Government College of Engineering (GCOEJ), Jalgaon

(An Autonomous Institute of Government of Maharashtra)

DEPARTMENT OF COMPUTER ENGINEERING

INDUSTRIAL LECTURE REPORT ON GSOC AND OPEN SOURCE

(Academic Year 2023-24)

Submitted by:

Bhargav Shamuvel Gurav (2041009)

Government College of Engineering (GCOEJ), Jalgaon

(An Autonomous Institute of Govt. of Maharashtra)

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the ***Industrial Lecture*** report**, "GSOC and Open Source"**, which is being submitted here with for the award of ***LY Computer Engineering (7th Semester)*** is the result of the work completed by ***Bhargav Gurav (2041009)*** under my supervision and guidance within offline mode of classes of the institute, in the academic year 2023-24.

.

Head of Department

Dr. D. V. Chaudhari

INDEX

CONTENTS **Page no.**

ABSTRACT

Open source software development has been revolutionizing the tech industry, fostering collaboration, innovation, and accessibility. Google Summer of Code (GSoC) stands as a testament to this synergy, providing a platform for budding developers to contribute meaningfully to open-source projects. This report delves into the profound impact GSoC has had on the open-source ecosystem, analyzing its role in fostering mentorship, nurturing talent, and accelerating project growth.

The report investigates the significance of GSoC in empowering students worldwide, offering them invaluable hands-on experience while contributing to impactful open-source initiatives. It examines case studies and success stories, illustrating how GSoC initiatives have led to significant advancements within various projects. Furthermore, it explores the mutual benefits accrued by both mentors and participants, highlighting the mentorship culture that fosters skill development, community building, and the sustainability of open-source projects.

Additionally, the report evaluates the challenges faced by GSoC, such as diversity and inclusivity concerns, and proposes strategies to address these issues. It also discusses the evolving landscape of open-source software development and the role GSoC plays in adapting to these changes, including its impact on emerging technologies and trends.

In conclusion, this report serves as a comprehensive analysis of the transformative influence of Google Summer of Code on the open-source community. It underlines the program's role in cultivating a generation of skilled contributors, enhancing the vitality of open-source projects, and shaping the future of collaborative software development.

INTRODUCTION

Open source

What is open source?

Open-source software is software whose source code—the instructions that define what the software does—is published and freely available. The opposite of open source is closed source. Source code is human-readable, and software developers create software by writing it. To run software on a device, though, the source code has to be transformed into a form that's mostly unreadable to humans. That unreadable form is what you get when you download an app. Open-source software is often developed in a collaborative manner, and considered a public good, free for anybody to use. The Brave Browser, Linux operating system, and OpenOffice are examples of open-source software.

Why does it matter whether software is open source?

With a closed-source app, it takes very specific expertise—which even many professional software developers don't have—to be able to understand exactly what that app is doing. Even for people with that expertise, it takes a lot of time and effort to fully analyze a closed-source app. It's much easier to understand what an open-source app is doing.

That means it's easier to hide nefarious behavior, or even malware, in a closed-source app. An app may be stealing your private information, and if that app isn't open source, you'll probably never know.

Another benefit of open-source software is that the community at large can contribute to it in various ways. Other developers can contribute new features and bug fixes, or inspect for security problems. Anyone can copy and modify the software to suit their own needs.

What are some examples of open-source software?

The Brave browser is one good example. It follows a common practice in the open-source community, called "forking." This is when a project is started as a copy of another open-source project, rather than starting from scratch. Brave is based on Chromium, an open-source Web browser that also forms the basis of Chrome and Edge. Another open-source project is the Linux operating system, which powers a wide range of devices.

Today, open-source software is ubiquitous. Any device you use includes at least some open-source software, and the foundations of the Internet are made of open-source software.

Note: Technically Linux is something called an "OS kernel," not an operating system. But that nuance is beyond the scope of this article. For now, you can think of it as an operating system.

Why do people and companies give away their source code?

It may not be obvious why anyone would give away their source code, rather than making money from it. But there are several reasons to publish (or open) source code:

- To contribute back to the community. Lots of companies use other open-source software, including for business-critical purposes, and publishing some of their own source code can be a way of giving back in kind.
- To receive community contributions, such as bug reports and improvements.
- To let others adapt the code however they need to.

- To be transparent, and show that there’s no nefarious behavior being hidden. This is especially important for browser extensions.

For a company, open-sourcing can improve the company’s image, and even help recruit talent. For example, if a volunteer makes particularly good contributions to a company’s open-source project, the company may try to hire that person. And the ability to work on open-source projects may be a reason for a developer to want to join a company.

Companies can open-source their software and still make money from it, such as by selling support contracts for it. Individuals who maintain open-source software can also accept payment in return for prioritizing specific features or providing support.

GSOC (Google Summer of Code)

Google Summer of Code is a global, online mentoring program focused on introducing new contributors to open source software development. GSoC contributors work on a 12+ week programming project with the guidance of mentors from their open source organization.

Since 2005, the Google Summer of Code program has connected 19,000+ new open source contributors from 112 countries with 18,000+ mentors from 133 countries. Google Summer of Code has produced over 43 million lines of code for 800+ open source organizations.

During Google Summer of Code, participating contributors are paired with mentors from open source organizations, gaining exposure to real-world software development techniques. Contributors will learn from experienced open source developers while writing code for real-world projects! A small stipend is provided as an incentive.

Participating organizations use the program to identify and bring in new, excited developers. Many of those new developers will continue to contribute to their new communities and open source long after GSoC is over.

How It Works

Contributors

Potential GSoC contributors contact the mentor organizations they want to work with and write a project proposal based on ideas the organization has suggested. Once accepted, GSoC contributors spend a few weeks becoming familiar with the community norms and codebase while determining expected milestones with their mentor for the summer. GSoC contributors then spend 12+ weeks coding on their projects.

Organizations

Open source projects apply to be mentor organizations. Once accepted, organizations discuss possible ideas with contributors and choose the proposals they wish to mentor for the summer. They provide mentors to help guide each contributor through the program.

Mentors

Community members and committers already active in the mentoring organizations can choose to mentor a contributor project. Mentors and GSoC contributors work together to determine appropriate goals for the program period. Mentor interaction is a vital part of the program.

Full Program Timeline

Organization Application Period

Open source organizations can submit their applications to be mentor organizations for GSoC.

Organizations Announced

Potential GSoC contributors discuss project ideas with accepted mentor organizations.

Contributor Application Period

Potential GSoC contributors can register and submit their proposals to the mentor organizations that interest them.

Proposal Review Period

Organizations review and select contributor proposals.

Contributor Projects Announced

Accepted GSoC contributors are paired with a mentor and start planning their projects and milestones.

Community Bonding

GSoC contributors spend 3 weeks learning about their organization’s community and preparing for their coding project.

Coding Period

GSoC contributors work on their Google Summer of Code projects.

Evaluations

Mentors and GSoC contributors submit their evaluations of one another.

GSoC contributors Submit Code and Final Evaluations

GSoC contributors submit their code, project summaries, and final evaluations of their mentors.

Mentors Submit Final Evaluations

Mentors review their GSoC contributor code samples and determine if they should pass the Google Summer of Code Program.

Results Announced

GSoC contributors are notified of the pass/fail status of their Google Summer of Code projects.

MENTORSHIP AND LEARNING EXPERIENCE IN GSOC

What Makes a Good Mentor?

Mentoring a GSoC contributor can be a very rewarding experience. However, being a good mentor is *not* just a matter of winding up the GSoC contributor and watching them go. Quality mentoring requires a substantial time commitment and the willingness and ability to take a leadership role.

There are specific skills that you can work on in order to be more effective; even experienced mentors can improve. This chapter highlights some of the capabilities of top mentors, by suggesting some self-assessment questions that can help you to evaluate your strengths and weaknesses in this role.

Are you already part of the developer community? If you are not then you are going to be less effective at introducing a GSoC contributor to the local culture and practices. Similarly, you are less likely to be able to propose, guide and integrate successful projects relevant to the larger effort. If you are new to the community, working as a backup mentor on a project may be an excellent way to get involved. Note that some projects are more community-oriented than others; assess the community skillset needed for your target group.

Do you have a real interest in potential GSoC projects? As a GSoC mentor you will be taking ownership of a project idea and seeing it through the summer. If you are not excited about the project, mentoring will be more difficult. You are an integral part of the process from project proposal to delivery.

Are you willing to dedicate significant time? While the time requirements for mentoring vary, you should seriously consider your prior mentoring experiences and your available time before committing to this role. If you really don't want to mentor, or really won't have a reasonable amount of time each week, *then don't offer.*

Are you keenly interested in mentoring GSoC contributors? A main goal of GSoC is mentoring GSoC contributors. Mentoring is important to the future of open source software, our immediate projects and the overall culture. Mentoring a GSoC contributor requires a combination of passion, responsibility and patience. A good mentor is willing to engage with GSoC contributors throughout their learning process.

Be Prepared to Seek Help

At all times don't forget that you have access to people, tools and resources that can make your job much easier *and* make you a better mentor. Make use of your org admin when you are not sure what is expected of you or have a difficult situation with your GSoC contributor. Make use of other mentors in your organization and the thousands of mentors on the mentor mailing list. Though it may be an annoying list at times *(don't feed the trolls!)*, it is a valuable resource. The GSoC admins are another important resource. They set the tone and standards for the entire program. They have heard it all, so don't hesitate to contact them when a problem arises.

What To Expect From Under-Mentoring

The GSoC contributor's project is never properly defined. The project goals and deliverables are unclear, and the work schedule is not set. The consequences of this are serious and impact the project if left unchecked.

The mentor has little idea what the GSoC contributor is doing. The state of the project is unclear, and its progress is uncertain. Evaluation is impossible to do well.

The GSoC contributor produces code that isn't useful. The GSoC contributor starts off on 'the wrong path', failing to use existing functions or established project idioms. Rather than fixing problems as they arise they keep on adding more. The code is never integrated into the main codebase because it doesn't work well enough and would require more work to fix than it is worth.

The GSoC contributor gets stuck. The GSoC contributor seems to be engaged, and to be working hard, but no apparent progress is being made. Alternatively, the GSoC contributor's communications are infrequent and terse, and seem to always be on the same issue or milestone.

The GSoC contributor disappears, perhaps for days or weeks at a time. If the GSoC contributor is under-mentored, it may be difficult to determine when this period began, and thus to know when to panic. Insufficient information is available for evaluation; thus it becomes impossible to fairly evaluate the GSoC contributor.

CHALLENGES AND OPPORTUNITIES IN GSOC

1. In-Depth Research: To prepare for this transformative experience, I delved deep into the annals of the GSoC program. I scoured its history and studied previous projects meticulously, gaining invaluable insights into the program's expectations and the nature of the work involved.

2. Skills Augmentation: Recognizing that coding prowess is the bedrock of any GSoC endeavor, I honed my coding skills assiduously. I focused particularly on the programming languages relevant to the project ideas that piqued my interest.

3. Communication Excellence: As I understood early on, effective communication would be paramount. Collaborating seamlessly with mentors and the vibrant open-source community necessitated honing my communication skills.

Obstacles Faced:

Time as a Challenge: The GSoC journey brought forth the formidable challenge of balancing academic commitments, personal life, and the rigorous demands of the program. I found my footing by crafting a well-structured schedule that allowed me to allocate time judiciously.

Navigating Technical Hurdles: The projects I undertook were undeniably complex, and I encountered a multitude of technical obstacles along the way. Overcoming these hurdles was a testament to the GSoC spirit, with mentor guidance and community support acting as my guiding stars.

Scope Clarification: Defining the scope of my project and setting attainable milestones required meticulous planning. Aligning my goals with the expectations of my mentors proved instrumental in charting a successful course.

Tips for Freshers:
Early Beginnings: Commence your GSoC preparations well in advance. Dive into the realm of research, explore potential projects, and refine your skills diligently.

Open Source Engagement: Prioritize contributions to open-source projects before applying to GSoC. Such involvement serves as a testament to your dedication and skills.

The Proposal Artistry: Craft a proposal that is not only concise but brimming with clarity. Showcase your profound understanding of the project, articulate your approach, and spotlight your unique contributions.

Some opportunities that participants may have include:

- Exposure to open source projects
- Learning about open source culture and community
- Interacting with experienced developers
- Building a network of mentors and programmers
- Internship opportunities
- Stipends of up to US\$3,000

FUTURE OF OPEN SOURCE

The future of open source software is poised for an intriguing evolution, driven by several key trends and factors shaping its trajectory.

Collaboration and Interconnectivity:

Open source's future lies in enhanced collaboration. Projects are increasingly interdependent, with developers focusing on interoperability and compatibility. The rise of modular architectures and microservices fosters collaboration among diverse projects, allowing for seamless integration and faster innovation.

Sustainability and Community Growth:
Sustainability remains a critical concern. As open source projects mature, maintaining momentum and interest becomes crucial. Initiatives centered on fostering inclusive communities, nurturing new contributors, and ensuring diverse representation are pivotal for long-term sustainability.

AI and Automation Integration:
The integration of AI and automation within open source projects is a transformative trend. Tools that automate coding, testing, and bug fixing, leveraging machine learning, are revolutionizing development processes. AI-driven code suggestion systems and automated documentation generators streamline workflows, empowering developers.

Security and Compliance:
Enhanced security measures are paramount. With the proliferation of cyber threats, open source projects are intensifying efforts to fortify code security and ensure regulatory compliance. Improved vetting processes, automated security scans, and proactive vulnerability management are becoming integral parts of development cycles.

Ethical and Responsible AI:
As AI becomes more prevalent, ethical considerations are gaining prominence. Open source communities are actively engaging in discussions and developing frameworks to ensure responsible AI development. Initiatives focusing on transparency, fairness, and accountability in AI algorithms are emerging.

Decentralization and Web3:
The concept of Web3, characterized by decentralization and blockchain technology, is driving innovation in open source. Decentralized applications (dApps) and blockchain platforms heavily rely on open source principles. Communities are exploring ways to integrate decentralized protocols and technologies into open source projects.

Continued Corporate Involvement:
Corporate involvement in open source will persist and possibly expand. Major tech companies recognize the benefits of open source collaboration. However, balancing corporate interests with the ethos of open source will be crucial to maintaining the community-driven nature of projects.

CONCLUSION

The future of open source software is a landscape of innovation, collaboration, and responsibility. Its evolution involves harnessing emerging technologies, nurturing vibrant communities, fortifying security measures, and navigating ethical considerations. As open source continues to underpin technological advancements, its adaptability and commitment to collaboration will be pivotal in shaping the digital future.

REFERENCES

1. ChatGPT
2. GeeksForGeeks
3. Intellipat
4. JavatPoint