A Structure for Innovation Reproduction in the Eclipse Ecosystem

Abstract: In an open source software (OSS) development community supported by spontaneous volunteers, personal interests, technical capabilities, hunger for fame, and the satisfaction of contributing to the public good are said to be motivating factor for participation. In that community, vendors always play auxiliary roles, and integrate the result of OSS into their business activities.

However, in the Eclipse open source software community, the main role of OSS development activities is taken over by vendors. The relationship between individuals and vendors is reversed. Therefore, it becomes important to maintain the motivation of the development community, promote innovation and link the activities to the profit of vendors. In other words, management of co-creation and competition are being conducted at the same time.

This paper tries to clarify internal and external structures in an OSS ecosystem led by vendors considering the Eclipse community as one particular case. It also constructs a co-creation model to promote sustainable development for an OSS ecosystem. Some mechanisms that drive this process are embedded everywhere in the Eclipse ecosystem.

Keywords: OSS, open source software, ecosystem, innovation management, open innovation

1 Introduction

The word "Ecosystem" has high visibility in the media nowadays. In this background, success in the ecosystem comes to mean success in business. In that, people who not only provide products and services but also build an ecosystem around them with stakeholders can lead their business to success by using market creation and the diffusion power of the ecosystem.

In this sense, the stakeholders who participate in the ecosystem share a common destiny. The ecosystem as the common destiny develops competition with other

outside ecosystems. Inside the ecosystem, participants simultaneously compete and cooperate with each other. Inside and outside of the ecosystem, complex relationships are constructed among participants.

On the other hand, it has been a long time since a software development activity called "open source" is widely recognized in the software industry. In an OSS community, a source code that is software intellectual property must be opened to the public, and developers all over the world develop it by cooperating with each other. Originally, an OSS activity began spontaneously based on researchers' and developers' volunteer work. However, people have realized the power of OSS that can build functions and qualities developed by the volunteer developers equal to those of a business product, and create new business opportunities since the success of Linux. It gathers momentum to leverage the OSS community.

Eclipse, which is this research target, constructs an ecosystem by using the power of OSS. The aim of this research is to clarify the structure of co-creation in an OSS ecosystem by investigating Eclipse as a case study.

1.1 Literature Review

Moore (1993, 1997, 2005) introduced the concept of an ecosystem into management strategy or inter-organizational relation at the beginning and called it "business ecosystem." The phenomenon in which competition and co-creation coexisted simultaneously in the relation among vendors was recognized. The word "ecosystem" which was a term borrowed from ecology was applied to and explained for this phenomenon.

Iansiti and Levien (2004) approached the business ecosystem using the knowledge of ecology or network theory. The role observed in a business ecosystem was classified, and also the indicator of the health of an ecosystem was proposed.

Regarding the OSS, there are some researches which studied the open source strategy as an open innovation(Chesbrough, 2003).

West and Gallagher (2006) researched the various licenses and business models in the open source, and summarized how value offer and value acquisition are balanced. The open source as an open innovation strategy is classified into four strategies.

Thomas and Hunt (2004) explained the cause for starting an open source activity of a community basis and developers participating in it. According to them, every open source project starts in order to satisfy a developer's needs. They found most probably boil down to a combination of need, pride, ambition, or fame of community.

West and O'mahony (2008) studied 12 open source projects initiated by corporate sponsors, and identified three design parameters that helped form a participation architecture. These 3 parameters are 1) organization of production - the way that the community conducts production processes, 2) governance - the processes by which decisions are made within the community, and 3) intellectual property - the allocation of rights to use the community's output. Johri et al. (2011) investigated the effect of OSS steward company. By studying a case of MySQL, they

found that the impact on participation depended on how the acquiring company was perceived.

These previous researches studied some factors to forming successful OSS community. In this paper, we discuss internal and external structures of co-creation and business competition of Eclipse ecosystem, and clarify the mechanisms in which the factors work to form the ecosystem.

1.2 Research Questions

Moore (1993, 1997) and Iansiti and Levien (2004) studied ecosystems of Wal-Mart, Intel and Microsoft and analyze them as these result from accidents. However, Eclipse ecosystem is built by design and some mechanisms are implemented in it to form the ecosystem. It is also a challenge of Eclipse ecosystem to utilize the power of OSS.

This paper explicates management for knowledge collection and creation and business competition and cooperation in the ecosystem by raveling the structure embedded in it. To understand the structure, we set research questions as follows:

RQ1 What mechanisms are implemented in the Eclipse ecosystem?

RQ2 What is the developing process of the Eclipse ecosystem?

RQ3 Why did IBM start the Eclipse ecosystem?

Fig. 1 illustrates the relation of each question. RQ1 and RQ2 clarify the internal structure of the ecosystem, and RQ3 clarifies the external structure against other ecosystems.

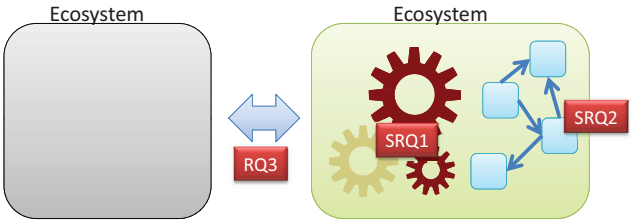

Figure 1 Research Questions

First, we shall clarify the mechanisms implemented in the Eclipse ecosystem from the side of both technology and system. Next, we show how those mechanisms are working effectively through interviews with participant vendors, and then suggest a model of the developing process of the Eclipse ecosystem. For the RQ3, we study the background of starting Eclipse in 2001 and infer its external role.

2 Architectures in Eclipse Ecosystem

Fujimoto et al. (2001) tried to understand various corporate activities by applying to them the concept of the architecture. They observed the activities from the viewpoints of division and coordination.

Baldwin and Clark (2006) studied the architecture of OSS codebase. They show that codebases that are more modular or have more option value (1) increase developers' incentives to join and to remain involved in an open source development effort; and (2) decrease the amount of free-riding in equilibrium.

In this paper, we focus on not only "technology architecture" like codebase but also "system architecture." We try to clarify the OSS activity called the Eclipse ecosystem from these two aspects. To explain these architectures, we introduce the purpose of the Eclipse ecosystem declared in the bylaws of the Eclipse Foundation.

Then, these purpose are how to implemented as the architecture of “technology” and “system.”

2.1 Purpose of Eclipse

In section 1.1 of the Bylaws of the Eclipse Foundation, the purpose of the Eclipse technology and foundation is stated(Eclipse Foundation, 2008).

- Purpose of Eclipse Technology

  To provide vendor-neutral, open development platform supplying frameworks and exemplary, extensible tools (Eclipse platform).

- Purpose of Eclipse Foundation

  To advance the creation, evolution, promotion, and support of the Eclipse Platform and to cultivate both an open source community and an ecosystem of complementary products, capabilities, and services.

The technology in the Eclipse ecosystem is software of the platform, and we call the system support that the Eclipse Foundation provides.

2.2 Technology Architecture

In a software development project, a development environment is tailored to a target language and system. Because the development environment is optimized for the purpose, there are some slightly different parts of functionality and also many common parts. Despite the common functions, different departments often develop a similar function separately for each programming language. Moreover,

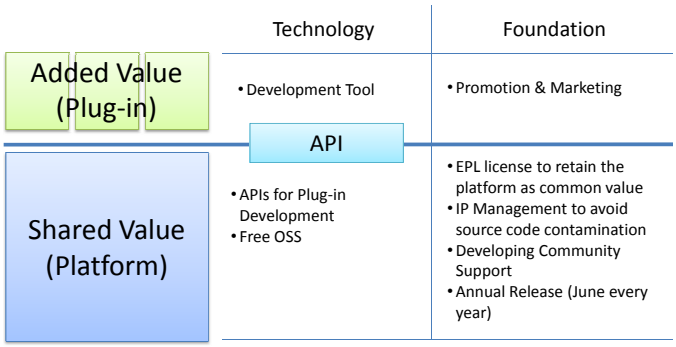

Figure 2 Eclipse Platform Architecture

due to independently developing tools, they are not seamlessly integrated and not ease to use.

IBM had a similar issue, so that providing a common tool platform gathered momentum. According to Cernosek (2005), Eclipse was started to avoid dual tool development in November 1998. Eclipse adopted platform runtime and plug-in architecture, namely modular architecture. A plug-in is the smallest unit of Eclipse Platform function that can be developed and delivered separately(desRivieres and Wiegand, 2004). The common platform avoids dual developments, works smoothly together with tools built on the platform, and increases usability by providing basic operations as the platform.

The Eclipse software architecture of Fig. 2 was designed as follows:

- The common function is provided as a platform, and additional functions are implemented as plug-ins;
- The platform Application Program Interface (API) is defined to use the common function; and

- The plug-ins are coded by using the APIs and a plug-in development tool is also provided.

In 1998, Eclipse software architecture implemented the modular architecture which separated the platform and the plug-ins of IDE (integrated development environment). Eclipse 3.0 released in 2004 adopted the OSGi (Open Services Gateway initiative) standard framework for modularity(Gruber et al., 2005). The OSGi framework is a module system and service platform for the Java programming language that implements a complete and dynamic component model. OSGi added a more flexible modularity management function to Eclipse. Eclipse 4.0 is next generation platform for component-based applications and tools which has been restructured to develop plug-ins by using common Web technology such as CSS (Cascading Style Sheets) and JavaScript.

Gawer and Cusumano (2002) point out the modular architecture as the following:

> For a modular architecture to encourage third-party innovation, the interfaces should be open.

Shintaku et al. (2006) summarize one of the characteristics of the modular architecture is to accelerate diffusion and popularization of technology and product.

Eclipse software architecture evolutions seem to intend to gather more participants than ever before and to promote innovation by adopting open standard module technologies.

2.3 System Architecture

This section describes how the system shown in Fig. 2 contributes to the development of the Eclipse ecosystem. We choose the following significant services in Eclipse ecosystem as the system:

- Eclipse Public License (EPL)
- IT Infrastructure
- Development Community Support
- Intellectual Property (IP) Management
- Ecosystem Development (Promotion and Marketing)

These services are provided by the Eclipse Foundation and defined in the document "About the Eclipse Foundation(Eclipse Foundation)."

2.4 EPL and Value Sharing

This section explains the EPL, which is one of the open source licenses adopted by Eclipse, and show how it works as a mechanism to share knowledge and value in the Eclipse ecosystem.

2.4.1 License, Development Model and Business Model

Before explaining the EPL, we shall clarify the relationships among the open source license, the development model and the business model. Software is a type of pure intellectual product that has no physical substance. A software license grants the use of software to users under certain conditions. In other words, the software license is a

method to control intellectual property. Because open source software is a software, if its source code is disclosed, its intellectual property is also disclosed. Open source activity is a development activity of public intellectual property. Almost all of them are formed collaborative development style. Open source business is based on public intellectual property.

A software license has a significant impact on the open source development model and the business model. Namely,

- A too closed license will not create the development communities,
- A too open license will limit business opportunities.

Therefore, a well-balanced license is important for management.

2.4.2 EPL

We will show how the balance is maintained by the EPL. In general, the open source license prescribes recipient's rights and obligations. The recipient's rights and obligations of the EPL are specified as follows:

- Rights:

  A non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense, and royalty-free patent license.

- Obligations:

  Providing software in source code form and distributing it with a license equivalent to EPL.

In the EPL, many software rights are permitted, and it is possible to execute, modify, distribute and sell it. Furthermore, it is the free of a licensing even if it is patented. The rights to distribute, extend and develop are fully granted to the recipients.

On the other hand, there is an obligation to re-distribute its source code with the EPL. If a developer modifies the software which is licensed under EPL (EPL'ed) and newly create a derivative from it, the developer has to grant a royalty-free copyright and a patent license to recipients of the derivative. And the developer must provide the software in source code form. This means sharing the newly created intellectual property on the EPL'ed IP as the public good. However, this obligation is applied only when the base software is EPL'ed. If a developer creates an add-on software like Eclipse plug-in, one can apply any license even if it is a commercial license.

By building a plug-in on top of the Eclipse platform, the value which Eclipse provides can be acquired and a developer's original value can be added in the form of a plug-in. New values by modifying the platform, such as an improvement, are shared with others since the license imposes the obligation to open the source code.

In Eclipse, a boundary of value acquisition and sharing is clearly prescribed by the EPL, and this boundary brings a balance.

2.5 IP Management: Clearance Procedure

When developing a software product based on open source codes, it is difficult to avoid violation of a license and/or a patent. It is troublesome if the GPL (Gnu Public License) is mixing in source codes(Founation, 2007). This is because the

source codes of software which is licensed under the GPL (GPL'ed) must be open. Source codes of derivative works from GPL'ed code must also be open. The GPL goes further. If GPL'ed code gets mixed in the proprietary software, the vendor must open its source code. There are some cases that software vendors finally open the all source codes of their proprietary software products because GPL'ed codes get mixed in even though it happens by accident.

Mixing of the patent of the other companies is also a troublesome problem. A patent infringement case occurs since a patent of another company is unawarely mixed in the open source code used as the base of a product. For instance, SCO sued DaimlerChrysler and U.S. AutoZone which were the users of Linux noting that the patent of its company was used for Linux(Thiessen, 2004).

Since open source software is not necessarily developed from the scratch in one company, it is difficult to eliminate the possibility that unknown developers have copied and pasted a source code of doubtful origin. Moreover, the OSS developer who introduces the patent into OSS will not be sued, but the vendor who uses and sells the software in which the patent mixed will be sued. Furthermore, because patent litigation requires time and expenses, once a patent litigation occurs, the impact on business is severe especially for a venture business. It could cost the vendor millions in either a judgment or a settlement.

As mentioned above, impurity incorporation of a license and/or a patent is one of the severe problems for the vendor who considers utilizing OSS as base of its commercial software.

In the Eclipse, in order to cope with this problem, the clearance of intellectual property is institutionalized. Thanks to this system, neither the source code of a GPL nor a third party's patent mixes in the code which Eclipse provides. And royalty-free use will be guaranteed even if the patent is contained.

In the Eclipse project, a source code is managed by a repository. The repository is a system which keeps and manages source codes. Before committing a code into the repository, a developer has to pass through an examination procedure called "Eclipse Legal Process." This procedure prevents a code with a license other than of the EPL or a lawsuit risk code from mixing.

In this procedure, the source code is examined as follows:

1. In the case of a code 100% developed from scratch, the code will be committed with the EPL because there is no room for mixing.

2. Also, when the source code is developed based on the EPL'ed code, the code will be committed with the EPL because there is no room for mixing.

3. In other than the above, the Eclipse Management Organization (EMO) judges whether it can be provided with the EPL. In that case, it is registered in the IP tracking system, and log of the judgment and process is recorded.

4. When it is judged that the EPL cannot be applied, the source code is eliminated.

By the above procedure, intellectual property clearance of the code is carried out. Because of this procedure, a vendor is reassured and can utilize a source code without a lawsuit risk.

Janet Compbell who designed this IP managing process says, "It is designed to reduce a developer's load as much as possible."

2.6 Development Community Support and IT Infrastructure

Everyone who sees the success of the Linux open source activity must dream of utilizing the market creation power of OSS, its high productivity and innovation speed. However, it is difficult to start an open source project and form its community by oneself because of a deficient know-how.

In 1998, Netscape released the source code of flagship browser product as an open source. However, they failed to launch its community. Jamie Zawinski, a powerful promoter for OSS of Netscape, mentioned "It is most definitely not a panacea. If there's a cautionary tale here, it is that you can't take a dying project, sprinkle it with the magic pixie dust of 'open source,' and have everything magically work out."(Weber, 2004)

Even if source codes of software are released as OSS, its community does not start and grow automatically. Therefore, Eclipse prepares a support scheme to start a OSS project. Support schemes are enumerated below:

- Stipulated incubation process of new project

  The responsibility range of a participating vendor is clarified by this document.

- Established support organization

  Responsibility and a role are made clear. The PMC (Project Management Committee) of a technology top project is responsible for an incubation project.

- Risk reduction measure

  A mentor is assigned at the time of the establishment of a project, and helps in developing the project. Moreover, the project is to be reviewed at some checkpoints in the middle of an incubation process.

- Guarantee of openness and transparency

  At the time the project established, a proposal is opened to public and its aim, scope, roadmap, etc. are clarified. Then, participants are attracted. Moreover, a review result and the minutes are also opened to public on website (www.eclipse.org).

- Support of IT infrastructure

  The Eclipse Foundation is preparing IT infrastructures required for project management, such as the website of a project, a repository of a source code, issue and bug management.

Through the above supports, developers are not so busy about complicated community management, and can concentrate on technology development.

2.7 Marketing and Promotion

In order to develop the Eclipse ecosystem, the Eclipse Foundation conducts common marketing with member companies, and community conferences (EclipseCon and Eclipse Summit Europe) annually in the U.S. and Europe, respectively. In the conferences, Eclipse-related vendors and engineers gather from all over the world, and opinions on state-of-the-art technologies and information of Eclipse are exchanged.

Moreover, the Eclipse Foundation provides Eclipse Live, which offers on-line seminars called Webinar, and Eclipse plug-in on-line catalog site Eclipse Plugin Central called EPIC. These activities can increase the visibility of Eclipse and of Eclipse plug-ins, and also present demonstrations of usages and descriptions that help to draw in users.

Furthermore, when a member company releases an Eclipse-related product, the Executive Director of Eclipse will endorse its press release if requested.

2.8 Resonance: Simultaneous Release

Release timing is an important factor in software ecosystems(Jansen et al., 2009). In some OSS communities, the release date of the new version is uncertain. Even if the release date is determined, it is often postponed. So it is difficult for participants to make a business plan because of the uncertain release date.

As we explained so far, the plus-ins depend on the Eclipse platform APIs, so that the version of the platform and the plug-ins must synchronize. In the Eclipse ecosystem, Eclipse Roadmap is published annually.And almost all of the projects

synchronize and release the upgrade versions all at once every year at the end of June. The vendor developing an Eclipse-related product or service easily makes a plan by simultaneous release.

To maximize the co-evolution in the ecosystem, it is important that the participants synchronize and resonate mutually.

2.9 Summary of Architecture

We can provide the answer to RQ1 "What is the mechanism embedded in the Eclipse ecosystem?"

The software architecture of Eclipse adopts a modular structure and we can add new features to the platform as a plug-in by using APIs. In addition, the structure of the modularization is evolving to adopt more open and standard technology. This is considered to have aimed at an entry cost reduction of plug-in implementation, while widening the developer base by adopting standard technology.

The Eclipse ecosystem clarifies the boundary line of value acquisition and of value sharing by the software architecture and the license.

The architecture of system described in the structure of Fig. 2 helps participating IT vendors concentrate on technology innovation, and the Eclipse Foundation supports other activities such as IP management, project management, and promotion as much as possible.

Furthermore, the simultaneous release for maximizing the effect of co-evolution among the participants in the Eclipse ecosystem is also implemented.

3 Investigation through Interviews

As explained above, Eclipse is equipped with a mechanism which supports a participating vendor in both sides of technology and system. How do the participating vendors consider these mechanisms? In order to clarify this, interviews with some vendors were held in EclipseCon 2009. The member companies of Eclipse which met in EclipseCon were targeted as interviewees. The interviews were carried out mainly with ventures.

As a result of the interviews, the following four points were discovered as participating motives of Eclipse.

- The Eclipse community is attractive as a market.
- A reduction of the development cost is expected.
- The IP managing process is serviced.
- The EPL is suitable for business.

3.1 Eclipse Marketability

Because of high marketability, there were a number of venture businesses which decided to participate in the Eclipse ecosystem.

According to the Eclipse Foundation report (Eclipse Foundation, 2010), the number of downloads is more than 1 million a month at the report time. Also it reports that Eclipse is used all over the world. Especially, it is actively used in Asia. It is natural that ventures are attracted by the number of Eclipse installed bases and worldwide usages.

ProteCode, one of Eclipse member companies, simply determined to participate in Eclipse by the difference in the number of user bases comparing with other open source Java IDEs. Sopera, a German venture company of SOA (Service Oriented Architecture), said that the reason for having participated in Eclipse was as follows:

> The most efficient way of approaching a user community was donation of the code to Eclipse. They start a SOA project in Eclipse based on their contributed source codes, and lead where the codes evolve. Their business is done by providing services like support, consulting, etc. instead of the software itself.

It is in the situation where huge competitors like IBM and Oracle already exist in the market of SOA. And it is difficult generally for a small venture to enter the market. However, it is possible for the small venture to enter the market if it contributes source codes to Eclipse as OSS.

Actuate is one of the successful companies which participates in Eclipse actively. Actuate reported the market cultivation power of Eclipse in the session of EclipseCon 2009. Actuate is a vendor of BI (Business Intelligence) tool, and is developing a visualization tool for business data. In the participation in Eclipse, they contributed the engine of the visualization tool as OSS. As a result, the visibility of Actuate improved. Actuate succeeded in penetrating the existing market and in expanding new markets.

According to Actuate's report, the number of downloads in 2008 recorded 6,500,000 or more. Only 1 % of the downloaded users became their commercial customers. Although 1% seemed to be small, it was realized as business because the number of total downloads was huge. Actuate said it was able to enter new markets in which Actuate had not done business. Actuate's main market was the financial market, and it had hardly entered other industry markets. However, it could enter the new markets like merchandising, manufacturing, telecom business,

etc., by the code donation of the engine. Also, Actuate could gain new users in overseas markets, like in India and China, which they had not been able to enter.

3.2 Development Cost Reduction

There are two kinds of costs in software. One is developing cost for new functions, and the other is maintenance cost. The former is reducible by developing a common function jointly with other vendors. For the latter, a comprehensive test is achieved by getting as many users as possible. Besides, the more there are users, the more bugs are detectable. In maintenance, the bug patch cost is expensive, and the bug detection cost is also quite expensive.

Webtide, a venture of the Eclipse members, said it was determined to participate in Eclipse expecting to increase the number of developer bases. Jetty web server which Webtide is developing uses the same OSGi technology as Eclipse uses as the base. By participating in Eclipse, it could be said that a synergistic effect due to the increase in a developer base was expected.

On the other hand, there is also a venture which expected a reduction of the maintenance cost. Sopera is one of such companies. In the interview, the CTO of Sopera told that "By contributing our code as open source, many people are able to use it, and then many faults are discovered at an early stage." He expected a 50% reduction in maintenance cost.

3.3 IP Management Process

There were some vendors which indicated the IP management process as a reason to join Eclipse.

As explained in 2.5, IP interfusion is one of the most severe problems for companies which use open source. However, the Eclipse Foundation provides the IP management process which avoids the interfusion of the codes. It eliminates vendor's anxiety and gives a sense of security.

3.4 Summary of the Interviews

There are three reasons for vendor participation: "marketability of Eclipse," "development cost reduction," and "IP management process."

Vendors feel attracted to the marketability of Eclipse. This means obtaining profits from Eclipse, namely they are trying to do "value acquisition" from the Eclipse ecosystem. On the other hand, a development cost reduction means cooperating with the participants in Eclipse. That is "value sharing." The EPL and IP management process provides the exclusion of risk. That is a "sense of security." These are summarized as follows:

Marketability of Eclipse	→	Value Acquisition
Development Cost Reduction	→	Value Sharing
IP Management Process	→	Sense of Security

The interviews discover that the vendors recognize that they can acquire the value from the ecosystem by participating in it. Acquiring values are the cost reduction such like high productivity and maintenance cost reduction and profit by market expansion. One interesting thing is that they consider providing their source code as OSS platform is the most effective way to approach users. This means there is the structure which yields more returns by promoting the software of a vendor to users indirectly than by doing it directly. In other words, Eclipse ecosystem returns more values to the vendor which shares its values.

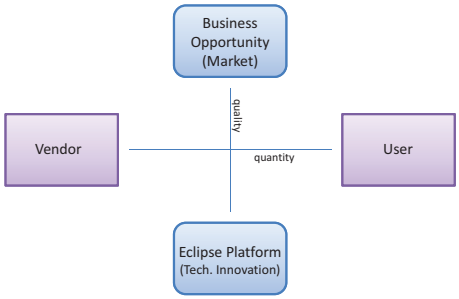

Figure 3 Two Axes and Four Aspects in Eclipse Ecosystem

It is risk for the vendor to share its own values with others. To reduce this anxiety, the Eclipse Foundation clarifies the boundary line of shared values provided as the platform and added values on top of it. The clear boundary line is built by EPL and strictly managed to avoid interfusion. EPL guarantees that the shared value continues to be a public good. Although the vendor provides its important source code to the ecosystem, it fails when the community hasn't been created. Eclipse Foundation provides some support scheme to reduce this anxiety about the failure.

These mechanisms reduce entry barrier of the ecosystem for the vendors.

In the following section, based on the interviews, we will show a model of co-creation process embedded in the Eclipse ecosystem.

4 Internal Structure: Co-creation Process Model

4.1 2 Axes and 4 Aspects in Ecosystem Development Process

To consider the ecosystem development process, we introduce quantitative and qualitative axes in Fig. 3.

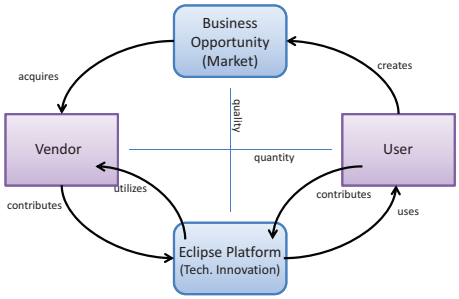

Figure 4 Innovation Cycle in Eclipse Ecosystem

Considering from the quantitative development point of view, the number of participants is an indicator. We can say that the ecosystem in which increasing participants in number is developing. In the Eclipse ecosystem, the participants consist of users and vendors. We draw a horizontal line as the quantitative-axis and allocate vendors and users on each end.

Considering from the qualitative development point of view, increasing business opportunities and evolving Eclipse platform can be indicators. In other words, the increasing of business opportunities is market creation and the evolving platform is innovation. We can say that the ecosystem which creates new business opportunities and new innovation is developing.

Fig. 4 illustrates the Eclipse ecosystem developing process model. This process has three multilayered cycles and the ecosystem is developed in each cycle. We explain these cycles in detail in the following sections, and describe how these work in order to keep the ecosystem active.

4.2 Cycle between Platform and Users

As Raymond (1999) points out, excellent OSS projects evolve in the better direction by involving users. Eclipse has some mechanisms to involve users in its community, such like world-wide download sites, web pages to publish information and to receive bug reports, and Eclipse Webinar and podcasts.

The following process is the developing ecosystem process between the OSS and the users:

1. First of all out-of-the-box OSS is provided;

   Eclipse is outstanding software for Java IDE which is easy to install, user-friendly and improves development productivity.

2. When the out-of-the-box software is provided, users come in the ecosystem to use it;

   To increase the number of users, it is also important that the software be easy to use. Eclipse is provided as a free OSS, moreover, it can be downloaded from anywhere in the world. Therefore, the barrier for users to enter the Eclipse ecosystem is very low. If the software is outstanding, a user will invite other users to use it.

3. When the number of users is increased, OSS receives many feedbacks.

   The more users OSS has, the more bugs will be reported. Then effective debugging is possible. The users of Eclipse are Java developers. So some of users will provide codes for bug-fix and improvement.

If the users report bugs actively and provide improvement codes, the software will be stable and improved rapidly. Then it will invite another user. Raymond (1999) indicates this OSS nature as one lesson:

> Treating your users as co-developers is your least-hassle route to rapid code improvement and effective debugging.

This process is not specialized in the Eclipse ecosystem and observable in the successful OSS projects. We point out the Eclipse ecosystem utilizes this OSS nature.

4.3 Cycle between Platform and Vendors

The relationship between vendors and the Eclipse platform is slightly different from the one between users and the platform.

Common functions for IDEs are provides by Eclipse as the platform. The vendors can concentrate on developing plug-ins that are added values on top it. Because plug-ins are developed by using the platform API, interoperability among plug-ins is seamless. If plug-ins are seamlessly interoperable, it will enable to increase programming productivity and the value of IDE. Geer (2005) picks up a comment of Evans Data's COO:

> There is thus a "huge number" of interoperable third-party plug-ins, which has made Eclipse very popular.

Geer (2005) also points out another merit of Eclipse platform as OSS.

> Because Eclipse is open source, Borland's Cheng said, developers have ready access to the source code and can modify it and innovate quickly to meet users' needs.

If the platform is modified, the source code of the modification will be open according to EPL. As a result, innovation to the platform added by any vendors will always return to Eclipse ecosystem.

To summarize the above, the process is the following:

1. Out-of-the-box platform is provided as OSS
2. Vendors utilize the platform and develop plug-ins on it

As explained before, one of the purpose of Eclipse is to develop tools effectively

by providing common functions as the platform. Some vendors appear to provide added value by developing original plug-ins if the platform is free OSS. The plug-in as added value is licensed with commercial license of the vendor and needs not to be open because of EPL. It is privately-owned property of the vendor.

3. Feedbacks return to the platform

   If the vendor finds a bug of the platform in developing the plug-in, it will write a patch code to fix it. The patch must be open according to EPL, and then return to the ecosystem as a pubic good. There is a case that the vendor needs to enhance the platform to develop its original plug-in. The enhancement code to the platform will return to the ecosystem as well according to EPL.

As described above, the platform is evolving when the vendors utilize it to develop their own plug-ins.

This cycle focuses on utilizing the platform by vendors. The vendor's contribution to the platform is patch. It is slightly small value sharing.

The next section describes the full-scale participation of the vendor in the ecosystem.

4.4 Innovation Reproduction Cycle

As we explained the interviews, some vendors provide the source code of their core technologies as OSS and start Eclipse projects to incubate communities. These vendors consider that contributing core source code to the ecosystem is the most effective way to appeal potential users of their software and/or services. In fact,

Actuate can penetrate new market and make more revenues by contributing core engine of software in source code form. This means there is a structure in which indirect appealing software through the Eclipse ecosystem is more effective and profitable than direct appealing. Fig. 4 illustrates this structure and we explain it in detail.

1. When the out-of-the-box OSS is provided, it attracts many users.

2. If the users increase in number, business opportunities will be produced and market will be created.

   As reported by Eclipse Foundation (2010), the number of downloads of Eclipse is more than one million per month. The number of users is huge. The huge Eclipse market makes good business sense even if the share is small part of the users.

3. Vendors participate in the ecosystem with new value

   To catch this business opportunities, vendors such as niche ventures participate in the ecosystem, bringing new values like new functions and services. Some vendors donate source codes to Eclipse, launch and lead an Eclipse OSS project aiming a market penetration and a reduction of development cost. In that case, as explained in section 3, the project starting support service and the IP management process which the Eclipse Foundation provides eliminate anxiety and give a sense of security.

4. New innovation is brought to Eclipse ecosystem and shared with other participants

Vendor participating in the ecosystem with source code means to bring new innovation. It becomes shared value and someone may innovate on it stimulated by the OSS.

This cycle allows the ecosystem to continuously bring external innovation and create new market. New innovation attracts new users. The new users create new market and it gives vendors an incentive to innovate. This virtuous innovation cycle is the mechanism embedded in the Eclipse ecosystem.

4.5 Summary of the Model

As we explained above, there are three multilayered developing cycles in the Eclipse ecosystem. The important thing in this model is that the ecosystem has turning mechanisms of the three cycles which increase the whole value by reproducing the new value of the ecosystem in each spin cycle. The lower two cycles exist in each project community. And these synchronized by the annual release explained in section 2.8. The simultaneous annual release optimizes the rotation cycles. The whole cycle is to bring core vendors with new innovation in the ecosystem. In the Eclipse ecosystem, innovation reproduction mechanism is embedded.

This is the answer to RQ2 “What is the developing process of the Eclipse ecosystem?”

5 External Structure of Eclipse Ecosystem

IBM announced to donate $40 million of its application development platform to open-source organization, Eclipse consortium. Many media reported this announcement with amazement(Junnarkar, 2001).

This section discusses RQ3. To find the answer of RQ3, we decompose RQ3 into two questions, why did IBM start it in 2001? and Why was it IDE?, by looking back upon the situation in 2001.

Eclipse is Java IDE. Java is a programming language developed by Sun Microsystems.J2EE stands for Java 2 Enterprise Edition (aka Java EE)which is used platform for server programming in the Java. The specifications of J2EE was mainly defined by Sun Microsystems. Some vendors like IBM are licensed from Sun Microsystems to implement and sell its J2EE application server to customers. Therefore, vendors are competitors each other. IBM's J2EE software product is called WebSphere.

.NET is a software framework for network-based application provided by Microsoft. ASP.NET is a Web application framework. It has the similar features of J2EE. Target market of Java and .NET (J2EE and ASP.NET) is almost the same, enterprise application market. Then these are competing software each other. In microeconomics, we call the relation of Java and .NET are substitute goods.

Microsoft was strongly promoting .NET Framework to penetrate server software market with its enormous wealth. .NET Framework ver 1.0 RTM(Release to manufacturing) was released in 2002. However, beta 1 was released in September

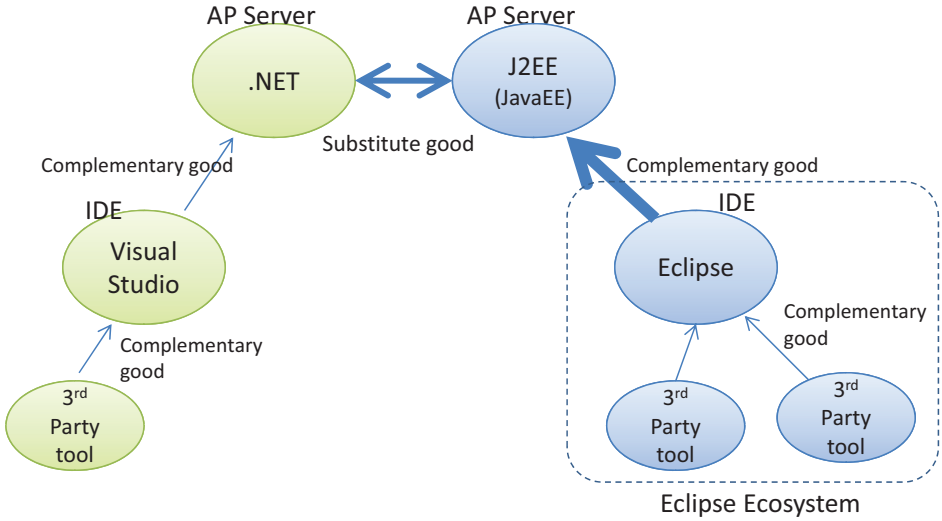

Figure 5 External Structure of Eclipse Ecosystem

2000 and beta 2 was released in June 2001. In 2001, it was attracting interests of medias, enterprise users and third-party vendors.

In those days, some research companies predicted the growth of .NET. One of the reasons of the growth was its IDE Visual Studio .NET. Vawter and Roman (2001) indicated as the following:

> Our conclusion is that Microsoft has the clear win when it comes to tools. While the functionality of the toolset provided by J2EE community as a whole supercedes the functionality of the tools provided by Microsoft, these tools are not 100% interoperable, because they do not originate from a single vendor. Much more low-level hacking is required to achieve business goals when working with a mixed toolkit, and no single tool is the clear choice, nor does any single tool compare with what Microsoft offers in Visual Studio.NET. Microsoft's single-vendor integration, the ease-of-use, and the super-cool wizards are awesome to have when building web services.

Year 2001 was the time to have a presentiment of rise of .NET. Java vendors must have felt fear of .NET. However, Java vendors were competitors each other. So it was difficult to compete Microsoft by mobilizing all. Especially, poor feature of Java IDE was fatal.

That's why IBM donated Eclipse to OSS community. We guess there were two reasons of the donation(Fig. 5).

The first reason is to enrich Java IDE with third-party plug-ins. As explained above, the Eclipse platform attracts third-party tools as plug-ins and increase interoperability of tools. The Eclipse attracts Java developers because it is free OSS. We guess IBM tried to mobilizing Java vendors by providing outstanding Java IDE as OSS and enrich it with third-party tools.

The second reason is to compete Microsoft. From microeconomics point of view, the development environment is a complementary good of runtime environment such like J2EE application server. This means a demand of J2EE is increased when the price of Java IDE is decreased. We guess IBM adopted a strategy against Microsoft by providing Java IDE as OSS that would stimulate demand of J2EE server.

This is the answer for RQ3.

6 Discussion

This research is about the Eclipse ecosystem studied by quantitative methods. So the following three areas are rooms for further research:

- Qualitative analysis,
- Generalization of the model,
- Formalization of the model.

We identify the vendors determine to participate the Eclipse ecosystem by weighing the sharing values and the acquiring values. It is not clarified how much the both should be well-balanced for the vendors to determine participation. We identify the mechanisms and the factors in the ecosystem, but we do not identify how much effect each factor contributes to work it. If we can identify coefficient rate of each factor, we would build an ecosystem more efficiently or control it easily.

We have to verify this structure model is adoptable for other ecosystems, for instance, Apache OSS community which has different license model and architecture or more generic business ecosystems.

To discuss the ecosystem modeling, we need a formalizing method. Boucharas et al. (2009) try to formalize Software Ecosystem Modeling, but they focus on software supply chain. Our model focuses on innovation in the ecosystem. We can't directly apply formalizing method of Boucharas et al. (2009) to the model of this paper. We need new formalizing method of the innovation ecosystem.

7 Conclusion

This paper discovers the internal and external structures of the Eclipse ecosystem.

For the internal structure, we identify the designed architecture in it from the technology and system point of views. We show that the both of architectures draw a clear boundary line between value sharing and value acquisition in both of technology and system which attract vendors to participate. As a result of interviews, the sense of security of the ecosystem is important factor as well. Then we propose the innovation reproduction model inside the ecosystem.

For the external structure, we show the role of the Eclipse ecosystem from the situation of those days when it was born. The Eclipse is a complementary good of J2EE application server. Building the Eclipse ecosystem enriches the server against .NET.

This is one case study of many ecosystems. Different ecosystem may have another structure. However understanding the structure of one ecosystem will be the first step to study other ecosystems. This study will give implications about the case where a vendor designs a strategic ecosystem and where it utilizes an existing ecosystem.

References

Baldwin, C.Y. and K.B. Clark (2006) "The architecture of participation: Does code architecture mitigate free riding in the open source development model?" *Management Science*, Vol. 52, No. 7, p. 1116.

Boucharas, V., S. Jansen, and S. Brinkkemper (2009) "Formalizing software ecosystem modeling," in *Proceedings of the 1st international workshop on Open component ecosystems*, pp. 41–50, ACM.

Cernosek, Gary (2005) "A brief history of Eclipse."

Chesbrough, Henry William (2003) *Open Innovation: The New Imperative for Creating and Profiting from Technology*: Harvard Business School Pr.

desRivieres, J. and J. Wiegand (2004) "Eclipse: A platform for integrating development tools," *IBM Systems Journal*, Vol. 43, No. 2, pp. 371–383.

Eclipse Foundation (2010) "Eclipse Members' Meeting."

Eclipse Foundation, Inc. (2008) "Bylaws of Eclipse Foundation, Inc.."

——— "About the Eclipse Foundation."

Founation, Free Software (2007) "GNU General Public License."

Fujimoto, T., A. Takeishi, and Y. Aoshima (2001) *Business architecture - The Strategic Design of Products Organizations and Process (Eds.)[in Japanese]*: Yuhikaku.

Gawer, Annabelle and Michael A. Cusumano (2002) *Platform Leadership: How Intel, Microsoft, and Cisco Drive Industry Innovation*: Harvard Business School Pr.

Geer, D. (2005) "Eclipse becomes the dominant Java IDE," *Computer*, Vol. 38, No. 7, pp. 16–18.

Gruber, O., BJ Hargrave, J. McAffer, P. Rapicault, and T. Watson (2005) "The Eclipse 3.0 platform: adopting OSGi technology," *IBM Systems Journal*, Vol. 44, No. 2, pp. 289–299.

Iansiti, M. and R. Levien (2004) "Strategy as ecology.," *Harv Bus Rev*, Vol. 82, No. 3, pp. 68–78.

Jansen, S., A. Finkelstein, and S. Brinkkemper (2009) "A sense of community: A research agenda for software ecosystems," in *Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*, pp. 187–190, Ieee.

Johri, Aditya, Oded Nov, and Raktim Mitra (2011) ""Cool" or "monster"?: company takeovers and their effect on open source community participation," in *Proceedings of the 2011 iConference*, iConference '11, pp. 327–331, New York, NY, USA: ACM.

Junnarkar, Sandeep (2001) "IBM makes $40 million open-source offer - CNET News," http://news.cnet.com/IBM-makes-40-million-open-source-offer/2100-1001_3-275388.html, November.

Moore, James F. (1993) "Predators and Prey: A New Ecology of Competition," *HARVARD BUSINESS REVIEW*, Vol. 71, pp. 75–75.

——— (1997) *The Death of Competition: Leadership and Strategy in the Age of Business Ecosystems*: Collins.

——— (2005) "Business Ecosystems and the View From the Firm," *ANTITRUST BULLETIN*, Vol. 51, No. 1, p. 31.

Raymond, E. (1999) "The cathedral and the bazaar," *Knowledge, Technology & Policy*, Vol. 12, No. 3, pp. 23–49.

Shintaku, J., K. Ogawa, and T. Yoshimoto (2006) "Architecture-based approaches to international standardization and evolution of business models," *IEC Century Challenge*, pp. 19–35.

Thiessen, Mark (2004) "SCO sues AutoZone, DaimlerChrysler over use of Linux," March.

Thomas, Dave and Andy Hunt (2004) "Open Source Ecosystems," *IEEE Software*, Vol. 21, No. 4, pp. 89–91.

Vawter, C. and E. Roman (2001) "J2EE vs. Microsoft .NET,"Technical report, The Middleware Company.

Weber, S. (2004) *The success of open source*: Harvard Univ Pr.

West, J. and S. Gallagher (2006) "Challenges of open innovation: the paradox of firm investment in open-source software," *R&D Management*, Vol. 36, No. 3, pp. 319–331.

West, J. and S. O'mahony (2008) "The role of participation architecture in growing sponsored open source communities," *Industry & Innovation*, Vol. 15, No. 2, pp. 145–168.