



EVOLUTION PATTERNS OF OPEN-SOURCE SOFTWARE SYSTEMS AND COMMUNITIES

(OSSシステムとコミュニティの進化パ
ターン)

NUR ATHIRAH
60226192

TABLE OF CONTENTS (目次)

AIM

(目的)

ROLES OF COMMUNITY MEMBER

(コミュニティメンバーの役割)

COMMUNITY STRUCTURE

(コミュニティ構造)

EVOLUTION OF SYSTEMS

(システムの進化)

3 TYPES OF OSS PROJECTS

(3種類OSSプロジェクト)

DISCUSSION

(考察)

CONCLUSION

(まとめ)



AIM

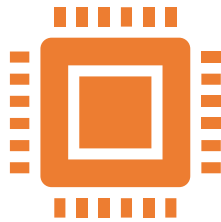
(目的)

- examines not only the evolution of OSS systems, but also the evolution of the associated OSS communities, as well as the relationship between the two types of evolution

(OSSシステムの進化だけでなく、関連するOSSコミュニティの進化、および2つタイプの進化の関係も調べる)

ROLES OF COMMUNITY MEMBER

(コミュニティメンバーの役割)



Passive User

(パッシブユーザー)

use the system in the same way as most of us use commercial software

(私たちが商用ソフトウェアを使用するとだいたい同じ方法でシステムを使用)



Reader

(読者)

active users of the system

(システムのアクティブユーザー)

not only use the system, but also try to understand how the system works (システムを使用するだけでなく、システムの仕組みを理解しようとする)

like peer reviewers in traditional software development organizations

(従来のソフトウェア開発組織の査読者のような)

ROLES OF COMMUNITY MEMBER

(コミュニティメンバーの役割)



Bug Reporter

(バグレポーター)

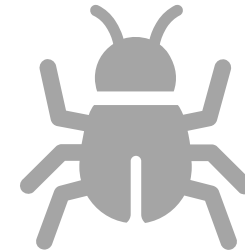
discover and report bugs
(バグを発見して報告する)

do not fix the bugs themselves, and they may not read source code either

(バグを修正しなく、ソースコードも読まない可能性がある)

same role as testers of the traditional software development model

(従来のソフトウェア開発モデルのテスターと同じ役割)



Bug Fixer

(バグ修正者)

fix the bug that is either discovered by themselves or reported by Bug Reporters

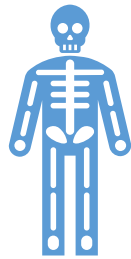
(自分で発見されたバグ、またはバグレポーターが報告したバグを修正)

have to read and understand a small portion of the source code of the system where the bug occurs

(バグが発生するシステムのソースコードのごく一部を読んで理解する必要がある)

ROLES OF COMMUNITY MEMBER

(コミュニティメンバーの役割)



Peripheral Developer

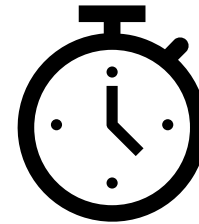
(周辺開発者)

contribute occasionally new functionality or features to the existing system

(既存のシステムに時々新しい機能を作成)

contribution is irregular, and the period of involvement is short and sporadic

(貢献は不規則であり、関与期間は短くて散発的)



Active Developer

(アクティブな開発者)

regularly contribute new features and fix bugs
(定期的に新機能を提供し、バグを修正する)

one of the major development forces of OSS systems
(OSSシステムの主要な開発力の1つ)

ROLES OF COMMUNITY MEMBER

(コミュニティメンバーの役割)



Core Member (コアメンバー)

responsible for guiding and coordinating the development
(開発の指導と調整を担当)

have been involved with the project for a relative long time and
made significant contributions to the development and
evolution of the system

(プロジェクトに比較的長い間関わっており、システム
の開発と進化に大きく貢献した人々)

they are also called Maintainers
(メンテナーとも呼ばれる)



Project Leader

(プロジェクトリーダー)

often the person who has initiated the project
(多くの場合、プロジェクトを開始した人)

responsible for the vision and overall direction of the project
(プロジェクトのビジョンと全体的な方向性の責任)

COMMUNITY STRUCTURE

(コミュニティの構造)

the role closer to the center has
a larger influence (中心に近い
役割のほうが影響力が大きい)



Figure 1: General Structure of an OSS Community

EVOLUTION OF SYSTEMS (システムの進化)

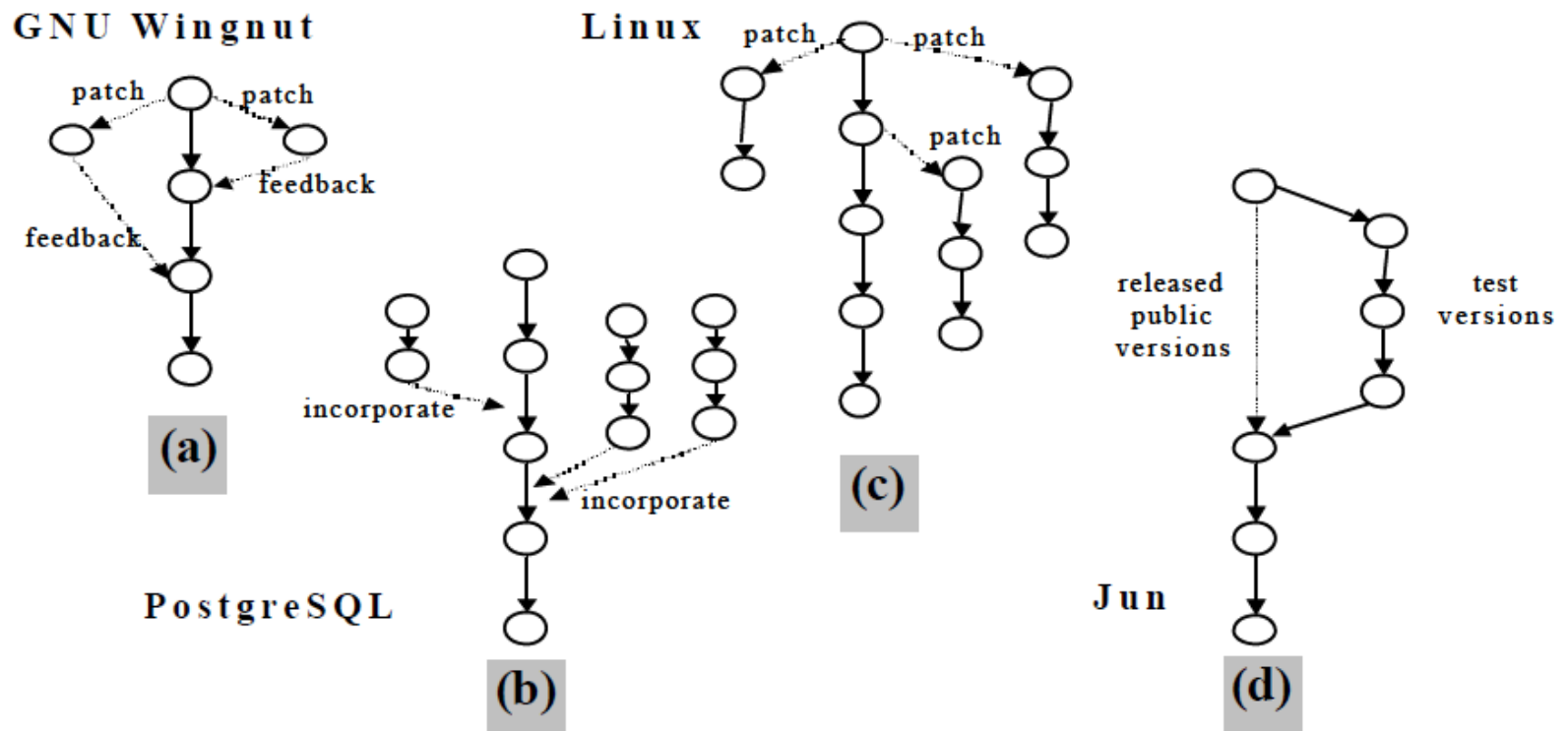
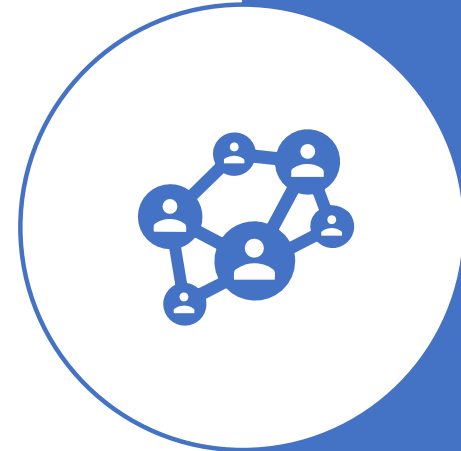


Figure 2: Evolutionary Patterns of the Four Projects

EVOLUTION OF COMMUNITY (コミュニティの進化)

- evolution of an OSS community is brought about by the role changes of its members (OSSコミュニティの進化は、そのメンバーの役割の変化によってもたらされる)
- As community members change the roles they play in the community, they also change the social dynamics, and reshape the structure, of the community (コミュニティのメンバーがコミュニティで果たす役割を変更すると、コミュニティの社会的ダイナミクスも変更され、構造が再構築される)
- evolution of an OSS community is determined by two factors (コミュニティのメンバーがコミュニティで果たす役割を変更すると、コミュニティの社会的ダイナミクスも変更され、構造が再構築される)
 - the existence of motivated members who aspire to play roles with larger influence (より大きな影響力を持つ役割を果たす意欲のあるメンバーの存在)
 - the social mechanism of the community that encourages and enables such individual role changes. (そのような個々の役割の変更を奨励し、可能にするコミュニティの社会的メカニズム)



3 TYPES OF OSS PROJECTS

(3種類OSSプロジェクト)



**EXPLORATION-
ORIENTED OSS**
(探査向けのOSS)



**UTILITY-ORIENTED
OSS**
(実用向けのOSS)



SERVICE-ORIENTED OSS
(サービス向けのOSS)

EXPLORATION-ORIENTED OSS

(探査向けのOSS)

Aim: to push the frontline of software development collectively through the sharing of innovations

(目的：イノベーションの共有を通じて一貫したソフトウェア開発の最前線を推進する)

becomes a learning resource for thousands of software developers

(数千のソフトウェア開発者向けの学習リソースになる)

quality requirements are often very high

(品質要件は非常に高い)

must be developed and maintained by expert programmers, such as Project Leaders

(プロジェクトリーダーなどの専門のプログラマーが開発および保守する必要がある)

success of such OSS projects depends greatly on the vision and leadership of the Project Leader

(このようなOSSプロジェクトの成功は、プロジェクトリーダーのビジョンとリーダーシップに大きく依存する)

- when the vision of the Project Leader conflicts with the needs of the majority of the OSS community members, forking might happen
(プロジェクトリーダーのビジョンが大多数のOSSコミュニティメンバーのニーズと対立する場合、分岐が発生する可能性がある)

UTILITY-ORIENTED OSS

(実用向けのOSS)



developed because the developers cannot find an existing program that fulfills their needs completely

(開発者がニーズを完全に満たす既存のプログラムを見つけることができないために開発された)



developers decide to develop their own systems and put it on the web for others

(開発者は自分でシステムを開発し、他の人のためにウェブ上に置く)



a typical bazaar type software development-no centralized control exists

(バザール型ソフトウェア開発-集中管理は存在しない)

UTILITY-ORIENTED OSS

(実用向けのOSS)

has a lot of forks, evolving in the form of developing new programs by reusing and modifying existing programs rather than replacing the old programs with the new ones

(古いプログラムを新しいプログラムに置き換えるのではなく、既存のプログラムを再利用および変更して新しいプログラムを開発するという形で進化する、多くの分岐点がある)

gives to the rise of multiple, often incompatible programs
(複数の台頭をもたらし、しばしば互換性のないプログラム)

programs that implement the similar functionality compete with each other, but the implementation that wins the most support in the community will finally excel and eliminate other competing versions
(同様の機能を実装するプログラムは互いに競合するが、コミュニティで最も多くのサポートを獲得したプログラムが勝つ)

This evolution pattern can be called as the *tournament style*
(この進化パターンはトーナメントスタイルと呼ぶことができる)

UTILITY-ORIENTED OSS

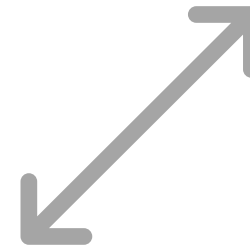
(実用向けのOSS)



may not have an independent community

(独立したコミュニティがない場合がある)

only want to develop a program for their own particular need,
such developers remain to be Peripheral Developers
(特定のニーズに合わせてプログラムを開発したいだけ
で、そのような開発者は周辺開発者のまま)



Passive Users who just want to use the system
need Readers and Peripheral Developers-to help
them choose the right configuration

(システムを使用したいだけのパッシブユー
ザーには、適切な構成を選択するための読者
と周辺開発者が必要)

SERVICE-ORIENTED OSS

(サービス向けのOSS)



aims at providing stable and robust services to all the stakeholders of OSS systems (OSSシステムのすべてのステークホルダーに安定したサービスを提供することを目指している)

stakeholder refer to both the member of the OSS community who uses the system and the end-user (ステークホルダーは、システムを使用するOSSコミュニティのメンバーとエンドユーザーの両方を指す)



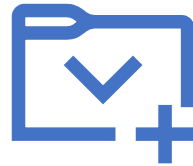
any changes made to the system have to be carefully considered so that they do not disrupt its provided services (システムに加えられた変更は、提供されるサービスを中断させないように慎重に検討する必要がある)

usually very conservative against evolutionary and rapid changes (通常、進化的かつ急速な変化に対して非常に保守的)



centralized in the Council style - not controlled by any individual person (評議会スタイルで一個人によって管理されていない)

the Council is the assembly of the Core Members, who earn their rights by long-time devotion and contributions to the OSS community (評議会は、長年の献身とOSSコミュニティへの貢献が持っているコアメンバー)



DISCUSSION

(考察)

- Recognizing the differences of different types of OSS projects help OSS developers to identify their projects with a particular type and to take appropriate measures to guide the management and operation of the OSS project
(OSSプロジェクトの異なるタイプの違いを認識することは、OSS開発者が特定のタイプのプロジェクトを識別し、OSSプロジェクトの管理と運用をガイドする適切な手段を講じるのに役立つ)

SUMMARY OF EACH TYPE

(各タイプの概要)

Table 1: Three Types of OSS Projects

Type	Objective	Control style	System evolution pattern	Community structure	Major problems	Examples
Exploration-Oriented	Sharing innovations and knowledge	Cathedral-like central control	Single branch Feedback from the community	Project Leader Many Readers	Subject to split	GNU systems Jun Perl
Utility-Oriented	Satisfying an individual need	Bazaar-like decentralized control	Multiple versions coexist Tournament style	Many Peripheral Developers Peer support to Passive Users	Difficult to choose the right program	Linux system excluding the kernel
Service-Oriented	Providing stable services	Council-like central control	Single branch Patches merged through control	Core Members instead of a Project Leader Many Passive Users that develop systems for end-users	Less innovation	PostgreSQL Apache

EVOLUTION OF OSS PROJECTS

(OSSプロジェクトの進化)

Exploration-Oriented OSS and Utility-Oriented OSS are good for the initiation of an OSS project, and Service-Oriented OSS is suitable for more mature OSS projects

(探査指向のOSSおよび実用指向のOSSは、OSSプロジェクトの開始に適しています。サービス指向のOSSは、より成熟したOSSプロジェクトに適している)



Why need to mutate to Service-Oriented type? (なぜサービス指向型に変更する必要があるのですか?)

Exploration-Oriented OSS projects

(探査指向のOSSプロジェクト)

- users at a certain point, will demand stability (特定の時点でのユーザーは、安定性を要求)
- original Project Leader has lost his or her exploration spirit (元のプロジェクトリーダーは自分の探査精神を失いました)
- system has grown too large to be controlled by any single person (システムが大きくなりすぎて、一人で制御できない)

Utility-Oriented OSS projects

(実用指向のOSSプロジェクト)

- developers of competing implementations for a similar functionality can join forces to create a team to develop a system collaboratively that accommodate the different needs of each developer (同様の機能の競合する実装の開発者は、各開発者のさまざまなニーズに対応するシステムを共同で開発するチームを作成するために力を合わせることができる)

EVOLUTION OF OSS PROJECTS

(OSSプロジェクトの
進化)

- Exploration-Oriented and Utility-Oriented OSS projects experience rapid evolution, mostly in linear fashion (探索指向および実用指向のOSSプロジェクトは、ほとんど直線的に急速に進化する)
- As the projects mature into Service-Oriented ones, the speed of evolution will slow down to a stable growth (プロジェクトがサービス指向のものに成熟するにつれて、進化の速度は安定した成長に減速する)
- A new round of rapid evolution will start again if the stable OSS system inspires new ideas or new requirements, giving birth to New Exploration-Oriented or Utility-Oriented OSS projects, which will again mature into new Service-Oriented ones (安定したOSSシステムが新しいアイデアや新しい要件を刺激し、新しい探索指向または実用指向のOSSプロジェクトが生まれ、新しいサービス指向のプロジェクトに再び成長すると、急速な進化の新しいラウンドが再び始まる)
- similar to the biological evolutionary process (生物学的進化プロセスに類似)

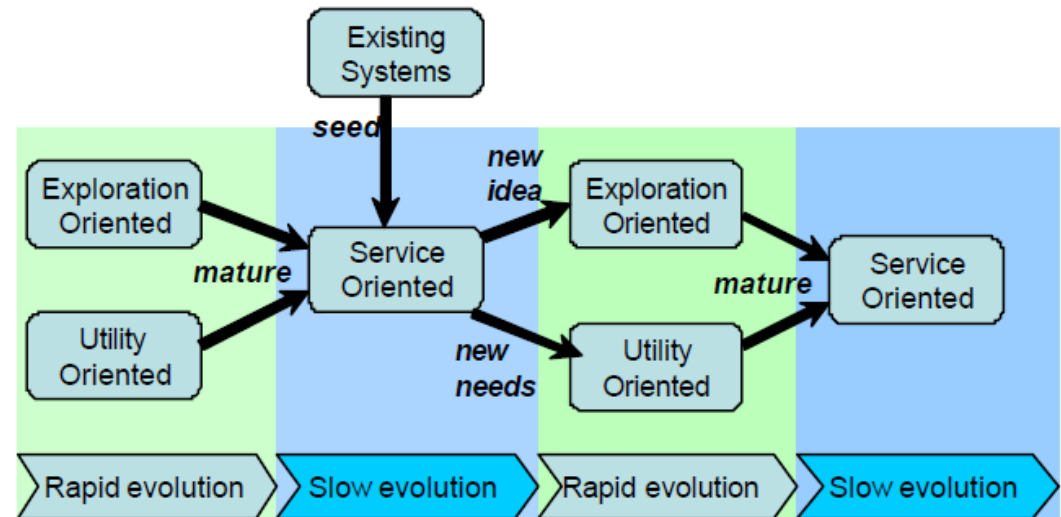


Figure 3: The Evolution Pattern of OSS Projects

CONCLUSION

(まとめ)

- As Open-Source gains popularity, many software development approaches are labeled Open-Source simply because one property of the outcome is Open-Source
(オープンソースの人気の高まるにつれて、多くのソフトウェア開発アプローチはオープンソースとラベル付けされている)
- However, objectives, motivations, collaboration models, system evolution patterns, and community structures and evolution patterns differ significantly from projects to projects. (ただし、目的、動機、コラボレーションモデル、システムの進化パターン、コミュニティ構造と進化パターンは、プロジェクトごとに大きく異なる)
- Rather than focusing on the common features of all Open-Source projects, this paper attempted to systematically treat the differences of OSS projects by examining the evolution patterns of systems and communities
(この論文では、すべてのOSSプロジェクトの共通機能に焦点を当ててではなく、システムとコミュニティの進化パターンを調べることで、OSSプロジェクトの違いを体系的に扱うことを試みました)
- Through analyzing a case study of four typical OSS projects, we proposed three types of OSS projects (4つ典型的なOSSプロジェクトのケーススタディを分析することにより、3種類のOSSプロジェクトを提案しました)
- Understanding the difference of the different types of OSS projects is the first step to build a theory that can guide the development of OSS projects
(OSSプロジェクトの異なるタイプの違いを理解することは、OSSプロジェクトの開発を導くことができる理論を構築する最初のステップ)

