

# **Code, Community, and Creativity in Processing**

Navigating the Complexities  
of Collective Contribution  
and Sustainability

**Tibor Udvari**  
**Advised by**  
**Prof. Nicolas Nova, Ph.D.**

**December 2023**

**Media Design Department,**  
**Geneva School of Art and Design**  
**(HEAD – Genève),**

**University of Applied Sciences and**  
**Arts Western Switzerland (HES-SO)**

# Abstract

This thesis examines the establishment and evolution of the Processing community, focusing on the interplay of software development practices and contributor motivations within this open-source ecosystem. Since its 2001 inception, the community, comprising artists, designers, and programmers, has been explored through interviews, archival research, and a detailed analysis of the software development cycle. The study employs Wenger's Communities of Practice framework to contextualize the community's formation, particularly highlighting the role of a user-friendly interface, mutual engagement, and situated learning. Furthermore, it scrutinizes the software development practices through the lens of the Cathedral and the Bazaar model, shedding light on its open-source methodology and the diverse motivations of contributors across economic, social, and technological dimensions. While acknowledging the transformations the community experienced with growth, the thesis primarily concentrates on the factors driving its initial formation and sustainability. The insights gleaned offer valuable implications for developing and maintaining other open-source creative tools.

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
1.1	The Legacy of Processing . . . . .	5
1.2	Research Questions . . . . .	6
1.3	Information Sources . . . . .	8
1.4	Structure . . . . .	11
<b>2</b>	<b>Language</b>	<b>12</b>
2.1	The Origins . . . . .	13
2.2	Values . . . . .	17
2.3	Design Implications . . . . .	19
<b>3</b>	<b>Software</b>	<b>22</b>
3.1	Software Development Models . . . . .	23
3.2	Core Contributions . . . . .	24
3.3	The Ecosystem of Libraries . . . . .	35
<b>4</b>	<b>Community</b>	<b>42</b>
4.1	A Community of Practice . . . . .	43
4.2	Forum Contributors . . . . .	47
4.3	Community Dynamics . . . . .	49
<b>5</b>	<b>Conclusion</b>	<b>56</b>
<b>6</b>	<b>Bibliography</b>	<b>59</b>
<b>7</b>	<b>Appendix</b>	<b>64</b>
7.1	Triangle Example Code . . . . .	65

# 1 Introduction



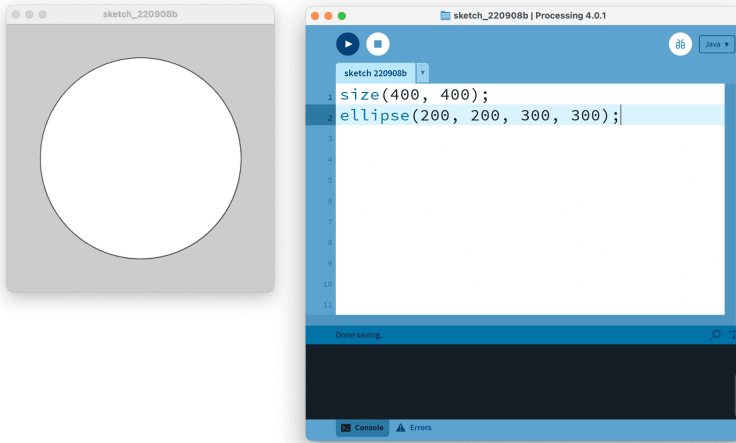


Figure 1: Processing IDE (Reas & Fry, 2015)

## 1.1 The Legacy of Processing

Processing is an open-source sketchbook and toolkit tailored for the electronic arts, new media art, and visual design communities, emphasizing the teaching of computer programming fundamentals within a visual context. Since its initial release on August 9, 2001 (Processing Foundation, 2022), Processing has had a lasting impact on computational design and creative coding.

This platform has not only persisted for over two decades but has also seen an expansion in its user base, indicating its continued relevance in a rapidly evolving field. Its role in shaping software literacy and education, especially for beginners, is well-documented, with its straightforward approach to coding and design principles that resonate with Maeda's laws of simplicity (Maeda, 2006).

The platform's design ethos of "Low Threshold, High Ceiling and Wide

Walls” principles, has facilitated a nurturing environment for learning and creation (Resnick et al., 2005). As indicated by user downloads that align with academic calendars (Fry & Reas, 2018), the steady increase in the adoption of Processing underscores its integration into educational structures. These metrics, alongside community surveys, suggest that Processing has transcended its educational utility, finding a place in artists’, designers’, and developers’ professional workflows (Processing Foundation, 2016).

Processing’s contribution to the field is further underlined by its influence on other significant projects, such as Wiring or Arduino, the micro-controller equivalent of Processing, which has roots in the same ethos and community (Barragán, 2016). Its interface adaptability is evident through its evolution into variants like p5.js, Processing for Android, and Processing for Python, suggesting a responsive growth to the needs of its community, as visible on the Processing website. (Processing Foundation, n.d.)

Given its historical context and the progressive adaptation to user needs, Processing is a distinguished example of enduring software in the computational design and creative coding community. This longevity and adaptability call for a closer examination of its design philosophy and community dynamics. As we witness the retirement of once-pivotal tools like Flash or Director, Processing’s sustained relevance raises questions about the factors contributing to its success. (Horton et al., 2020) (Jobs, 2010) (Adobe Creative Cloud Team, 2017).

However, despite its success and influence, a concerning aspect is the project’s sustainability, considering the reliance on a small number of volunteer developers for most of the codebase (Fry & Reas, 2018). This situation poses significant questions about the long-term viability of open-source projects that are driven by community rather than commercial support.

## 1.2 Research Questions

The preceding section contextualized Processing as an influential tool that has withstood the test of time, promulgating computational design and creative coding. The software project, however, does not exist in isolation,

as per Ben Fry's insightful statement :

“The processing project is a community, a piece of software that you run, and a language. And that order is important.” – Ben Fry (Arts at MIT, 2017, 19:22)

This thesis seeks to unearth the necessary dynamics that underpin the formation and perpetuation of such a rich ecosystem around open-source software such as Processing. As such, the investigation pivots around a central research question:

“What were the foundational dynamics that not only instigated but also sustained the collaborative spirit of the Processing community? Moreover, how did these dynamics, entwined with the intrinsic motivations of open-source contributors, influence the evolution of Processing's software development over the years?”

This question aims to dissect the symbiotic relationship between community involvement and software innovation, probing into how the motivations of volunteer contributors shaped the trajectory of Processing's growth. The thesis will examine:

- The initial factors that attracted contributors to Processing and the elements that fostered their long-term engagement, as indicated by Fry's prioritization of the community.
- The various forms of contributions made to the project, from coding to forum participation, reflecting the intertwined nature of the software and language.
- The transformation of these community dynamics from the nascent stages of Processing to its present-day stature.
- The underpinning reasons for continued voluntary participation in the project's development, despite the absence of direct financial incentives, aligning with Fry's understanding of the community's primacy.

Processing stands out from typical open-source software communities often studied because it is a Creativity Support Tool for the visual arts (Shneiderman, 2002). Its main aim is to enhance the creative abilities of its users. This purpose departs from the practical goals usually central to well-studied, large-scale open-source communities, like those involved in the Linux Kernel development, which are frequently examined through models like the Cathedral and the Bazaar (E. S. Raymond, 2002). The findings from this investigation will broaden our comprehension of how sustainable, collaborative, open-source software initiatives can be developed specifically for creative applications.

### 1.3 Information Sources

Given the potential for memory bias due to the passage of time since the project's inception, this thesis employs a mixed-methods approach. While human memory can be fallible, introducing biases and even constructing false memories, quantitative analysis serves as a foundational component to mitigate these challenges. It allows for identifying key contributors based on metrics such as software version control commit frequency, forum participation, and library contributions. These quantitative findings inform the selection of interview subjects, acting as a preliminary filter to locate contributors for qualitative interviews. By purposefully integrating quantitative methodologies with qualitative ethnographic approaches, this research aspires to offer a nuanced understanding of both the structural and phenomenological aspects of the Processing community.

The research draws upon multiple data sources to comprehensively understand the Processing community and its development practices—from forum discussions at various project phases to software version control logs and issue tracker reporting. The parsing status in Table 1 indicates the extent to which each data source has been prepared for analysis.

Interviewees were selected based on a strategic analysis of the collected data, especially in data-rich areas. Given the considerable quantity of data for the scope of the research project, particular attention was given to the project's origin, precisely the period of activity before and during the ac-

Name	Type	Status
Processing alpha forum	Forum	Parsed
Processing beta forum	Forum	Parsed
Processing 1.0 forum	Forum	Parsed
Processing 2.0 and 3.0 forum	Forum	Downloaded
Current processing forum	Forum	Downloaded
Github project	Commit history	Parsed
Processing Release Data	Release notes	Parsed
Github Release Data	Release notes & download statistics	Parsed
Processing libraries <sup>*</sup>	Software release information	Parsed
Bugzilla	Software issue tracker	Not downloaded
Github Issues	Software issue tracker	Not downloaded

Table 1: Overview of Data Sources Used for Analyzing the Processing Community

(\* The dataset is derived from an archival processing website and is incomplete.)

tivity of the initial alpha forum. The selection yielded three main categories of interview subjects:

- Code contributors to the core project during the activity period of the initial alpha forum, from August 2, 2002, to April 19, 2005.
- Active forum members during the same time frame.
- Active library authors with frequent software release activity spanning from October 26, 2011, to June 8, 2014, based on available data

The primary qualitative method utilized was semi-structured interviews. The questions focused on how the individuals got involved, interacted with the community, perceived community dynamics, and their motivations to contribute. This choice was driven by the need for a flexible yet organized approach to gather in-depth insights from participants.

The qualitative insights gathered through interviews are essential in this research, allowing for a deeper understanding of the personal motivations and experiences of the contributors. Table 2 presents the names and roles of specific interview subjects to provide a comprehensive view

Interviewee	Main Contribution	Date
Jacob Schwartz (benelek)	Alpha forum participant	2023-10-25
Ariel Malka (arielm)	Alpha forum participant	2023-10-25
Ricard Marxer (Geomerative)	Library contributor	2023-10-26
Simon Greenwold	Early contributor, ACG member	2023-10-27
Martin Gomez (MartinG)	Alpha forum participant	2023-10-30
Andreas Schlegel	ControlP5, oscP5 library contributor	2023-10-31
Karsten Schmidt	Code and library contributor	2023-11-07
Ben Fry	Processing co-founder	2023-11-14

Table 2: Contributor interviews

of the primary sources that inform the analysis. Each interviewee was selected based on their unique contributions and roles in the Processing community, offering diverse perspectives that underpin the analysis of the following chapters. These dialogues not only reflect the quantitative data but also provide nuances that numbers alone cannot capture, thus enriching the narrative of this thesis.

However, the richness of these qualitative accounts brings into focus the scope and boundaries of the study. One of the most significant limitations is that it only covers a specific time frame and certain aspects of the Processing ecosystem. For instance, the research does not comprehensively cover all the noteworthy contributors to the Processing ecosystem. The individuals engaged primarily in writing documentation, examples, and books, those participating in or organizing local Processing Community Days or other community events, Google Summer of Code participants and mentors, and recipients of fellowship programs are all meaningful contributors to the ecosystem that fall outside the scope of this writing. Moreover, the research did not go into detail regarding the role of the Processing Foundation, a non-profit organization established to oversee the development and distribution of the processing software, in organizing the community and ensuring the sustainability of the Processing project. Therefore, it is critical to note that while the study has presented significant findings, it is not a comprehensive analysis.

## 1.4 Structure

This thesis adopts a structured approach to exploring the Processing project, guided by Ben Fry's insightful quote as cited on page 7. However, for this analysis, the examination will proceed in the reverse order of Fry's listing—beginning with the language, moving to the software, and concluding with the community. This intentional inversion allows for a foundational understanding of the technical aspects before delving into the broader social and communal implications.

The first section focuses on the Processing language. It will delve into the syntax, structure, and unique aspects that make Processing an accessible yet powerful tool for creative coding. The evolution of the language from previous projects, such as *Design by Numbers*, will be examined and contextualized in the ecosystem of the tools of the time.

Following the exploration of the language, the thesis will address the software development aspect of Processing. This section aims to unpack how the software was built and maintained by the community.

The final section is dedicated to the community surrounding Processing. Here, the focus is on the growth, culture, and impact of the community around Processing. This section will investigate how community interactions, contributions, and collaborations have played a crucial role in shaping the project and expanding its reach and influence in computational design and creative coding.

By examining these components in this particular order, the thesis intends to build a layered understanding of Processing, from its technical core to its expansive community impact. This approach aims to provide a holistic view of the Processing project, showcasing how its technical aspects feed into and are influenced by its vibrant and dynamic community.

## 2 Language



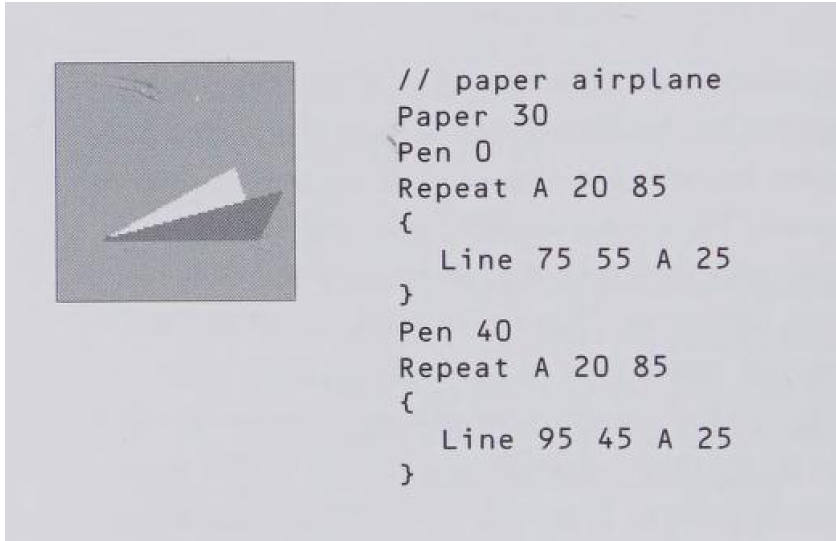


Figure 2: Design By Numbers example (Maeda, 2001, p. 66)

## 2.1 The Origins

The conceptualization of Processing is intricately linked to the environment at the MIT Media Lab's Aesthetics + Computation Group (ACG). Under the leadership of John Maeda, the ACG embraced a philosophy that dismantled traditional academic divisions between technological and artistic disciplines. Maeda articulated this vision: "Hybrids that can fluidly cross the chasm between technology and the arts are mutations in the academic system...During the 1990s the mutants that managed to defy this norm would either seek me out, or else I would reach out to find them myself. Bringing these unique people together was my primary passion, and that's how I came into contact with Casey Reas and Ben Fry" (Reas & Fry, 2007).

The necessity for Processing emerged from a critical assessment of the limitations inherent in computational design tools at the time. Systems

like OpenGL required extensive boilerplate code for rudimentary tasks, a process detailed in the appendix, impeding the fluidity of visual experimentation. This challenge highlighted the need for a platform conducive to the iterative processes typical within artistic practices. Design by Numbers (DBN)(Maeda, 2001), an earlier project led by Maeda and taught by Reas and Fry, presented an accessible introduction to programming concepts. Nevertheless, its scope was limited—operating within a monochrome color scheme and constrained to a 100x100 pixel canvas. These constraints, along with the limited command set of DBN (illustrated in figure 2), were reminiscent of Seymour Papert’s work with LOGO and its associated turtle graphics. The turtle graphics system, represented in figure 4, was similarly characterized by a limited set of commands geared towards educational purposes.

This realization of DBN’s limitations was particularly pronounced during Reas and Fry’s workshops with students at the Rhode Island School of Design (RISD), where the need for a more expressive and flexible tool was evident(Fry & Reas, 2018). The response to this need was Fry’s creation of Bagel, the 2D rendering engine that laid the groundwork for Processing’s graphic capabilities that would offer an array of graphic primitives, such as `arc()`, `ellipse()`, `line()`, `point()`, `quad()`, `rect()`, and `triangle()`, and represented a significant departure from the more complex programming requirements of Java or Lingo, as delineated in Table 3.

Fry’s experience with the advanced graphics framework ACU, inspirations from Flash and Director, and developing and teaching DBN culminated in the creation of Processing. This evolution was a response to the pedagogical and practical challenges faced in teaching computational design, aiming to expand beyond DBN’s black-and-white, pixel-restricted canvas to a platform that could accommodate the breadth and depth of artistic expression.



Figure 3: Physical turtle (Papert, 1980, p. ii)

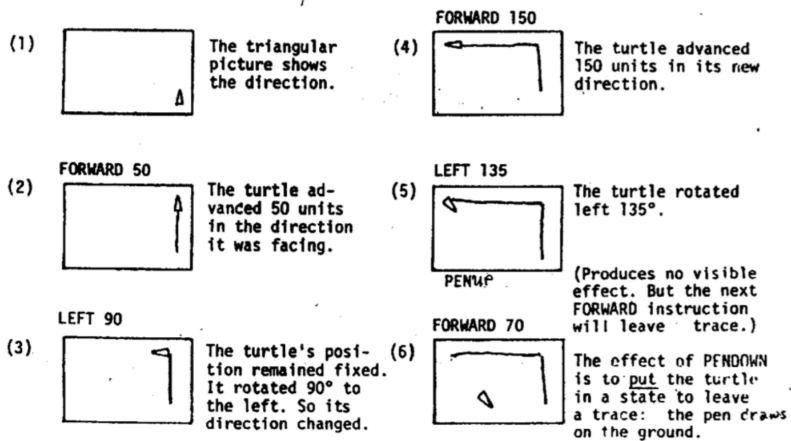


Figure 4: Turtle commands (Solomon et al., 2020, p. 827)



Figure 5: Ben Fry at Processing Community Day 2018

## 2.2 Values

The success of the Processing language is evidenced by its proliferation into various flavors such as Processing for Python, Android, and p5.js, each adaptation carrying the core philosophy into new contexts and platforms. This diversification attests to the robustness and adaptability of the language, reflecting its foundational values of accessibility, diversity, and inclusivity in the realm of creative coding.

These guiding principles originate from “Processing: A Programming Handbook for Visual Designers and Artists” (Reas & Fry, 2007) which lays out the philosophical and operational framework of the Processing project. The handbook serves as a manifesto detailing how Processing is not just a programming language but a pedagogical tool designed to bridge the gap between code and visual art. It emphasizes the unique qualities of

Processing	Java
<code>background(0);</code>	<code>g.setColor(Color.black); fillRect(0, 0, size.width, size.height);</code>
<code>background(255);</code>	<code>g.setColor(Color.white); fillRect(0, 0, size.width, size.height);</code>
<code>background(255, 204, 0);</code>	<code>g.setColor(new Color(255, 204, 0)); fillRect(0, 0, size.width, size.height);</code>
<code>stroke(255);</code>	<code>g.setColor(Color.white)</code>
<code>stroke(0);</code>	<code>g.setColor(Color.black)</code>
<code>stroke(255, 204, 0);</code>	<code>g.setColor(new Color(255, 204, 0));</code>
<code>fill(0, 102, 153);</code>	<code>g.setColor(new Color(0, 102, 153));</code>
<code>point(30, 20);</code>	<code>g.drawLine(30, 20, 30, 20);</code>
<code>line(0, 20, 80, 20);</code>	<code>g.drawLine(0, 20, 80, 20);</code>
<code>rect(10, 20, 30, 30);</code>	<code>g.fillRect(10, 20, 30, 30); g.drawRect(10, 20, 30, 30);</code>

Table 3: Comparison of Processing and Java graphics commands

software as a medium, advocating for a broadening understanding of programming languages as varied materials suited for different creative expressions.

The integration of sketching into the development process via Processing reflects a deliberate choice to align programming practices with those of traditional arts. This design decision underscores a commitment to making programming an intuitive and immediate experience, akin to the fluidity and exploratory nature of sketching in artistic disciplines.

By actively seeking to dismantle the barriers that have traditionally made programming an esoteric and often inaccessible field, Processing has opened its doors to a boarder audience. It champions the idea that programming should be within reach of anyone interested in visual exploration, not just those with a technical pedigree.



Figure 6: Chronotext project by Ariel Malka

### 2.3 Design Implications

The inception of Processing has significantly democratized the field of creative coding, with its workshop-centric educational approach enabling rapid assimilation by novices. This educational framework, spanning brief afternoon sessions to extensive week-long courses, has been pivotal in achieving the project's intent to simplify the initial learning curve for beginners.

Nonetheless, the early tool has posed challenges for experienced users, particularly in 3D graphics. For instance, Karsten Schmidt mentioned in the context of a project that “doing fullscreen in Processing was not possible back then. It needed to be written in C and just using OpenGL, which we didn’t have at that point. Although we could do 3D graphics in those initial versions of Processing, it was all just a software renderer”. Similarly, Ariel Malka transitioned from Processing to OpenGL due to the 3D limitations at the time. His project Chronotext 6 needed hardware performance, which Processing could not provide.

On the other hand, Ben Fry reflected on the unexpected adherence

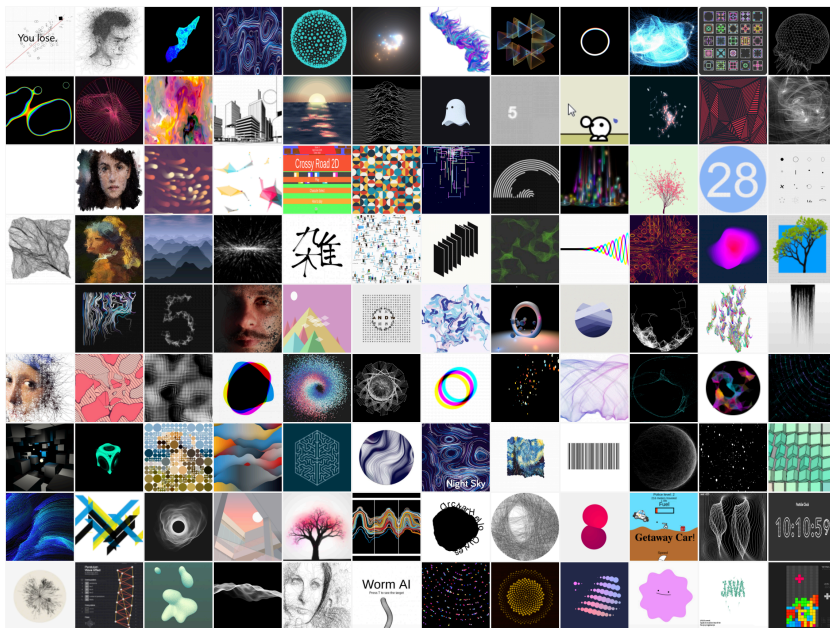
of beginner users to Processing. Even as projects expand in complexity, many users remain within the tool's confines, managing expansive code-bases within its singular file structure. Fry said, "We wanted to keep the environment so simple that it would work for single file sketches, and after that, it was time for people to move to a full IDE. But instead of moving, we saw a lot of people simply putting thousands of lines of code into that single file!"

This observation raises intriguing questions about beginner users' attachment to the tool and its consequences when more advanced tools and IDEs may be more appropriate. Similarly, the ease of use of specific graphical primitives over others can inadvertently steer users towards a specific aesthetic expression, which may promote a normative creative ethos.

Li poignantly describes this phenomenon: "Normative ground established by a tool not only affects how users can practically accomplish things, but also structures how they think and react. This represents a trade-off by granting tool designers power over creative practitioners." (Li, August-2023, p. 65). Thinking on the same Schmidt reflects: "I think every tool has that to some extent, but the tool you use on a daily basis becomes the lens through which you see the world".

While the obfuscation of the unnecessary elements in the graphics pipeline offers tremendous benefits for quick uptake in novices, it can make it more difficult to transition to other, perhaps more challenging, tools to use, which can offer greater latitude in artistic expression.





## 3 Software



Figure 8: 2019 p5js contributor conference

### 3.1 Software Development Models

The Processing project is commonly perceived as mainly the Integrated Development Environment (IDE) and its core components, which are crucial for basic operations. However, external libraries augmenting the system's capabilities are also essential to the ecosystem.

This chapter delves into the distinctive mix of software development as applied to these two foundational aspects, from the core project, relying on a few dedicated contributors to the decentralized bazaar of library contributions and their interplay. Afterward, the mixed motivations of these different groups of code contributors are analyzed through a lens of technological, social, and economic factors.

Eric S. Raymond's "The Cathedral and the Bazaar" (E. Raymond, 1999) identifies two open-source software development methodologies relevant to the processing project.

The Cathedral model, exemplified by early GNU projects under Richard Stallman, is characterized by meticulous planning and centralized control (Stallman & Stallman, 2002). This model functions much like the construction of a cathedral, where a small group of experts crafts a complex, well-organized structure over a long period, similar to the tiny number of contributors to the core processing project.

In contrast, the Bazaar model, which gained prominence with the development of the Linux Kernel espouses a more decentralized approach. Spearheaded by Linus Torvalds, this model resembles a bazaar, where contributors from diverse backgrounds bring in their unique contributions, leading to rapid iterations and an evolving software landscape reflective of the library's ecosystem. These two models represent endpoints on a continuum, with many real-world projects, including Processing, displaying characteristics of both.

While the Cathedral and Bazaar models provide a structural understanding of software development practices, exploring why individuals engage in these projects is equally essential. The sustainability of open-source projects like Processing hinges on the continuous flow of software contributions. Without the active and sustained involvement of contributors, the long-term viability of such projects is at risk. To gain a deeper understanding of what drives these contributions, we turn to the established taxonomy by Bonaccorsi et al. as seen in Table 4 (Bonaccorsi & Rossi, 2006) to facilitate comparison with other open-source projects.

This framework categorizes the motivations behind open-source contributions into economic, social, and technological domains.

The following sections analyze the dynamics and motivations of contributions to the core and libraries, respectively. The analysis combines quantitative data from software releases and version control systems with insights from interviews with key code contributors.

### 3.2 Core Contributions

The development environment plays a vital role in the processing project, setting the foundation for all functionalities. Contributions to the main code of the project are essential to the ecosystem, and without them, the

Motivation area	Micro level
Economic	Monetary rewards Low opportunity costs Gaining a reputation among peers Gaining future career benefits
Social	Fun to program (Loving to code) Altruism (gift economy) Sense of belonging to the community Fight against proprietary software
Technological	Learning Contributions and feedback from the community Working with a bleeding-edge technology Scratching a personal itch

Table 4: Taxonomy of individual programmers' Motivations. Adapted from (Bonaccorsi & Rossi, 2006)

other elements cannot function properly. It is essential to understand the motivations of the people who built and contributed to the project to analyze it effectively.

The analysis draws on public release logs and version control histories but with limitations. Key early project details are missing, as initial documentation practices were suboptimal. Multiple version control system changes likely caused data inconsistencies. While the Processing project's GitHub logs offer extensive data, they only partially capture the earliest contributions. The complexity of the early 2000s software release, exemplified by the mini CD distribution in October 2002 (Figure 9), contrasts with today's streamlined Git updates.

Fry indicated that early version control system limitations led him to manually commit code from other contributors, ensuring recognition in the logs despite lacking direct attribution in the repository's interface. This practice initially over-represented Fry's contributions, owing to challenges in proper crediting using CVS and later Subversion. "While things were in

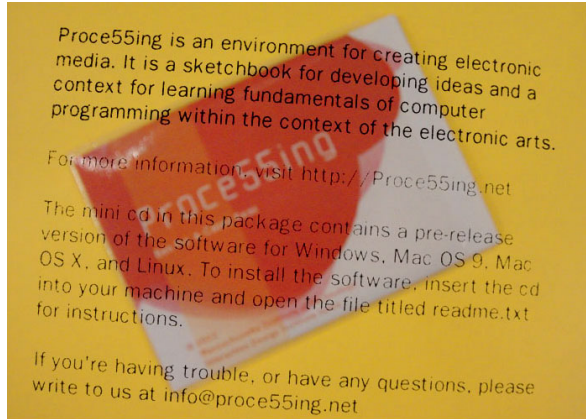


Figure 9: Processing Mini CD, October 2002

CVS, I had to do the commits myself, but always noted where things came from in the commit history”, Fry stated, noting a slight improvement with Subversion and Google Code. He emphasized, “GitHub was the first meaningful change where it really helped with the collaborative side of development”.

Following this, interviews were conducted with key figures: Ben Fry, the lead software engineer; Karsten Schmidt, a contributing coder; and Simon Greenwold, a core contributor and member of MIT’s Aesthetics and Computation Group during the project’s inception. These individuals were chosen based on their contributions, as evidenced in the Alpha Forum’s active period (Figure 12).

Ben Fry’s role as the leading technical engineer was significant in developing the Processing project. The observations of his collaborators underscore this. Casey Reas emphasizes Fry’s primary role: ‘I think one thing that’s important to clarify is that Ben Fry, my collaborator, is the primary software engineer of the project’ (Conrad et al., 2021, p. 330). Simon Greenwold further corroborates this, highlighting Fry’s authoritative approach: “He’s maintaining strong control, like it’s his code base. I think that’s part of why it’s great”.

Version Control

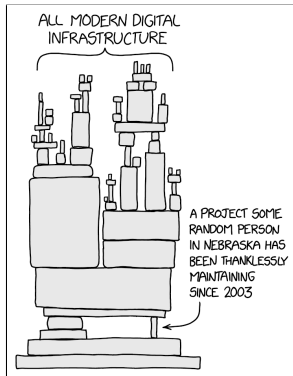


Figure 10: Dependency comic

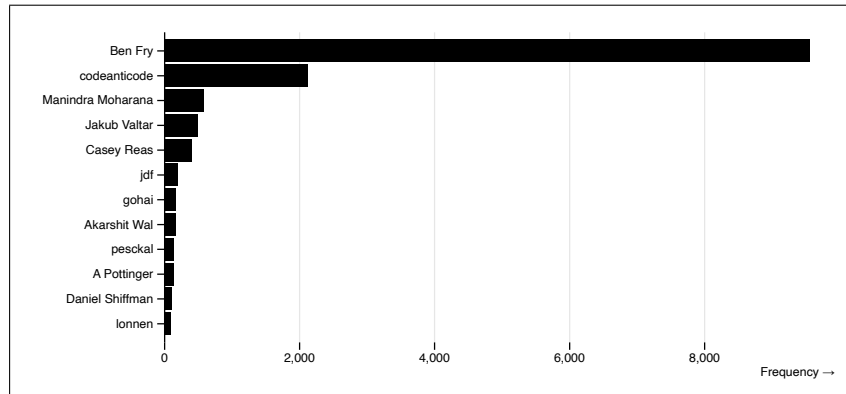


Figure 11: Top 12 source code contributors by number of commits in October 2023

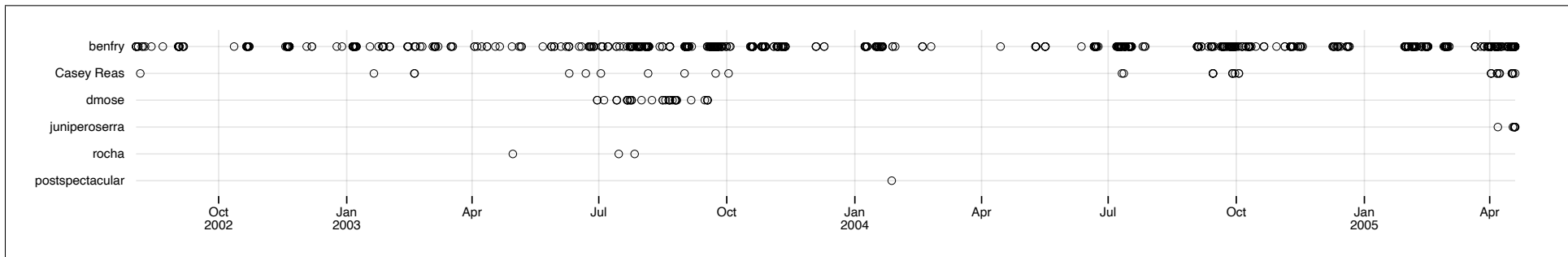


Figure 12: Updates before and during Processing forum's alpha stage

The version control statistics portray a significant concentration of contributions by Ben Fry, indicating his pivotal role in maintaining the project across its timeline. The bar chart demonstrates Fry's preeminence through the sheer volume of his contributions relative to others. However, it's important to recognize

that such visualizations do not fully encapsulate the breadth of contributions, especially in the project's nascent stages. Interviews suggest that many early contributions, made before the advent of sophisticated version control systems, remain unrepresented in these statistics. Thus, while the data underscores Fry's cen-

tral involvement, it also omits a spectrum of foundational efforts that were integral to the project's initial development.



Completing the development of Bagel, the initial render engine, Ben Fry had established a robust initial code base a year before the debut of the Processing Alpha forum involving other contributors. As Fry recounts, this foundational phase was critical but had challenges. He faced significant hurdles with MIT's Technology Licensing Office (TLO), specifically regarding intellectual property rights and the public release of the code. These challenges, stemming from MIT and the Media Lab's evolving stance on open-source software, resulted in prolonged delays and a waiting period for necessary clearances. These early struggles with IP rights and public code release paved the way for Processing's distinctive approach to Creativity Sustaining Tools (CST) for visual arts (Shneiderman, 2002).

Simon Greenwold underscores this transition, noting Processing's early adoption of an open-source model, a notable deviation in its field. He states, "Processing was notable in being open source early, ... probably unique in its class", thereby highlighting its pioneering role. In contrast, contemporaneous toolkits like Director or Flash were predominantly closed-source.

Processing's development initially adhered to a centralized, cathedral-style model characterized by structured, top-down control. The introduction of the alpha forum, however, marked a significant shift towards a decentralized, bazaar-style approach, encouraging community participation and collaborative code sharing.

This transition in Processing's development mirrors the early stages of other open-source projects like Linux, which also started with a solid foundation built by a single developer and evolved through community contributions. This shift to a more inclusive and collaborative model is a common trend in developing open-source software.

The impact of this shift was evident in specific revision logs. For instance, the release notes from 27/05/2003 - rev 55 marked a turning point with the incorporation of code from external contributors, reflecting a move towards a community-driven development model. As Fry points out in these notes, "This is the first release to include explicit support for audio, video, and network. Perhaps more exciting is that this code was developed by developers other than me". Following this, rev 56 further started to showcase a diversity of community involvement, not limited to code

contributions; members engaged in various activities, including testing, providing feedback, and enhancing documentation. This broadened participation exemplified the multifaceted nature of the bazaar model, where contributions extend beyond programming to encompass various aspects of software development.

Building on this community-centric approach, the project adhered to Eric S. Raymond's principle: 'Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging' (E. Raymond, 1999, p. 27). This ethos was crucial in Processing's evolution, particularly in the role of Ben Fry, who was integral in reviewing and integrating these diverse contributions, a strategy vital for the project's sustained success and functionality (Fry, 2022).

As the community's aspirations grew, a library system was introduced to accommodate the diverse needs of the project, a topic explored in greater depth in the following section.

Eric S. Raymond's 'Release early, release often' philosophy further propelled Processing's vibrant development (E. Raymond, 1999, p. 28). This approach, which greatly benefited Linux, enabled the rapid incorporation of contributions and fostered a responsive environment. The early phase of Processing, with 162 revisions prior to version 1 as shown in Figure 14 and Table 5, was characterized by frequent updates, echoing Raymond's observation about Linux's development: 'In those early times (around 1991) it was not unknown for Linus Torvalds to release a new kernel more than once a day!' (E. Raymond, 1999, p. 28).

However, Processing's development was occasionally constrained by its nature as a side project. A comment from a revision in January 2003 highlights this: 'hopefully January 2003 will be a good month for p5, as I have a short bit of time to work on it [...] I hope to get a few revisions out this month so I can get back to my 'real' work.' This comment reveals Processing's status as a secondary commitment for its core contributors.

This aspect of Processing's development is detailed in a book titled *Graphic Design in the Post-Digital Age*, which notes that Processing began as a personal initiative, mainly developed during nights and weekends. The project's funding sources, including MIT's indirect support through Fry's graduate stipend and the Interaction Design Institute Ivrea (IDII)'s support through

Reas's salary, further underscore its ancillary status (Conrad et al., 2021, p. 396).

This part-time commitment likely impacted the level of engagement in the project. As Raymond puts it, regarding Linux: "Linus was keeping his hacker/users constantly stimulated and rewarded—stimulated by the prospect of having an ego-satisfying piece of the action, rewarded by the sight of constant (even daily) improvement in their work" (E. Raymond, 1999, p. 28). In contrast, Processing's intermittent development schedule may have limited its potential for maximal engagement.

This engagement disparity becomes more evident when examining the contribution landscape within the Processing project, as depicted in Figure 12. The figure highlights Ben Fry's central role, especially during the alpha forum phase. According to the available data, only six individuals, including Fry, appear to have actively contributed code to the repository, starkly contrasting the more than 1000 individuals engaged in forum discussions. Such a scenario suggests that Fry's involvement was central and potentially overshadowing, limiting the scope for broader community contributions.

As observed in many open-source initiatives, Fry's predominant role in Processing raises critical concerns about the project's resilience and sustainability. As corroborated by the subsequent visualizations <sup>11</sup> his contributions were essential to keep the project running all this time. Over the past two decades, his extensive contributions have been indispensable. However, they also introduce a high-risk factor known as the "bus factor" ("Bus Factor," 2023), reflecting the project's vulnerability due to its heavy reliance on a few key individuals. This phenomenon, while common across open-source projects and often humorously depicted in popular culture (Munroe, 2020), poses a significant challenge to the long-term health of Processing.

Release Frequency

Descriptive statistics	Time between releases
count	60
mean	20 days 15:11
std	47 days 00:10
min	0 days 00:00
25%	1 days 00:00
50%	3 days 00:00
75%	26 days 18:00
max	338 days 00:00

Table 5: Release statistics

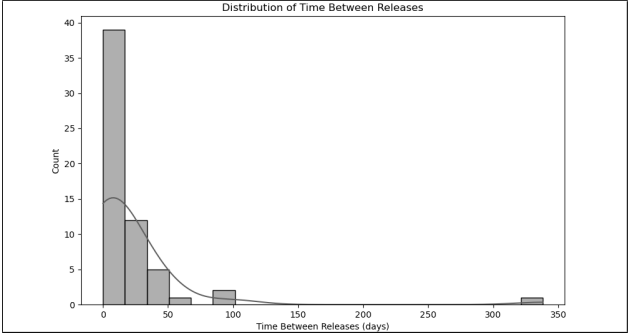


Figure 13: Frequency of releases

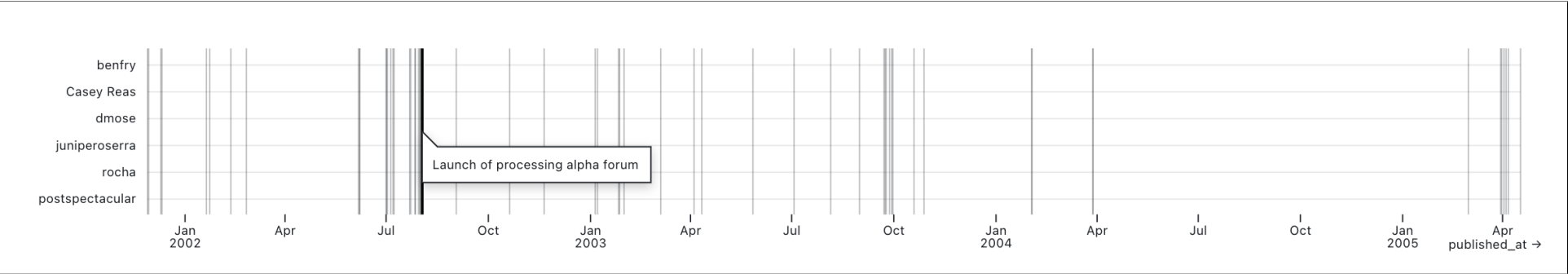


Figure 14: Time between releases

The graphical representations depict a distinctive pattern of release clustering, a testament to the sporadic and intermittent nature of the project's development lifecycle. A notable surge can be observed leading up to pivotal milestones, such as the alpha forum's initial release, indicating a focused burst of activity during these critical junctures. The distribution of releases further accentuates the project's non-linear progress, with a marked increase in releases as key points approach, followed by periods of relative inactivity. Moreover, the extended intervals devoid of any releases implies that the project advancement is a secondary endeavor for the contributors.

Transitioning from the observed discrepancy between the comprehensive forum engagement and the limited core code contributors, it is pivotal to contextualize the commitment of these few key individuals within a structured framework. The taxonomy by Bonaccorsi et al. Bonaccorsi and Rossi, 2006 provides a valuable lens through which to examine their motivations. This taxonomy categorizes the motivations into Economic, Social, and Technological factors, each offering a distinct perspective on the individual's involvement.

Ben Fry's technological motivation for developing Processing is deeply rooted in his personal experience with the cumbersome nature of existing tools for computational design. He aimed to create a platform that could sidestep these complexities, thus "scratching his own itch" and aid in teaching: "As researchers, all those extra steps just got in the way: we already knew how to do those things, and it just made the process tedious. So with Processing, we wanted to get closer to just having things show up and work".

This technological motivation is also echoed by Karsten Schmidt, whose initial involvement was propelled by the technological potential and the opportunities within the young Processing project: "The project was still young, there was a lot of stuff to do ... So I was not really a Processing user in the beginning. I became a user as I was contributing code to the actual tool".

The economic incentives for engaging with Processing were not clear; Simon Greenwold's statement highlights the absence of monetary gain and the low economic competition in the field: "There is no money in it. So who is going to come challenge it, right?".

Nevertheless, despite the lack of financial motivation, Processing found significant traction in educational settings, where the alternatives left much to be desired. Greenwold's assessment of the other tools available at the time underlines this point: "They were terrible. Just terrible. Yeah, Flash the director and Flash weren't the same yet. They came together and then both died".

The role of Processing in education is further substantiated by visual evidence, as illustrated in the community survey results conducted in 2016, which underscore its widespread adoption as a teaching tool (Figure 15).

The social aspect of Fry's motivations cannot be understated. His commitment to teaching and sharing knowledge, particularly with individuals who may not have coding expertise, signifies a strong social incentive that aligns with the open-source ethos of knowledge dissemination: "One of the most important things about the community for me personally is the motivation that comes from seeing really talented people use tools like this. That's really exciting and makes me want to work harder", furthermore he states: "I learned to code at a very young age because other people shared their code... Their willingness to share, or even answer questions of a 9- or 10- or 12-year-old was a pretty incredible gift, and had a huge impact".

Fry's dedication to education and community engagement is a testament to the social motivations central to the Processing project. This commitment has fostered a supportive community and helped proliferate computational design skills among a broader audience, further amplifying the project's impact.

In sum, the core contributors to Processing, who already possess strong technical skills, have been driven by a confluence of motivations. Technologically, they aimed to create a tool that would overcome the limitations of existing software. Economically, while there was no direct financial incentive, the educational adoption of Processing has increased its use and, indirectly, its importance in the computational design landscape. Socially, the emphasis on teaching and sharing knowledge has been a cornerstone of the project's philosophy, ensuring that Processing remains an accessible and community-focused platform.

### 3.3 The Ecosystem of Libraries

The integration of libraries marked a significant turning point in the Processing ecosystem. These additions expanded the core platform's capabilities, thus drawing a more comprehensive array of users and contributors. In the Processing community, libraries have been indispensable, acting as fundamental components that enhance the platform's functionality and encourage inventive exploration.

This evolution in development philosophy is captured through the in-

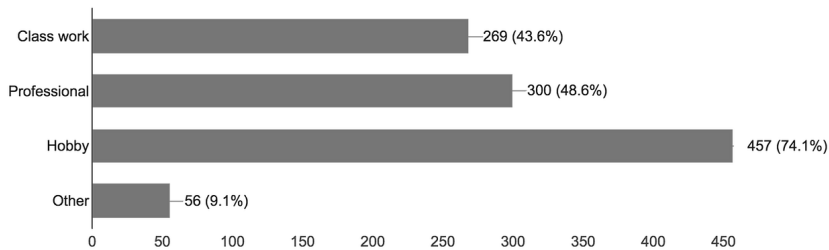


Figure 15: Processing 2016 community survey result (Processing Foundation, 2016)

sights of Casey Reas, a co-founder of Processing. Reflecting on the initial phases of Processing’s development, Reas observed, “One of the important early decisions was to make what we call libraries. Ben [Fry] and I realized that we were in a bottleneck and getting in the way of how people wanted to expand and extend Processing” (Conrad et al., 2021, p. 329). This statement underscores the pivotal shift towards an approach that embraced openness and collaborative progress in the development process.

The development and integration of libraries within the Processing ecosystem primarily exemplify the bazaar software development model, with a nod to the modularity characteristic of the GNU model. This is particularly evident in how some libraries evolved into core components of Processing.

A notable example is the development of the OpenGL renderer by Andrés Colubri. Initially developed independently to enhance Processing’s performance, this library was later integrated into the main software following its widespread acceptance within the community. This integration process is a classic example of the bazaar model, where development is open, collaborative, and evolutionary, driven by the diverse needs and contributions of the user base. (Fry, 2022)

While the modularity and individual contributions in this process resemble the GNU model, it is necessary to note that Processing’s approach reflects the dynamic and decentralized nature of the bazaar model more. The bazaar model emphasizes open collaboration and iterative development, aligning more with the community-driven development culture in



Processing.

Utilizing publicly available data from the Processing website archive, active libraries' release history was successfully reconstructed between October 2011 and June 2014. This period reflects active development and frequent releases across multiple library categories, as Figure 16 illustrates. As a result of the absence of earlier data, it led to a somewhat arbitrary choice of interview subjects from this period. Consequently, the insights obtained may not fully represent the experiences within the Processing community at its beginning. Despite these limitations, the interviews provided valuable qualitative insights into the motivations and contributions of library developers.

Library Releases

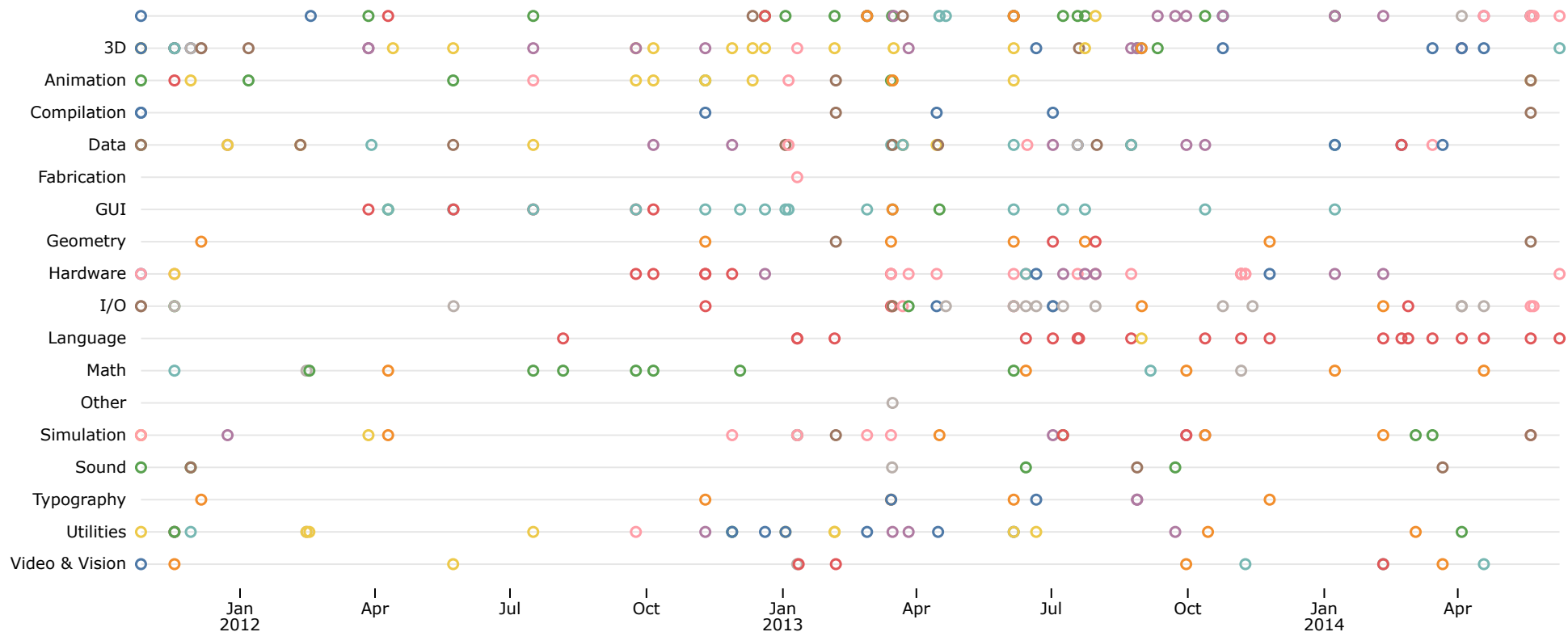


Figure 16: Distribution of Libraries in the Processing Project

This graph illustrates the distribution of Processing library releases over a period from January 2012 to April 2014. Each dot on the graph signifies a release event for a library in various categories such as 3D, Animation, and GUI. The process of releasing these libraries was manual, requiring approval on the Processing website before they became available for update. This artisanal method meant that only the latest versions of the libraries were accessible, without an option to retrieve

previous iterations. The data for this graph was meticulously reconstructed using a 'git blame' operation on the data folder containing links to these libraries. It's worth noting that this data may not fully represent the entirety of the Processing library space. The reconstruction is based on a single version of the website's data, lacking a comprehensive historical record that could have been obtained from multiple website versions. Thus, while the graph provides valuable insights

into the release patterns and activity within the Processing community, it comes with the caveat of being an incomplete representation due to the availability of only one snapshot of the website data. Despite the limitations in data scope, it reflects a significant portion of the community's contributions and the evolution of library offerings during the specified timeframe.

When analyzing interviews guided by Bonaccorsi et al.'s taxonomy, it becomes clear that technological innovation and community engagement were the primary driving forces behind the contributors. For instance, Karsten Schmidt's experiences reflect a strong technological motivation. He aimed to push the limits of computational design's expressive potential and, as a result, was compelled to create tools such as *toxiclibs*. This library aligned more closely with his vision than other frameworks, such as *Flash* or *Director*, which constrained creative expression within predefined roles and stage-based metaphors like directors, actors, and scripts. He said, "I really wanted something new, the way I can be more expressive with the actual computational side, the algorithm side".

Similarly, Andreas Schlegel's development of *ControlP5* was driven by a practical need in his projects. He sought a tool that could provide a more intuitive and effective way of managing user interfaces within *Processing*. Schlegel's creation of *ControlP5* can be seen as 'scratching his own itch', addressing personal challenges that benefit the wider community, as he noted: "*ControlP5* started as a need for sliders and buttons to control the visuals I was working on".

Marxer emphasized the collective effort and the shared sense of purpose among diverse individuals involved in open-source projects. He pointed out that these collaborative environments provide a platform for individuals from various backgrounds to learn from each other, contribute to a larger cause, and collectively advance the field. The social dynamics of open-source projects are crucial, as Marxer noted: "So it's a sense of community. I think, in a sense, of working together as a society".

According to Schlegel, his first library, *oscP5*, resulted from his final year project in his master's in media arts and technology, which focused on designing a computer network for artistic applications. Initially, he did not intend to write it for the community, but that came as an afterthought. Similarly, Marxer reflected that *Geomerative* was initially a technological tool to "do letters that were joined by the serifs", but it ended up being a whole project in generative typography.

As contributors like Schmidt and Schlegel delved deeper into the technology, they also became integral members of a burgeoning community. The collaborative ethos of the *Processing* community is evident in Marxer's

statement: “The goal is that all the work you do, nobody else should go through that and do it”. He was a DIY enthusiast, and he enjoyed seeing how things could be done.

Schmidt also mentions that “open source was essentially amateur culture... there are labors of love for most people; it’s what’s done in their spare time or what comes out from some interesting work projects, what people want to share”. Although not the primary draw, economic factors emerge in the narrative, albeit as a secondary consideration. Schmidt’s indirect financial benefits from his open-source work underline this point: “in other times, I have indirectly made a living through my open-source work, but not directly from it”. Schlegel’s experience similarly reveals that while economic gain was not the goal, the recognition and opportunities that arose from his contributions had positive professional implications.

The interplay between these motivations underscores the complexity of open-source software development. The contributors are often initially attracted by the technological challenges. As they immerse themselves in the work, the community aspect becomes more significant, providing a supportive network for sharing and growth. This dynamic is particularly relevant when the necessary tools and infrastructure have not yet been built out at the beginning of a project.

## 4 Community

Processing 1.0 - ALPHA			
Forum name	Topics	Posts	Last post
<b>Discussion</b>			
<b>Community, Collaboration, Status</b> Introduce yourself, join the group of Processing developers, and read about the current status Moderator: REAS	77	531	Apr 19 <sup>th</sup> , 2005, 1:10am by fry
<b>Events, Publications, Propaganda</b> Processing related conferences, courses, workshops, concerts, articles, etc. Moderator: REAS	77	225	Apr 18 <sup>th</sup> , 2005, 6:32am by mflux
<b>General Processing Discussion</b> Other topics related to Processing Moderators: fry, REAS	297	1263	Apr 18 <sup>th</sup> , 2005, 10:28am by mflux
<b>Programming Questions &amp; Help</b>			
<b>Syntax</b> Questions about the Processing Language Moderators: fry, REAS	695	2780	Apr 18 <sup>th</sup> , 2005, 11:35pm by st33d
<b>Programs</b> Questions about a specific Processing program Moderators: fry, REAS	327	1400	Apr 18 <sup>th</sup> , 2005, 5:02pm by Fish-face
<b>Integration</b> Processing and other systems (Java, PHP, MAX, MySQL, etc) Moderators: fry, REAS	83	405	Apr 17 <sup>th</sup> , 2005, 2:20am by gli
<b>Topics &amp; Contributions</b>			
<b>Tools</b> Developing software (or parts of software) for building software Moderator: REAS	136	776	Apr 19 <sup>th</sup> , 2005, 1:33am by deltasounds

Figure 17: Screenshot of the Processing Alpha Forum

## 4.1 A Community of Practice

It is vital to acknowledge that software development encompasses more than just the technical aspects of the project. Understanding the interactions and dynamics within the community is just as crucial. The Processing community, for example, values various valuable ways to contribute beyond writing code, such as teaching, organizing events, and creating tutorials and documentation. Some contributors even consider end-users as contributors as they provide the reason and purpose for developing and teaching software (Processing Foundation, 2022, p. 41).

Wenger's Communities of Practice (Wenger, 1998) theoretical framework was employed in our analysis to understand these additional relationships further. This model is particularly relevant, given that the focus on education and the dynamics of the early Processing community por-

tray an informal learning organization. The early community succeeds in peer-to-peer interaction, collaborative discussions, and shared creative exploration. These characteristics align closely with the three prevalent fundamental elements of Wegner's model: mutual engagement, joint enterprise, and shared repertoire, but they also incorporate less discussed concepts like legitimate peripheral participation.

Mutual Engagement refers to the collaborative interaction and relationships within the community, where members actively participate, share knowledge, and support each other. Joint Enterprise is the collective goal or endeavor that unites the community, providing a shared purpose and direction for its activities. Shared Repertoire encompasses the range of communal resources - tools, language, stories, and practices - developed over time, which embody the community's collective knowledge and cultural identity. These elements foster a dynamic, collaborative environment that promotes learning and knowledge sharing.

The forthcoming analysis seeks to delineate and interpret the identified elements, employing the Communities of Practice framework as a guiding lens. This methodological choice is advantageous, given the diverse identified dynamics and relationships. Furthermore, the application of this model not only facilitates an analysis of the evolution of these elements over time but can yield insights into the broader implications of community-led software development and learning.

At the beginning of the Processing project, the alpha forum was the community's central means of exchange. As asserted by Reas, the forum cultivated a 'unified international community' from 2002 to 2006 (Conrad et al., 2021, p. 331).

As such, the forum was subjected to data scraping and analysis to identify the most engaged members based on their posting activity. The three most active participants, apart from the project's co-creators, Ariel Malka, Martin Gomez, and Jacob Schwartz, were interviewed.

The chapter begins with an analysis of the integration of this particular group into the community, followed by a broader examination of community dynamics in general.



Forum Contributors

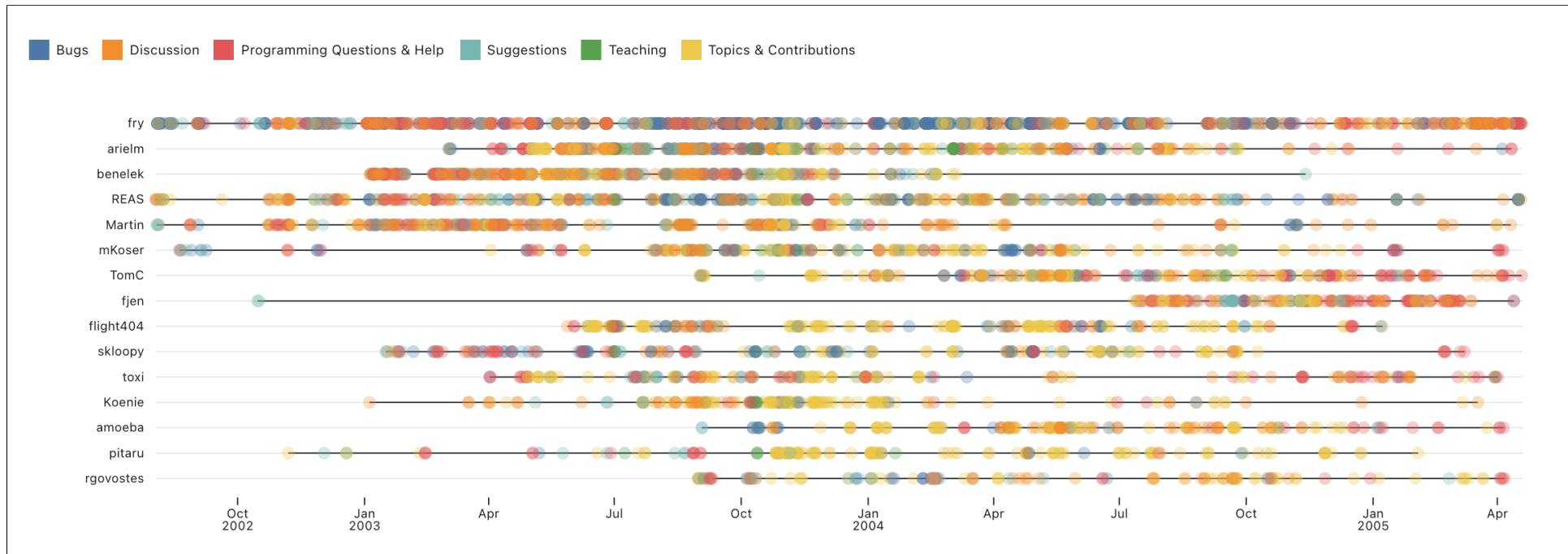


Figure 18: Posting activity of the 15 most active contributors on the alpha forum.

Figure 18 shows a timeline of color coded post by category of the 15 most active contributors to the Alpha forum. Among them Ariel Malka (arielm), Jacok Schwartz (benelek) and Martin Gomez (Martin) were interviewed, on the basis of their participation in the forums, with no definitive indications of them assuming additional roles such as library contributor. We can ob-

serve certain patterns emerging such as initial posts being often in the Programing Help & Questions section, often spanning out to others in the future. A part from co-creators Ben Fry (fry) and Casey Reas (REAS) we can also observe clustering of posts for users, suggesting periods of activity and inactivity.

Forum name	Years	URL
Processing alpha forum	2002-2005	<a href="http://forum.processing.org/alpha">forum.processing.org/alpha</a>
Processing beta forum	2005-2010	<a href="http://forum.processing.org/beta">forum.processing.org/beta</a>
Processing 1.0 forum	2010-2013	<a href="http://forum.processing.org/one">forum.processing.org/one</a>
Processing 2.0 and 3.0 forum	2013-2018	<a href="http://forum.processing.org/two">forum.processing.org/two</a>
Current processing forum	2018 - now	<a href="http://discourse.processing.org">discourse.processing.org</a>

Table 6: Archival forums composition

## 4.2 Forum Contributors

Within the Processing forums, the evolution from newcomer to integral community member exemplifies Wenger’s notion of legitimate peripheral participation (Wenger, 1998, p. 117). This concept describes how individuals initially engage at the community’s margins and, through increased involvement, move toward a more involved role. Ariel Malka, Martin Gomez, and Jacob Schwartz are case studies of this journey as they navigated from the outskirts to significant presence, carving out their professional identities without local counterparts in computational design.

Initially, these individuals engaged primarily in observing and learning from the exchanges of others. Interviewees mentioned reading and getting inspired by various technical discussions, such as how to simulate physical hair models by other users, such as Karsten Schmidt, or being impressed by the work of other forum participants, such as Glen Murphy. This observational phase is crucial in legitimate peripheral participation, allowing individuals to acclimate to the community norms, language, and practices.

Figure 18 shows the transitional phase as participants move from observers to active forum contributors. The friendly atmosphere of the forums has facilitated this transition, as we can observe the participants shifting their focus from “Programming Questions and Help”-related topics to more vast topics. This positive environment is unlike the often confrontational atmospheres of other online forums, which can be daunting for new posters. Gomez found the processing forums to be a conducive

environment for learning, where experienced users were always willing to help and point new users in the right direction, as he said: “In the processing forums, I never encountered things like that. They would always assume that you are there to learn and (...) they would usually just point you in the proper direction and be able to help you out”.

Schwartz’s words also capture the essence of evolving involvement: “I got a lot out of posting what I was working on and then getting feedback from that and looking at what other people were working on and seeing how they did those things”. This reflection highlights his transition from passive observation to active contribution and underscores the reciprocal nature of learning within the community. By sharing his work, receiving feedback, and drawing inspiration from others, Schwartz moved from the periphery of the community towards a more central, contributory role.

Similarly, Ariel Malka and Martin Gomez’s narratives reflect this transformative community engagement. Malka got involved more profoundly and transitioned from basic forum participation to contributions, such as participating in community exhibitions and testing software. Gomez, who felt isolated in his home country’s computational design domain, found a vital platform for development and collaboration in the Processing forum, moving from a solo enthusiast to an active community contributor. He pointed out that he was later able to create the first exhibitions in the Philippines relating to computational design.

These narratives collectively illustrate how online platforms, like the Processing forums, serve as critical enablers for legitimate peripheral participation. They provide a conduit for individuals without local communities to engage, learn, and contribute meaningfully.

The experiences of Malka, Gomez, and Schwartz not only shed light on the practical application of this concept but also underscore the importance of digital communities in nurturing individual growth, fostering skill development, and establishing a sense of belonging within specialized fields.

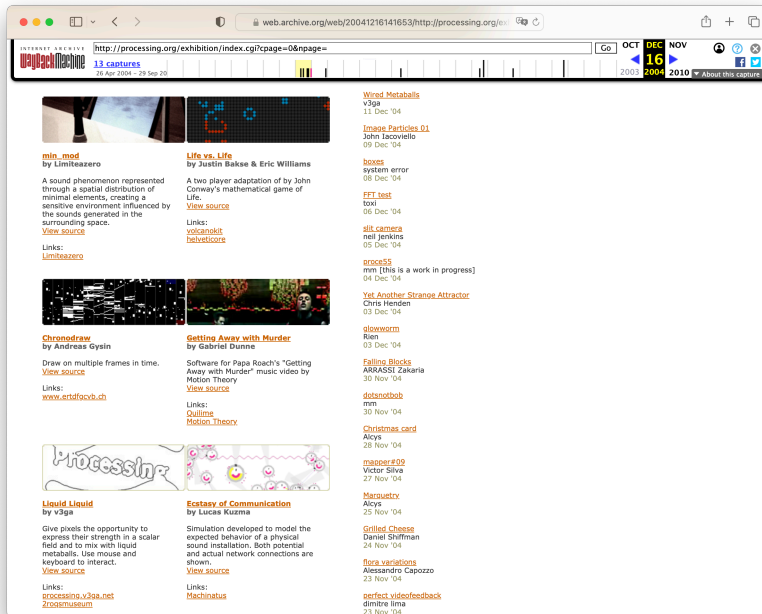


Figure 19: Screenshot of community exhibition

### 4.3 Community Dynamics

The Processing distinguishes itself by appreciating commonly overlooked “community maintenance” activities such as creating examples, writing documentation, and organizing exhibitions. As Wenger points out, these are intrinsic parts of the practice but are usually much less visible than instrumental parts, in the case of Processing releasing a new version of the IDE, for instance. (Wenger, 1998, p. 78)

Through the proce5sing.net website, the community has strongly emphasized sharing concise prototypical code examples that illustrate a feature or concept. This trait was already present in the first version of the

website, which only included an index, examples, and reference pages. By establishing these norms and best practices early on, newcomers could embrace and propagate these shared standards, contributing to the shared repertoire of the group.

While the example projects showcased the basics, the exhibition page featured more significant works. The co-founders of Processing, Ben Fry and Casey Reas shared their work, *Valence* and *Tissue*, which set a standard and expectation for others to follow. Other community members were encouraged to participate, resulting in similar works of similar scope. The source code for all of these works was made available, making the exhibition a showcase of projects and a learning opportunity for the community.

These methods create a low barrier to entry through examples and a high ceiling by sharing more consequential works. As noted by Fry, the project showcases would attract new members to the forums: “With each person who created something there, we would see an uptick of people participating in the forum who were followers of that person’s work”.

The project saw increased user engagement thanks to its features facilitating the onboarding of new users and community members. The website presentation effectively showcased the shared repertoire, which encouraged users to become active participants in the community of practice.

These efforts were exploratory. Schmidt points out that the initial community shared a common goal of exploration. “In the beginning, it was a small group, and everyone was learning together what this thing is, or what it could do”, he notes, highlighting the joint enterprise of the whole community. The software’s development can be seen as an artifact of this joint enterprise. It embodied the community’s learnings from experimentation as a whole. It featured an evolutionary process influenced by the development of libraries, which were the outcomes of earlier experiments.

The term “processing” itself evokes computational and artistic processes involving iterating through variations. In this community, it is primarily associated with software sketching, but it can be applied to the community as a whole. People in the forums discuss various topics such as Information Visualization, Simulation, Artificial Life, and more. The community’s

direction, shaped through these vibrant discussions and experiments, resonates with Wenger's CoP principles. As Schmidt pointed out, people at the time were thinking, "Perhaps the tool can be modified in the future to make these things more possible". This reflects the CoP's core aspect of dynamic meaning negotiation. This continual dialogue and experimentation within the community are emblematic of the learning and communal development emphasized in Wenger's framework.

Moreover, the community's engagement extends beyond technical aspects to philosophical debates, further illustrating the CoP's concept of shared meaning creation. Discussions in the forums, debating the intersections of code, design, and art, exemplify this. Topics such as "code != art" or "Processing = Design? Art? Other?" generate substantial interaction and highlight the community's diverse perspectives on project priorities. These debates, sometimes questioning the project's inclusivity towards artists and designers, underscore the ongoing process of negotiating meaning and priorities, a fundamental element in Wenger's CoP framework.

The richness of the community's engagement was only possible with the mutual engagement of forum participants and the formation of relationships between them. Interviewees often mentioned other users from whom they learned, as Jacob Schwartz notes, for example, "I still remember some of these usernames here. I was so impressed with Glen Murphy's projects".

It was common for experienced individuals like Murphy to share their work-in-progress posts, even on complex subjects such as Fluid Dynamics. For instance, on November 5th, 2002, he shared two code examples and the message "Neither of these are 'full' fluid simulations. For example, Navier-Stokes is nowhere to be seen, and the pressure calculations are pretty raw. May be useful as a base for something". This highlights the importance of giving back, inviting feedback and conversation, and integrating the learning process. This approach contrasts with solely sharing work to impress peers or showcasing finished products only.

Reflecting on his experience, Schwartz emphasized the significance of the community in addition to the tool, stating, "I would say that the tool itself was very interesting to me, but the community was a huge part of it

for me as well". This perspective emphasizes the value placed on the initial community, implying a collective responsibility for the community.

Schmidt builds on this notion, focusing on the community aspect of open-source projects. He articulates the implicit social contract involved in using open-source tools, which entails actively participating in the community. Schmidt explains, "When you sign up to use open source tooling, you make a kind of social agreement to be part of that culture, which gives you so much that there should be a social expectation also to want to give something back".

Marxer adds to this topic by highlighting the responsibility he felt towards users of his library. Reflecting on user interactions, he notes, "Every time that somebody would use it or would point out maybe some mistake, you said, 'Oh, then people are relying on this.' So you try to fix the mistake and put it into the next version". This perspective emphasizes the dynamic relationship between users and community participants, where feedback leads to continuous improvement and engagement.

The combination of community dynamics has created a positive feedback loop that engages users. The exhibition's potential draws them in, and they quickly start experimenting with different examples. They actively participate in the forum and gradually become community members by making diverse contributions. Measurable data such as forum activity and software usage statistics confirm this success. According to the Processing Foundation, over 250,000 unique users utilize Processing monthly.

As the project grew and matured, there was a noticeable shift in the community's dynamics and user perceptions. Schmidt pointed out that the Processing community has undergone a cultural shift where contributions are now sometimes viewed more as products rather than collaborative efforts. This indicates a disconnection from the shared repertoire and mutual engagement that once surrounded the original alpha forum. Compared to before, typical users integrate less into the ecosystem.

There is a viewpoint that the project has matured, and due to its widespread adoption, community participation in developing the tool is no longer considered necessary. As the project's primary goal has shifted towards educating novices in computer literacy, the joint exploration of computa-



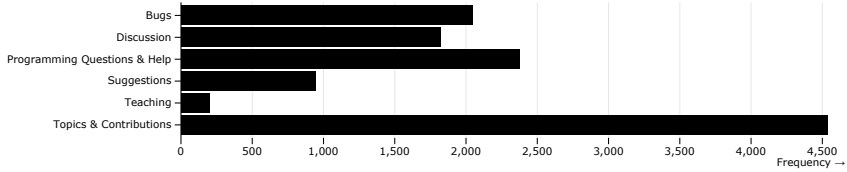


Figure 20: Topics by post number

tional design needs to be re-addressed. As a result, the tool's usage has increased, but the dynamics have shifted away from being a community of practice, especially in the official discussion forums.

This shift from a general interest in computational design towards a more tool-centric approach is reflected in the current organization of the processing forum, which does not prioritize main categories such as automated systems, tangible computing, or artificial life. Instead, the forum is organized based on software flavors, including p5.js, Processing, Processing for Android, etc.

In sum, the early community exemplifies Etienne Wenger's principles of mutual engagement, joint enterprise, and shared repertoire in a community of practice. This group, driven by a unified enthusiasm for computational design, created a collaborative environment where members exchanged knowledge, experiences, and resources. Their concerted efforts to develop and enhance the platform underscored a collective goal, fostering technical skill advancement and philosophical discussions. This interaction resulted in a comprehensive repository of communal assets, signifying a successful, cooperative, and productive community. Although the dynamics of this project have evolved, its foundational success and growth can be attributed to these elements.

## Community Connectedness

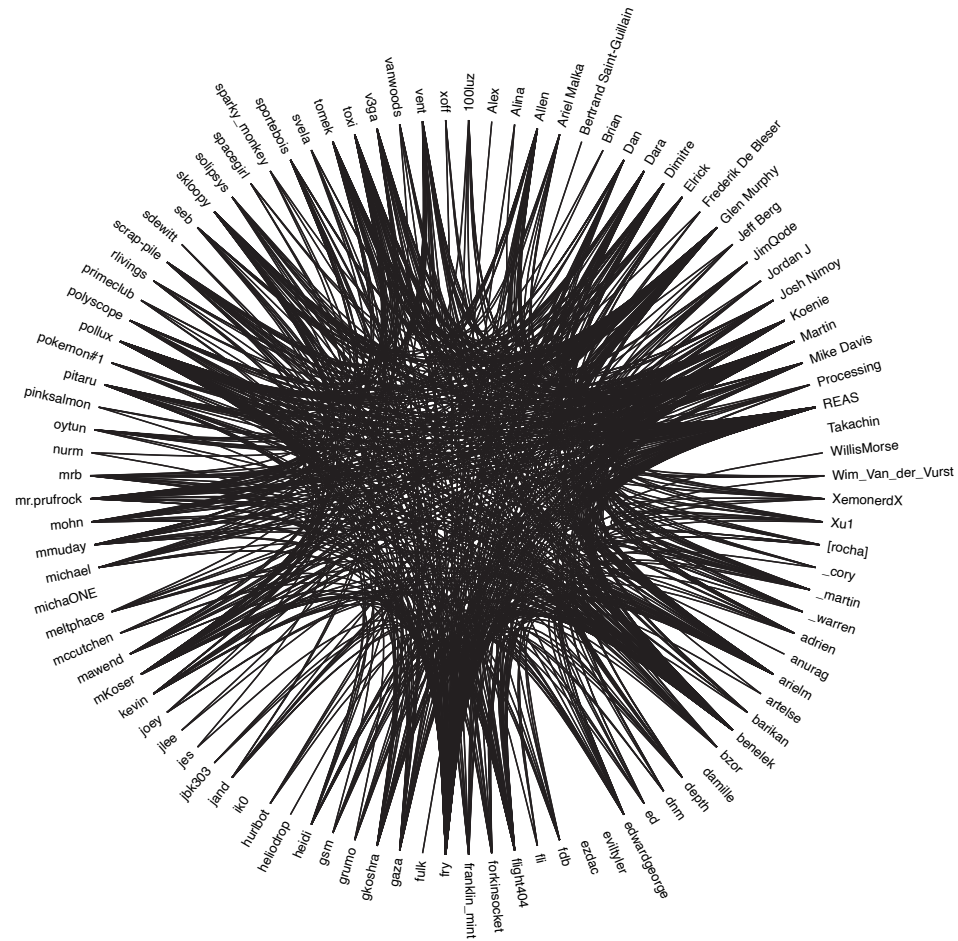


Figure 21: Connectedness through forum threads of top 100 alpha forum contributors

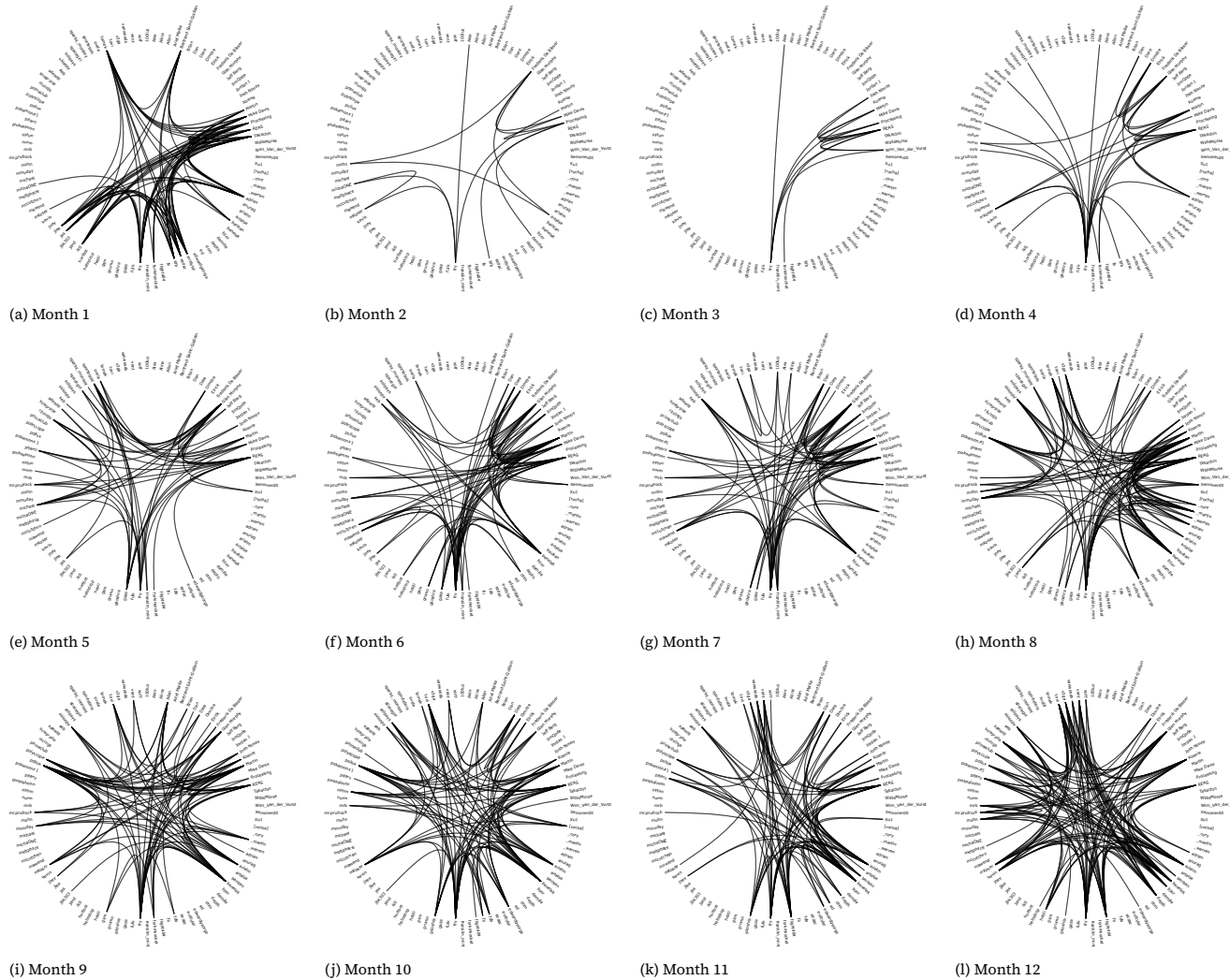


Figure 22: First year of the Alpha forum

The Alpha forum's developmental trajectory over its first year, as visualized through network graphs, reveals a dynamic growth narrative. Initially, a compact hub of activity signals robust engagement from a committed group of participants, laying the groundwork for the community's expansion. This is evidenced by an increase in the density and reach of interactions, suggesting a broadening in the diversity of contributors and a rise in active membership. The persistent complexity of the network and uniform distribution of connections throughout the year highlight an environment where individuals are actively integrated, promoting a collective and inclusive exchange of knowledge.

The absence of isolated clusters within these visualizations speaks to a forum culture that successfully assimilates newcomers into the ongoing dialogue, maintaining an equitable and collaborative discourse. This enduring interactivity, underscores the forum's efficacy as a thriving digital ecosystem conducive to sustained collaboration and exchange.

## 5 Conclusion

This thesis investigates the foundational dynamics that initiated and sustained the development of the Processing project. It identifies the key factor that stimulated the project as forming a strong community of practice around the alpha forum. While the motivations of individual contributors played a preliminary role, mutual community engagement provided a long-lasting social motivation. The research emphasizes distinct choices made in the project's design, such as the simplicity of the language, the modularity of the software library system, and particular community norms that fostered positive synergies. The above factors, together with adoption by educational institutions, helped create a dedicated community of Processing users, significantly increasing the platform's user base. However, this significant growth has hurt the peer-to-peer learning environment. Additionally, Ben Fry's unwavering commitment to software maintenance is praiseworthy, but it also raises concerns about Processing's sustainability in the long run, as it creates a dependency on him.

As a result of analyzing this thesis, several suggestions emerge for improving open-source creative tools. Firstly, it is recommended to have one or more centralized shared repositories with mechanisms in place to ensure that new and innovative creative applications are given prominence through peer review or curation by experts, such as on the exhibition page of the Processing website. Secondly, it is suggested to encourage the exploration of larger domains like computation, computer graphics, graphic design, and visual arts and communicate the results through shared tools and language. Experienced practitioners should have long-term mutual engagement outlets to facilitate learning while allowing new learners easy entry. Finally, community members should be encouraged and guided to contribute to the main software base to empower the community and encourage long-term sustainability.

The data collection, analysis code, and resulting dataset have been made public. Although missing some initial software development information, it offers a comprehensive parsable archive of the forums. As such, future research could include more quantitative methodologies using this data to highlight this research's conclusions better.

The future of Processing depends on contributions in coding, documentation, and a mutually engaged community. The thesis concludes by

inviting readers to consider their potential role in advancing Processing or similar communities, directing them to the contribution guidelines on its official repository.

In the spirit of the shared journey that defines Processing's legacy, there is a genuine expectation for the future. This anticipation is rooted in the personal stories and creative breakthroughs that Processing has enabled. As each new contribution adds to the vibrant community, it's not just about maintaining Processing as a tool, but celebrating it as a catalyst for individual creativity and global collaboration. These contributions, big and small, ensure that Processing can remain a deeply an indispensable resource for creators everywhere.

## 6 Bibliography

# References

- Adobe Creative Cloud Team. (2017, January 27). *The Future of Adobe Contribute, Director and Shockwave*. Retrieved November 15, 2023, from <https://web.archive.org/web/20190204065858/https://theblog.adobe.com/the-future-of-adobe-contribute-director-and-shockwave>
- Arts at MIT (typedirector). (2017, July 28). *2017 CAST Symposium BEING MATERIAL: Ben Fry and Casey Reas, PROGRAMMABLE* [Video]. Retrieved September 15, 2023, from <https://www.youtube.com/watch?v=9BtqBjGEpA0>
- Barragán, H. (2016). *The Untold History of Arduino*. The Untold History of Arduino. Retrieved November 15, 2023, from <https://arduinohistory.github.io/>
- Bonaccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. *Knowledge, Technology & Policy*, 18(4), 40–64. <https://doi.org/10.1007/s12130-006-1003-9>
- Bus factor. (2023, July 31). In *Wikipedia*. Retrieved September 15, 2023, from [https://en.wikipedia.org/w/index.php?title=Bus\\_factor&oldid=1168093480#cite\\_note-3](https://en.wikipedia.org/w/index.php?title=Bus_factor&oldid=1168093480#cite_note-3)  
Page Version ID: 1168093480.
- Conrad, D., Leijssen, R. van, & Hérítier, D. (Eds.). (2021). *Graphic design in the post-digital age: A survey of practices fueled by creative coding* (First edition). Onomatopée.
- Fry, B. (2022). *Processing Contribution Guide*. GitHub. Retrieved November 23, 2023, from <https://github.com/benfry/processing4/blob/main/CONTRIBUTING.md>
- Fry, B., & Reas, C. (2018, June 8). *A Modern Prometheus, The History of Processing by Casey Reas and Ben Fry*. Processing Foundation. Retrieved May 1, 2023, from <https://medium.com/processing-foundation/a-modern-prometheus-59aed94abe85>



- Horton, J. J., Sloan, M. I. T., Tambe, P., & Penn, U. (2020). The Death of a Technical Skill.
- Jobs, S. (2010, April). *Thoughts on Flash*. Retrieved November 15, 2023, from <https://web.archive.org/web/20170615060422/https://www.apple.com/hotnews/thoughts-on-flash/>
- Li, J. (August-2023). *Rethinking Power Dynamics in Software Tools for Artists*. <https://purl.stanford.edu/xf605mk2773>
- Maeda, J. (2001). *Design by numbers* (1. paperback ed). MIT Press.
- Maeda, J. (2006). *John Maeda: The Laws of Simplicity*. design manifestos .org. Retrieved October 25, 2023, from <https://designmanifestos.org/john-maeda-the-laws-of-simplicity/>
- Munroe, R. (2020, August 17). *Dependency*. xkcd. Retrieved September 15, 2023, from <https://xkcd.com/2347/>  
Licensed under CC BY-NC 2.5.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.
- Processing Foundation. (n.d.). *Processing Website*. Processing. Retrieved November 28, 2023, from <https://processing.org/>
- Processing Foundation. (2016). *2016 Community Survey*. Retrieved October 17, 2023, from <https://medium.com/processing-foundation/community-survey-5784e4ec74fc>
- Processing Foundation. (2022). *20th Anniversary Processing Community Catalog*. Retrieved October 8, 2023, from <http://archive.org/details/processing-community-catalog-2021>
- Raymond, E. (1999). The cathedral and the bazaar. *Knowledge, Technology & Policy*, 12(3), 23–49. <https://doi.org/10.1007/s12130-999-1026-0>
- Raymond, E. S. (2002). The cathedral and the bazaar: Musings on Linux and open source by an accidental revolutionary. *Choice Reviews Online*, 39(05), 39-2841-39-2841. <https://doi.org/10.5860/CHOICE.39-2841>
- Reas, C., & Fry, B. (2007). *Processing: A programming handbook for visual designers and artists*. MIT Press.
- Reas, C., & Fry, B. 2015 *Processing IDE*. Retrieved November 8, 2023, from [https://processing.org/static/d577f8358172ec82a2ac07d2c5ac1993/9344c/Fig\\_02\\_01.png](https://processing.org/static/d577f8358172ec82a2ac07d2c5ac1993/9344c/Fig_02_01.png)

- Resnick, M., Myers, B., Nakakoji, K., Shneiderman, B., Pausch, R., Selker, T., & Eisenberg, M. (2005). Design Principles for Tools to Support Creative Thinking.
- Shneiderman, B. (2002). Creativity support tools. *Communications of the ACM*, 45(10), 116–120. <https://doi.org/10.1145/570907.570945>
- Solomon, C., Harvey, B., Kahn, K., Lieberman, H., Miller, M. L., Minsky, M., Papert, A., & Silverman, B. (2020). History of Logo. *Proceedings of the ACM on Programming Languages*, 4, 1–66. <https://doi.org/10.1145/3386329>
- Stallman, R., & Stallman, R. M. (2002). *Free software, free society: Selected essays* (J. Gay, Ed.; 1st. ed). Free Software Foundation.
- Wenger, E. (1998, July 28). *Communities of Practice: Learning, Meaning, and Identity*. Higher Education from Cambridge University Press. <https://doi.org/10.1017/CBO9780511803932>

## List of Figures

1	Processing IDE . . . . .	5
2	Design By Numbers example . . . . .	13
3	Physical turtle . . . . .	15
4	Turtle commands . . . . .	16
5	Ben Fry at PCD 2018 . . . . .	17
6	Chronotext project by Ariel Malka . . . . .	19
7	Openprocessing screenshot . . . . .	21
8	2019 p5js contributor conference . . . . .	23
9	Mini CD . . . . .	26
10	Dependency comic . . . . .	28
11	Top source code contributors . . . . .	28
12	Early commits . . . . .	28
13	Frequency of releases . . . . .	33
14	Time between releases . . . . .	33
15	Processing 2016 community survey . . . . .	36
16	Distribution of Libraries in the Processing Project . . . . .	39

17	Screenshot of the Processing Alpha Forum . . . . .	43
18	Posting activity of the 15 most active contributors on the alpha forum. . . . .	46
19	Screenshot of community exhibition . . . . .	49
20	Topics by post number . . . . .	53
21	Connectedness through forum threads of top 100 alpha forum contributors . . . . .	54
22	First year of the Alpha forum . . . . .	55

## List of Tables

1	Data Sources . . . . .	9
2	Contributor interviews . . . . .	10
3	Processing Java comparison . . . . .	18
4	Motivations taxonomy . . . . .	25
5	Release statistics . . . . .	33
6	Archival forums composition . . . . .	47

## 7 Appendix

## 7.1 Triangle Example Code

Listing 1: Hello Triangle Example

```
/*  
  
    Copyright 2010 Etay Meiri  
  
    This program is free software: you can redistribute it and/or modify  
    it under the terms of the GNU General Public License as published by  
    the Free Software Foundation, either version 3 of the License, or  
    (at your option) any later version.  
  
    This program is distributed in the hope that it will be useful,  
    but WITHOUT ANY WARRANTY; without even the implied warranty of  
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the  
    GNU General Public License for more details.  
  
    You should have received a copy of the GNU General Public License  
    along with this program. If not, see <http://www.gnu.org/licenses/>.  
  
    Tutorial 03 - First triangle  
*/  
  
#include <stdio.h>  
#include <GL/glew.h>  
#include <GL/freeglut.h>  
#include "ogldev_math_3d.h"  
  
GLuint VBO;  
  
static void RenderSceneCB()  
{  
    glClear(GL_COLOR_BUFFER_BIT);  
  
    glBindBuffer(GL_ARRAY_BUFFER, VBO);  
  
    glEnableVertexAttribArray(0);  
  
    glVertexAttribPointer(0, 3, GL_FLOAT, GL_FALSE, 0, 0);  
  
    glDrawArrays(GL_TRIANGLES, 0, 3);  
  
    glDisableVertexAttribArray(0);  
  
    glutSwapBuffers();  
}  
  
static void CreateVertexBuffer()  
{
```

```

Vector3f Vertices[3];
Vertices[0] = Vector3f(-1.0f, -1.0f, 0.0f); // bottom left
Vertices[1] = Vector3f(1.0f, -1.0f, 0.0f); // bottom right
Vertices[2] = Vector3f(0.0f, 1.0f, 0.0f); // top

glGenBuffers(1, &VBO);
glBindBuffer(GL_ARRAY_BUFFER, VBO);
glBufferData(GL_ARRAY_BUFFER, sizeof(Vertices), Vertices, GL_STATIC_DRAW);
}

int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGBA | GLUT_DEPTH);
    int width = 1920;
    int height = 1080;
    glutInitWindowSize(width, height);

    int x = 200;
    int y = 100;
    glutInitWindowPosition(x, y);
    int win = glutCreateWindow("Tutorial_03");
    printf("window id: %d\n", win);

    // Must be done after glut is initialized!
    GLenum res = glewInit();
    if (res != GLEW_OK) {
        fprintf(stderr, "Error: '%s'\n", glewGetErrorString(res));
        return 1;
    }

    GLclampf Red = 0.0f, Green = 0.0f, Blue = 0.0f, Alpha = 0.0f;
    glClearColor(Red, Green, Blue, Alpha);

    CreateVertexBuffer();

    glutDisplayFunc(RenderSceneCB);

    glutMainLoop();

    return 0;
}

```