The government itself receives a license to practice the invention, but cannot transfer said license. A goal of this law was to facilitate commercialization of publicly funded inventions [54]. Consequently, the power to publish hardware designs and source code generally resides with the contractor. In the biomedical sector, some have argued for institutionalized norms around using Bayh–Dole to preserve the open science commons while also creating incentives for commercialization of technologies, on the grounds that public domain technologies are more difficult to leverage commercially [e.g. 55]. These arguments, which look favorably upon Bayh–Dole, are basically analogous to those underlying the open core business model. We recommend that space industry borrow these ideas from biotech and institutionalize the open science norm for publicly funded innovations — beyond current open access policies for peer-reviewed, government-funded research. That could involve carrot-shaped incentives from funding agencies or stick-like pressure from other stakeholders, and though we expect the latter would require significant cultural shifts over a longer time frame, it is a common strategy in engineering cultures which harness collective invention.

Funding sways culture. The availability of funding for commercial projects like Human Landing Systems, Commercial Lunar Payload Services, Commercial Crew, and so on has — for example — swayed the culture away from government-run projects like the Space Launch System, and has enabled unprecedented growth in the New Space sector over the last decade.

4 Conclusion

In this work, we discussed several mechanisms through which competing organizations might collaborate to produce rapid advance, and argued that such collaboration seems to be less common in the commercial space industry than it is in the incredibly successful engineering culture of Silicon Valley. We described the relationship between collective invention, pre-competitive collaboration, and free/libre and open source software, while avoiding idealism around the behavior of businesses in the current legal and socioeconomic framework. We considered the current industry-specific obstacles and advantages to collaboration, and made recommendations for circumventing obstacles and utilizing advantages. We emphasized the importance of the long-term market as compared to short-term competition in the fairly limited present-day market. We conclude now with a few additional thoughts about longer-term vision.

The founder effect is a well-studied phenomenon in evolutionary biology, linguistics, sociology, and anthropology, wherein the traits of the founders of a population are over-represented in the population. As with genes, the founders' cultural memes are preserved, recombined, and magnified over many generations.

We stand at an inflection point in the growth of the commercial space industry. The cultural norms at space industry companies today will shape the norms and laws of the societies we build in space. We need to ask ourselves if we wish to build company towns or open societies. Ideals aside, in the incredibly harsh environment of space, rapid adaptation as well as a culture of open standards and free sharing of information is essential to the preservation of human lives. Standards promote interchangeability of parts and compatibility of interfaces, which — as required under the Outer Space Treaty — will save lives when rescue or aid is inevitably rendered.

5 Acknowledgements

The authors wish to thank Chelsea Robinson, Jessy Kate Schingler, Angeliki Kapoglou, Michael Taczak, Lydia Laurenson, Hugh Wimberley, Jonathan Stray, and Ian Baker for helpful comments. This work was supported by Open Lunar Foundation.

References

[1] R. C. Allen, "Collective invention," *Journal of Economic Behavior and Organization*, vol. 4, no. 1, pp. 1–24, 1983. [Online]. Available: http://dx.doi.org/10.1016/0167-2681(83)90023-9

[2] A. Nuvolari, "Collective invention during the British Industrial Revolution: The case of the Cornish pumping engine," *Cambridge Journal of Economics*, vol. 28, no. 3, pp. 347–363, 2004.

[3] A. Huang, "Tech trend: Shanzhai," in *Bunnie's Blog*, 2009. [Online]. Available: https://www.bunniestudios.com/blog/?p=284

[4] H. I. Fusfeld and C. S. Haklisch, "Collaborative industrial research in the U.S." *Technovation*, vol. 5, no. 4, pp. 305–315, feb 1987. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0166497287900691

[5] P. A. David, "Common agency contracting and the emergence of 'open science' institutions," *American Economic Review*, vol. 88, no. 2, pp. 15–21, 1998.

[6] J.-F. Schrape, “Open-source projects as incubators of innovation: From niche phenomenon to integral part of the industry,” *Convergence*, vol. 25, no. 3, pp. 409–427, 2019.

[7] C. Peterson, “How I coined the term ‘open source’,” February 2018. [Online]. Available: https://opensource.com/article/18/2/coining-term-open-source-software

[8] G. von Krogh, S. Haefliger, S. Spaeth, and M. W. Wallin, “Carrots and rainbows: Motivation and social practice in open source software development,” *MIS Quarterly: Management Information Systems*, vol. 36, no. 2, pp. 649–676, 2012.

[9] Microsoft, “Annual Report 2017,” Redmond, WA, 2017.

[10] L. Hall. (2021, Feb) 2020 NASA Announcement of Collaboration Opportunity (ACO) Selections. [Online]. Available: https://www.nasa.gov/directorates/spacetech/2020_NASA_Announcement_of_Collaboration_Opportunity_ACO_Selections

[11] J. W. Spencer, “Firms’ knowledge-sharing strategies in the global innovation system: Empirical evidence from the flat panel display industry,” *Strategic Management Journal*, vol. 24, no. 3, pp. 217–233, 2003.

[12] K. Spreeuwenberg, “Android and the political economy of the mobile Internet: A renewal of open source critique,” *First Monday*, vol. 17, no. 7, 2012. [Online]. Available: https://firstmonday.org/ojs/index.php/fm/article/download/4050/3271

[13] L. Dahlander and M. Magnusson, “How do firms make use of open source communities?” *Long Range Planning*, vol. 41, no. 6, pp. 629–649, 2008.

[14] J. Lerner, *The Architecture of Innovation: The Economics of Creative Organizations.* Harvard Business Review Press, 2012.

[15] C. Beaudry and S. Allaoui, “Impact of public and private research funding on scientific production: The case of nanotechnology,” *Research Policy*, vol. 41, no. 9, pp. 1589–1606, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.respol.2012.03.022

[16] S. Zeloof, "Home chip lab," 2017. [Online]. Available: http://sam.zeloof.xyz/category/semiconductor/

[17] A. Huang, "The $12 'Gongkai' phone," in *Bunnie's Blog*, 2013. [Online]. Available: https://www.bunniestudios.com/blog/?page_id=3107

[18] S. Lindtner, "Hacking with Chinese characteristics: The promises of the maker movement against China's manufacturing culture," *Science Technology and Human Values*, vol. 40, no. 5, pp. 854–879, 2015.

[19] G. M. P. Swann, "The Economics of Standardization," University of Manchester, Manchester, United Kingdom, Tech. Rep., 2000.

[20] "AUTOSAR introduction: The vision, the partnership, and current features in a nutshell." October 2020. [Online]. Available: https://www.autosar.org/fileadmin/ABOUT/AUTOSAR_EXP_Introduction102020.pdf

[21] G. Andersson and T. Guild, "Common Vehicle Interface Initiative: A standards-based approach to vehicle data & services," April 2021. [Online]. Available: https://at.projects.genivi.org/wiki/display/DIRO/Common+Vehicle+Interface+Initiative+--+Home?preview=/63799455/70878272/CVII%20First%20Introductions%20v3.pdf

[22] "About Automotive Grade Linux," 2016. [Online]. Available: https://www.automotivelinux.org/about/

[23] C. Ruiz and W. Robinson, "Measuring open source quality: A literature review," *International Journal of Open Source Software and Processes*, vol. 3, no. 3, pp. 48–65, 2011.

[24] J. Margono and T. E. Rhoads, "Software reuse in the Air Traffic Control Advanced Automation System," in *Proceedings of the Software Reuse and Reengineering Conference*, 1991.

[25] J. Favaro, "What price reusability? A case study," in *Proceedings of the first international symposium on Environments and tools for Ada*, 1991, pp. 115–124.

[26] R. Joos, "Software reuse at Motorola," *IEEE Software*, vol. 11, no. 5, pp. 42–47, sep 1994. [Online]. Available: http://ieeexplore.ieee.org/document/311058/

[27] W. C. Lim, "Effects of reuse on quality, productivity, and economics," *IEEE Software*, vol. 11, no. 5, 1994.

[28] A. Lynex and P. J. Layzell, "Understanding resistance to software reuse," *Proceedings of the International Workshop on Software Technology and Engineering Practice, STEP*, pp. 339–349, 1997.

[29] R. Sells, "Ieee aerospace conference, big sky, montana, 12 march 2020," *Montana*, 2020.

[30] D. Landau, K. Martin, T. Minkoff, P. Landon, and B. Gray, "Julia language 1.1 ephemeris reader and gravitational modeling program for solar system bodies," in AIAA *Scitech 2020 Forum.* Pasadena, CA: Jet Propulsion Laboratory, National Aeronautics and Space . . . , 2020.

[31] M. Wilhelm and M. Stuber, "EAGO.jl: easy advanced global optimization in Julia," *Optimization Methods and Software*, pp. 1–26, 2020.

[32] F. Oz and K. Kara, "A CFD tutorial in Julia: Introduction to laminar boundary-layer theory," *Fluids*, vol. 6, no. 6, p. 207, 2021.

[33] "COSSI: $100M+ revenue commercial open-source software (COSS) company index." [Online]. Available: https://docs.google.com/spreadsheets/d/17nKMpi_Dh5slCqzLSFBoWMxNvWiwt2R-t4e_l7LPLhU/edit?usp=sharing

[34] J. M. Pearce, "Emerging business models for open source hardware," *Journal of Open Hardware*, vol. 1, no. 1, pp. 1–14, 2017.

[35] J. P. Johnson, "Open source software: Private provision of a public good," *Journal of Economics and Management Strategy*, vol. 11, no. 4, pp. 637–662, 2002.

[36] E. Ostrom, *Governing the commons: The evolution of institutions for collective action.* Cambridge University Press, 1990.

[37] B. Perens, "The emerging economic paradigm of open source," *First Monday*, no. 2, 2005. [Online]. Available: https://journals.uic.edu/ojs/index.php/fm/article/download/1470/1385#a13

[38] S. Weber, "The political economy of open source software," 2000.

[39] J. Woods, "An engineer's history of U.S. and multilateral export controls and their application to the modern space industry," 2021.

[40] C. Cox, N. Dicks, P. Goss, D. Bereuter, J. V. Hansen, J. M. J. Spratt, C. Weldon, L. Roybal-Allard, and B. Scott, *Report of the Select Committee on* US *National Security and Military/Commercial Concerns with the People's Republic of China.* U.S. Government Printing Office, May 1999, vol. Vol. 2.

[41] C. Marquis, "Satellite maker fined $20 million in China trade secrets case," *New York Times*, January 2002. [Online]. Available: https://www.nytimes.com/2002/01/10/world/satellite-maker-fined-20-million-in-china-trade-secrets-case.html

[42] A. Pasztor, "Boeing, Hughes settle case over satellite technology," *Wall Street Journal*, March 2003. [Online]. Available: https://www.wsj.com/articles/SB1046893957378720160

[43] A. F. R. Lab, "Defense Industrial Base Assessment: U.S. Space Industry," Department of Defense, Tech. Rep., aug 2007.

[44] S. Navigation, "Libswiftnav: line 575," 2021. [Online]. Available: https://github.com/swift-nav/libswiftnav/blob/59ca6b59e41924fd096756baabb87911a91ebe10/src/single_epoch_solver.c#L575

[45] O. R. Institute, "ITAR and EAR strategy," 2020. [Online]. Available: https://www.openresearch.institute/itar-and-ear-strategy/

[46] Linux Foundation, "Understanding open source technology & U.S. export controls," Linux Foundation, Tech. Rep., July 2020. [Online]. Available: https://www.linuxfoundation.org/wp-content/uploads/UnderstandingOpenSourceTechnologyandUSExportControls_Whitepaper_070220-1.pdf

[47] S. A. Squassoni, "Weapons of mass destruction: Trade between North Korea and Pakistan," Congressional Research Service, Tech. Rep., November 2006. [Online]. Available: https://fas.org/sgp/crs/nuke/RL31900.pdf

[48] E. Albert, "North Korea's military capabilities," 2020. [Online]. Available: https://www.cfr.org/backgrounder/north-koreas-military-capabilities

[49] K. Davenport, "The Missile Technology Control Regime at a glance," 2021. [Online]. Available: https://www.armscontrol.org/factsheets/mtcr

[50] D. Messier, "NASA selects Astrobotic for two SBIR awards," *Parabolic Arc*, June 2019. [Online]. Available: http://www.parabolicarc.com/2019/06/22/nasa-selects-astrobotic-sbir-awards/

[51] C. Owens, K. Macdonald, J. Hardy, R. Lindsay, M. Redfield, M. Bloom, E. Bailey, Y. Cheng, D. Clouse, C. Y. Villalpando *et al.*, "Development of a signature-based terrain relative navigation system for precision landing," in *AIAA Scitech 2021 Forum*, 2021, p. 0376.

[52] E. Amoroso, H. Jones, N. Otten, D. Wettergreen, and W. Whittaker, "Quatitative evaluation of a planetary renderer for terrain relative navigation," in *Annual Meeting of the Lunar Exploration Analysis Group*, 2016.

[53] S. Ord, J. Kelly, C. Baker, and J. Thomson, "Tech Flights 2020 — Q & A sessions," March 2020. [Online]. Available: https://www.nasa.gov/sites/default/files/atoms/files/dr-fop-pre-10_tech_flights_2020_qna_session_slides.pdf

[54] A. J. Stevens, "The enactment of Bayh–Dole," *Journal of Technology Transfer*, vol. 29, pp. 93–99, 2004.

[55] P. Lee, "Contracting to preserve open science: Lessons for a microbial research commons," in *Designing the Microbial Research Commons: Proceedings of an International Workshop*, P. F. Uhlir, Ed. Washington, D.C.: National Academies Press, 2011, pp. 69–75.