NAIST-IS-MT0851047

修士論文

OSS 開発における時差の分析: OSS 開発者の情報交換への影響

小山 貴和子

2010 年 2 月 4 日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
修士 (工学) 授与の要件として提出した修士論文である。

小山 貴和子

審査委員：

松本 健一 教授	（主指導教員）
関 浩之 教授	（副指導教員）
門田 暁人 准教授	（副指導教員）
大平 雅雄 助教	（副指導教員）

OSS 開発における時差の分析: OSS 開発者の情報交換への影響*

小山 貴和子

内容梗概

OSS は世界中に点在する開発者によって開発され，開発者はタイムゾーンの異なる地域間で情報交換を行う．多数の開発者が参加する大規模な OSS プロジェクトでは，開発者による情報交換を通じた意思決定や合意形成が重要である．しかし，時差の存在する地域間で行われる情報交換はタイムラグを引き起こすと考えられ，タイムラグは迅速な開発の妨げとなる可能性がある．近年，不具合やセキュリティの脆弱性など OSS に発生する問題に対する早急な対応が求められているため，OSS 開発者間の効率的な情報交換を実現するためにタイムラグの実態を調査する必要がある．本研究は，OSS プロジェクト内における情報交換のタイムラグの実態を解明し，情報交換の所要時間を短縮化するための指針を得ることを目的とする．本論文では，OSS 開発における情報交換においてタイムラグが発生する要因に関する仮説と，タイムラグが開発者間の情報交換に与える影響に関する仮説を，複数の OSS プロジェクト対象に検証した．その結果，メッセージの送信先地域の時間が深夜である場合，日中に比べて開発者間の情報交換にタイムラグを引き起こす可能性が高いことがわかった．また，時差の影響を受けずに情報交換を行うことが可能な時間は少なくとも 3 時間存在することがわかった．

キーワード

分散ソフトウェア開発, オープンソースソフトウェア, 情報交換のタイムラグ, 時差分析

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 修士論文, NAIST-IS-MT0851047, 2010 年 2 月 4 日.

Analysis of Time Differences in OSS Development: Impact on Communications among OSS Developers*

Kiwako Koyama

Abstract

OSS (Open Source Software) is developed by globally distributed developers who communicate between different time zones. In a large-scale project where a number of developers are involved, it is important to make a decision and build a consensus through communication among developers. The communication between different time zones, however, would cause a delay of information exchange due to time lag, resulting in impeding rapid development. Since even OSS products are required to quickly respond to issues such as defects and security vulnerabilities, better understandings of the time lag in an OSS project must be constructed in order to facilitate efficient communication among developers. The goal of this thesis is to reveal the existence of the communication delay in OSS project and then to achieve a guide for improving the communication among distributed OSS developers. The thesis frames two hypotheses on factors of the communication delay in an OSS project and on influences of the delay on the communication among developers. As a result of analysis, the thesis found that the communication delay is likely to happen when a messages is received midnight and that there are at least three hours for which developers can communicate each other without the influence of time-lag.

*Master's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-MT0851047, February 4, 2010.

Keywords:

distributed software development, open source software, time-lag communication, time-lag analysis

関連発表論文

研究会・シンポジウム

1. 小山 貴和子, 伊原 彰紀, 柗本 真佑, 亀井 靖高, 大平 雅雄, 松本 健一. OSS 開発における情報交換の効率改善へ向けたタイムラグ分析手法の提案，情報処理学会シンポジウム グループウェアとネットワークサービス・ワークショップ 2009 論文集, Vol.2009, No.8, pp.81–86, 2009.

国際会議

1. Masao Ohira, Kiwako Koyama, Akinori Ihara, Shinsuke Matsumoto, Yasutaka Kamei, and Ken-ichi Matsumoto. A time-lag analysis toward improving the efficiency of communications among oss developers. In *Proceedings of the 3rd International Workshop on Knowledge Collaboration in Software Development (KCSD'09)*, pp.49-62, 2009.

目 次

図 目 次

表 目 次

1. はじめに

1.1 研究の背景と目的

近年のソフトウェア開発企業は，短期間かつ限られたコストの中で高品質なソフトウェアを生産する必要性に迫られている．開発コスト削減のため多くの企業では，低コストで導入可能な Open Source Software (OSS) を自社製品の一部として利用している [25]．OSS はソフトウェア開発企業のみならず行政機関や教育機関においても広く導入されつつある [36]．商用のソフトウェアに劣らない品質と機能を備えた OSS が数多く登場してきており，OSS の社会的影響は今後も拡大していくものと予想される．

OSS 開発は，世界中に点在する開発者が共同開発を行うという分散開発の形態をとる．ただし，OSS の分散開発は企業で実施する一般的な分散開発とは異なる点が多い．例えば OSS 開発では，開発者が開発プロジェクトへの参加・離脱を自由に行うことができる．また，開発が自発的に参加する開発者個々人の自由意志に基づき行われており，個々の開発者が特定の作業や作業に対する責務を負う必要がないという特徴もある [29]．

OSS 開発に限らず分散開発における開発者同士の情報交換は，主に電子メールを代表とする非対面・非同期のコミュニケーション手段が用いられる．このような環境では，開発者同士の情報交換にタイムラグが少なからず発生する．情報交換におけるタイムラグは結果的に，開発スピードや生産性，ソフトウェアの品質など，ソフトウェア開発全体に大きな影響をもたらすとされている [1][5][6][18][23][26]．

特に OSS 開発プロジェクトでは，一部の開発者の多大な貢献に依存する開発が行われている場合が多く [24][32][33]，開発の中心となる開発者らの情報交換におけるタイムラグは OSS 開発に大きな影響を与えるものと考えられる．例えば，Robles らの調査 [30] によると，SourceForge.net[1] における開発者数はアメリカが最も多く，次いで西ヨーロッパ地域，中国となっている．これら 3 つの地域間では少なくとも 5 時間以上の時差があり，中心的な開発者がこれらの地域に分散し

[1] OSS 開発のための開発環境を提供する web サイト．2009 年 2 月現在，23 万以上の OSS プロジェクトが登録されている．

た場合，リアルタイムに情報交換を行うことは容易ではない．企業の分散開発では予め作業範囲を明確化し作業割当をスケジューリングすることで，時差を活用した効率的な分散開発が可能である [3]．一方，OSS 開発は明確な作業割当が存在せず，かつ，世界中に点在する不特定多数の開発者が関与している．そのため，企業における分散開発のように時差を活用した分担作業を行うことが困難である．

さらに，多数の開発者が参加する大規模な OSS プロジェクトでは，情報交換を通じた意思決定や合意形成の重要性が増す．一方で，様々な地域の開発者らの情報交換はタイムラグが発生しやすく，迅速な開発の妨げとなる可能性がある．特に，重大な不具合や脆弱性の修正などといった早急な対応が求められるケースにおいては，情報交換のタイムラグによる意思決定や合意形成の遅延は，ソフトウェアの信頼性のみならずユーザからの信頼を損なう可能性がある．

本研究の目的は，OSS プロジェクト内における情報交換のタイムラグの実態を解明し，情報交換の所要時間を短縮化するための指針を得ることである．本論文では，OSS 開発における情報交換においてタイムラグが発生する要因に関する仮説と，タイムラグが開発者間の情報交換に与える影響に関する仮説を，複数の OSS プロジェクトを分析することにより検証する．

1.2 論文構成

本論文の構成は以下の通りである．続く 2 章では OSS 開発における情報交換とタイムラグについて述べる．3 章では情報交換のタイムラグに関する 3 つの仮説について述べ，4 章は分析方法について説明する．5 章では複数の OSS 開発プロジェクトを対象としたケーススタディについて述べ，6 章で分析結果を報告する．7 章では分析結果に対する考察と本論文の制約について議論を行う，8 章では関連研究を述べ本研究の立場を明らかにし，最後に 9 章で本論文の結論について述べる．

2. OSS 開発における情報交換とタイムラグ

本章では本論文が対象とする OSS 開発における情報交換とタイムラグについて述べる．2.1 節では開発者の情報交換手段について説明し，2.2 節では情報交換におけるタイムラグについて述べる．次に，2.3 節では情報交換のタイムラグの実態に関する予備調査を報告し，2.4 節では情報交換のタイムラグを解消するための要件について述べる．

2.1 OSS 開発の情報交換手段

OSS は世界中に点在する開発者によって開発されるため，開発者は情報交換手段として非対面でかつ非同期な媒体である Mailing List (ML) や掲示板を主に用いている．特に ML は，開発者がいつでもメッセージを読むことができるという特徴を持つため，ほとんどの OSS プロジェクトで利用されている．

ML は，送信されたメッセージが参加者全員に配信される．世界中に開発者が点在している OSS 開発では，常に誰かが開発に従事しておりメッセージを読んでいるため，素早い情報交換が可能となり得る．しかしながら，不特定多数の開発者が参加する OSS 開発では，送信されたメッセージに対して誰が返信を行うか定かではない．また，非同期な媒体は情報交換を行う双方のタイミングが考慮されず一方的にメッセージの送信が可能なため，返信者のタイミングと合致しなければリアルタイムな情報交換を行うことは容易ではない．

2.2 情報交換におけるタイムラグ

従来の研究では OSS 開発が成功する要因の一つとして，開発者が世界中に点在していることを挙げている [8][17][35]．開発者が世界中に点在することで 24 時間体制で開発を行うことが可能となり，開発者間の情報交換や不具合修正，脆弱性の修正などの対応が迅速という主張をしている．従来研究では，開発者間の情報交換にはタイムラグが存在しないものと考えられてきた．

しかしながら，OSS 開発の多くは一部の開発者を中心に開発が行われていることが明らかになりつつある [24][30][32]．さらに，開発者らは世界中に均等に分散して開発を行っているのではなく，特定の地域に集中するなど大きな偏りが存在する分散体制となることが多い．例えば，Tang らの調査 [33] によると，PostgreSQL プロジェクトと GTK+プロジェクトの開発者数はアメリカが約 30%，ドイツが約 10%を占め，アメリカ，EU，カナダに在住する開発者が多い．このように，特定の中心的開発者が局所的に分散する体制で行われる OSS 開発では，開発者間の情報交換にタイムラグが発生しやすいと考えられる．

2.3 タイムラグの実態に関する予備調査

本研究を実施するにあたり Python プロジェクトの情報交換におけるタイムラグの実態を予備的に調査した．ML 上でのアメリカ在住の開発者とヨーロッパ在住の開発者との情報交換の様子を観察した結果，ヨーロッパの現地時刻 1 時頃（アメリカの現地時刻 19 時頃）にアメリカ在住の開発者がメッセージを送信しても，ヨーロッパ在住の開発者からすぐに返信されることはなく，ヨーロッパの現地時刻 9 時頃になってから返信されるケースが多く見られた．一例を図 1 に示す．

メール本文に “Good morning.” と記述されていることと，朝になってからメッセージが返信されていることから，ヨーロッパ在住の開発者はアメリカ在住の開発者がメッセージを送信した時間帯には就寝していたものと考えられる．図 1 の例では，返信までに要した時間（**返信時間**）は約 8 時間であり，時差の存在する地域に居住する開発者間の情報交換は，情報交換を行う時間帯が異なるためタイムラグが発生しやすい．

2.4 タイムラグ解消のための要件

2.2 節で述べたように，特定の中心的開発者が局所的に分散する体制で行われる OSS 開発では，開発者間の情報交換にタイムラグが発生しやすいと考えられる．情報交換のタイムラグを解消するためには，**開発者の多くが在住する地域を把握すること**が重要となってくる．開発者の地域を把握することで，開発者は在

図 1　情報交換の一例

住する開発者が多い地域に合わせて情報交換を行うことが可能となり，リアルタイムに情報交換を行うことが可能となる．

2.3 節で述べたように，時差のある地域間における情報交換は，情報交換を行う時間帯が異なるためタイムラグが発生しやすい．情報交換のタイムラグを解消するためには，開発者の地域を把握することだけでなく，**双方が頻繁に情報交換を行う時間帯を把握すること**が重要となってくる．双方が情報交換を行う時間帯を把握することで，それぞれの地域に合わせて情報交換を行うことが可能となり，リアルタイムに情報交換を行うことが容易となる．

さらに，ヨーロッパや北アメリカに在住する OSS 開発者を中心にアンケートを行った FLOSS-US の調査 [7] によると，OSS 開発者の多くは，OSS 開発と関係のない企業に勤務しているため，勤務終了後や週末にかけて OSS 開発に従事することが多い．OSS 開発者の 75%は 1 週間あたり 7 時間，すなわち，1 日 1 時間程度開発に従事している．開発者はこの開発従事時間に情報交換を行っていると考えられるが，開発にはコーディングや不具合修正などの作業も含まれるため，情報交換を行うことができる時間はさらに短い．たとえ時差のない地域間の情報交換であっても，開発者個々人の限られた開発に従事する時間を 1 日の中で合わせることは困難であり，開発者間の情報交換にタイムラグが発生しやすいと考えら

れる．情報交換のタイムラグを解消するためには，開発者の地域や情報交換を行う時間帯を把握することだけでなく，**素早い返信が期待できる時間帯を把握すること**が重要となってくる．素早い返信が期待できる時間帯を把握することで，リアルタイムに情報交換を行うことが容易となる．

本論文では，時差の有無のみが情報交換のタイムラグを発生させる要因ではなく，開発者の多くが在住する地域が局所的であること，長時間開発に従事できる開発者が少なく，多くの開発者が開発に従事できる時間がわずかであること，情報交換を行う時間帯が異なることなど，様々な要因が作用して情報交換にタイムラグが発生すると考える．情報交換におけるタイムラグの実態把握へ向けて，次章では，タイムラグが発生する要因とタイムラグが情報交換に与える影響についての本研究の仮説について述べる．

3. 情報交換のタイムラグに関する仮説

本章では情報交換のタイムラグから導いた仮説について述べる．3.1 節では情報交換のタイムラグが発生する要因に関する仮説について述べる．次に，3.2 節ではタイムラグが開発者間の情報交換に与える影響に関する 2 つの仮説について述べる．

3.1 情報交換のタイムラグが発生する要因

2.4 節で述べたように，OSS 開発者の多くは OSS 開発と関係のない企業に勤務し，勤務終了後，OSS 開発に従事する．OSS 開発以外の場で勤務している開発者（**ボランティアの開発者**）が多く在住する地域では，夕方から夜にかけて情報交換が活発に行われていると考えられる．一方，OSS 開発を仕事として企業に勤務している OSS 開発者も存在する．OSS 開発を仕事としている開発者（**企業の開発者**）が多く在住する地域では，勤務時間帯に情報交換を行っていると考えられ，朝から夕方にかけて情報交換が活発に行われていると考えられる．朝を中心に情報交換が行われている地域や，夜を中心に情報交換が行われている地域など，地域によって情報交換を行う時間の違いが情報交換のタイムラグを発生させる要因であると考え，次の仮説を導いた．

仮説 1: 地域によって情報交換を行う時間帯が異なる

3.2 タイムラグが情報交換に与える影響

情報交換を行う開発者間に時差がある場合と時差がない場合とでは，時差がない場合の方が開発に従事する時間帯を一致しやすく，リアルタイムに情報交換を行うことが容易となり，返信時間が短くなると考えられる．例えば，アメリカとヨーロッパでは少なくとも 6 時間以上の時差があり，これらの地域間では開発に従事する時間帯が異なる可能性が高く，返信時間が長くなると考えられる．

さらに，情報交換のタイムラグを解消するためには，返信者側の OSS 開発に従事する時間帯を考慮して，メッセージを送信する必要があると考えられる．例え

ば，時差のある地域間の情報交換では，送信者が日中，返信者が深夜となる時間帯が存在する場合もあり，このようなケースでは朝や夜にメッセージを送信すると素早い返信が期待できると考える．一方，時差のない地域間の情報交換では，日中にメッセージを送信すると素早い返信が期待できると考える．つまり，開発者の情報交換は返信者によって素早い返信が期待できる時間が異なると考えられる．

本論文では，開発者間に時差があると返信時間を長くさせ，情報交換にタイムラグが発生する可能性が高くなると考えた．また，素早い返信が期待できる時間帯にメッセージを投稿すれば，情報交換のタイムラグを解消することができると考え，次の仮説を導いた．

仮説 2: 時差のある地域間の情報交換は返信時間が長くなる
仮説 3: 返信者の地域によって素早い返信を期待できる時間帯が異なる

4. 分析方法

本章では，分析に先立ち準備する必要のあるデータの収集と整形の方法，および，仮説を検証するための分析方法，分析の観点について述べる．4.1 節では分析データの収集方法と整形方法について説明する．次に，4.2 節では仮説の分析方法と仮説の検証方法について説明し，4.3 節では分析の観点に関する OSS プロジェクトの分類について説明する．

4.1 分析データの収集と整形

本論文では，OSS 開発に携わる開発者が情報交換のために利用している ML などのアーカイブを対象とする．多くの OSS プロジェクトが情報交換に用いている ML を分析することで，OSS 開発者間のタイムラグの実態を明らかにすることができると考える．

本論文では，まずメッセージの送信日時と送信場所のデータを収集する．送信日時とは送信者の現地時刻を指し，送信場所とは協定世界時（UTC: Coordinated Universal Time）と現地時刻との時差で表す．例えば，日本からメッセージを送信した場合の標準時は UTC より 9 時間進んでいるため “UTC+9” と表す．

図 2 は ML 利用時の送信メッセージと返信メッセージを収集する方法を示した概略図である．図 2(a) はメッセージのスレッド構造を表し，図 2(b) はメッセージの送信日時を表す．メッセージのスレッド構造と送信日時からメッセージの送信と返信の関係を図 2(c) に示す．mail A に対し mail B が返信され，さらに mail B に対し mail C が返信される場合（図 2(a)），図 2(b) を基に送信と返信の関係を図 2(c) のように一行（送信情報と返信情報の対）で表す．ただし，mail E のように返信が行われていないメッセージは分析対象外とするため，図 2(c) に現れない．以降，地域 A から送信されたメッセージに対して，地域 B から返信したメッセージの対を “A → B” と示す．また，収集した送信情報と返信情報を用いて，返信時間を求める（図 2(c) の最右列）．

(a) メッセージのスレッド構造

mail A : _____
- **mail B** : Re:_____
  - **mail C** : Re:Re:_____
- **mail D** : Re:_____

mail E : _____

(b) メッセージの送信日時

ID	送信日時	送信場所	協定世界時(UTC)
mail A	2009/04/29 10:33:53	UTC＋9	2009/04/29 01:33:53
mail B	2009/04/28 22:40:04	UTC－4	2009/04/29 02:40:04
mail C	2009/04/29 09:12:30	UTC＋3	2009/04/29 06:12:30
mail D	2009/04/29 02:26:59	UTC－10	2009/04/29 12:26:59
mail E	2009/04/29 08:09:27	UTC＋7	2009/04/29 15:09:27

(c) メッセージの送信・返信関係表

送信情報			返信情報			表記方法	返信時間（時）
ID	送信日時	送信場所	ID	送信日時	送信場所		
mail A	2009/04/29 10:33:53	UTC＋9	mail B	2009/04/28 22:40:04	UTC－4	A→B	1.10
mail B	2009/04/28 22:40:04	UTC－4	mail C	2009/04/29 09:12:30	UTC＋3	B→C	3.54
mail A	2009/04/29 10:33:53	UTC＋9	mail D	2009/04/29 02:26:59	UTC－10	A→D	10.88

図 2 分析データの整形方法

4.2 仮説の分析と検証

本節では，3 章で述べた情報交換のタイムラグが発生する要因と，タイムラグが情報交換に与える影響に関する 3 つの仮説それぞれについて，分析方法と検証方法を述べる．4.2.1 項では仮説 1，4.2.2 項では仮説 2，4.2.3 項では仮説 3 を検証するための分析方法を説明する．

4.2.1 開発者の地理的分布と情報交換を行う時間帯

仮説 1 を検証するために，まずタイムゾーン（UTC-11〜UTC+12）別にメッセージ数を求め，OSS 開発者の地理的分布を把握する．OSS 開発者の地理的分布からメッセージ数の多い地域を分析対象の地域とする．そして，分析対象の地域毎に各時刻のメッセージ数の分布を確認する．各時刻におけるメッセージ数の分布を確認することで，開発者が情報交換を最も行う時間帯や最も行わない時間帯をそれぞれ把握することが可能となる．なお，情報交換を行う時間帯が地域によって異なれば，仮説 1 が支持されたとみなす．

4.2.2 情報交換の所要時間

仮説 2 を検証するために，同一タイムゾーン内（時差のない地域間）の返信時間と異なるタイムゾーン間（時差のある地域間）の返信時間の分布を確認し，時差の有無による返信時間の差が統計的に有意かどうかを検定する．検定にはウィルコクソンの順位和検定を用いる．時差の有無別に返信時間の分布を確認することで，情報交換のタイムラグに時差が影響するかを把握することが可能となる．なお，時差のある地域間の方が返信時間が長くなれば，仮説 2 が支持されたとみなす．

ただし，ML などの非同期な媒体を用いた情報交換は，開発者の都合の良いときに送信されたメッセージに返信することができるため，メッセージが送信されてから何日も経過した後に返信される場合もある．返信時間が長いケースは，送信されたメッセージがすぐに返信できない内容である場合，時差の影響や開発者個々人の都合によりすぐに返信されない場合などが，返信時間を長くした要因であると考えられる．本研究では，時差による情報交換のタイムラグを把握するため，返信時間が 24 時間未満のものを対象とする．返信時間が 24 時間未満のケースでは，地理的要因（時差）が影響して情報交換のタイムラグを発生させているものが多く存在すると考えられる．一方，返信時間が 24 時間以上のケースでは，地理的要因よりも開発者個々人の都合といった人的要因が大きく影響して情報交換のタイムラグを発生させているものが多く存在すると考えられる．

4.2.3 素早い返信が期待できる時間帯

仮説 3 を検証するために，各時刻における返信時間の分布を示し，返信の早い時間帯を確認する．具体的には，0 時に送信されたメッセージに対する返信時間の分布や 12 時に送信されたメッセージに対する返信時間の分布など，各送信時刻に対してそれぞれの返信時間の分布を確認する．特に，日本→日本や日本→アメリカなど地域別に返信時間を分析し，各地域のリアルタイムに情報交換を行うことができる時間帯を確認する．地域別に返信時間を分析することで，返信者の地域によって素早い返信が期待できる時間帯が異なるかを把握することが可能となる．なお，本項においても返信時間が 24 時間未満のものを対象とする．返信

者の地域によって素早い返信を期待できる時間帯が異なれば，仮説 3 が支持されたとみなす．

4.3 分析の観点: OSS プロジェクトの分類

OSS 開発の中には開発当初，少数の開発者により発足され，その後大きく発展したケースや，企業のソフトウェア開発がオープンソースへと移り変わったケース，OSS 開発が企業のソフトウェア開発へと移り変わったケースなど様々なケースに分類できる．3.1 節で述べたように，OSS 開発にはボランティアの開発者や企業の開発者が存在するため，開発者が所属する環境によって情報交換を行う時間帯が異なると考えられる．開発当初少数の開発者により発足され，その後大きく発展したプロジェクトではボランティアの開発者が多く，企業のソフトウェア開発がオープンソースへと移り変わったプロジェクトでは企業の開発者が多いと考えられる．また，大規模なプロジェクトではソフトウェアシステムを小さな独立した機能の単位（**モジュール**）毎に分割し，開発が行われている場合がある．そのため，同じプロジェクトであってもモジュール毎に開発担当者が異なる場合があり，情報交換を行う時間帯も異なると考えられる．

本論文では分析対象プロジェクトを，OSS 開発プロジェクト，企業と混合の OSS 開発プロジェクト，モジュール毎に分かれた OSS 開発プロジェクトの 3 つのタイプに OSS プロジェクトを分類し，この分類を基に分析と考察を行う．開発当初少数の開発者により発足され，その後大きく発展したプロジェクトを **OSS 開発プロジェクト**とする．さらに，企業のソフトウェア開発がオープンソース化と移り変わったプロジェクトを**企業と混合の OSS 開発プロジェクト**とする．そして，モジュール毎に情報交換を行う方法が分けられているプロジェクトを**モジュール毎に分かれた OSS 開発プロジェクト**とする．

5. ケーススタディ

本章ではOSSプロジェクトを対象としたケーススタディについて述べる．5.1節では分析対象プロジェクトについて説明し，5.2節ではOSSプロジェクトの分類について述べる．次に，5.3節では分析対象データの基本統計量について述べる．

5.1 分析対象プロジェクト

数多くの開発者が参加する大規模なプロジェクトを分析対象とすることにより，OSSプロジェクト内の情報交換におけるタイムラグの実態を明らかにできると考え，以下のプロジェクトを分析対象とする．

Pythonプロジェクト[28]（以降，Python）

1990年にGuido van Rossumによって開発され，ABC言語の後続言語として発足したオブジェクト指向のスクリプト言語の開発プロジェクトである．PythonはPerlとともに欧米に広く普及しており，多くのプラットホームをサポートし，豊富なドキュメントや豊富なライブラリがあることから，Webプログラミング，GUIベースのアプリケーション，CAD，3Dモデリング，数式処理等幅広い分野で使用されている．

PostgreSQLプロジェクト[27]（以降，PostgreSQL）

BSDライセンスによって配布されているオブジェクト関係データベース管理システム（ORDBMS）の開発プロジェクトである．1986年にカリフォルニア大学バークレー校（University of California, Berkeley）でMichael StonebrakerによってPOSTGRESプロジェクトとして発足し開発されたものが，1995年にオープンソース版（Postgres95）としてWeb上に公開された後、1996年に名称をPostgreSQLに変更して1997年1月にver.6.0として公開され現在まで開発が続いている．

Apache HTTP Serverプロジェクト[2]（以降，Apache）

1995年にNCSA (National Center for Supercomputing Applications) のRobert McCoolらによって開発されたWebサーバ (NCSA HTTPd) にパッチを提供していた開発者らによって発足されたプロジェクトである．NCSA HTTPd 1.3をベー

スに障害修正や新機能の追加を繰り返し行い，1996 年に Apache 1.0 が公開され，現在は ver.1.3 系，ver.2.0 系，ver.2.2 系の開発が並行して進められている．

Eclipse プロジェクト[13]（以降，Eclipse）

IBM (International Business Machines Corporation) によって開発された統合開発環境 (IDE: Integrated Development Environment) の開発プロジェクトである．1998 年 11 月に IBM カナダでプロジェクトが開始されたが，2001 年 11 月にオープンソース化し，現在 50 以上のサブプロジェクトの開発が並行して進められている．本論文では，Eclipse の中心プロジェクトやダウンロード数の多いプロジェクトの中から，メッセージ数が 2,000 件以上のプロジェクトを対象とし，各プロジェクトを以下に記す．なお，Eclipse Platform プロジェクトはモジュール毎に分かれており，それぞれを分析する．

- Java development tools プロジェクト [14]（以降，JDT）
- Plugin Development Environment プロジェクト [15]（以降，PDE）
- Business Intelligence and Reporting Tools プロジェクト [9]（以降，BIRT）
- Eclipse Modeling Framework プロジェクト [11]（以降，EMF）
- C/C++ Development Tooling プロジェクト [10]（以降，CDT）
- Eclipse Web Tools Platform Project プロジェクト [16]（以降，WTP）
- Eclipse Platform プロジェクト [12]（以降，Eclipse Platform）
  - Platform
  - SWT
  - RCP

5.2 プロジェクトの分類

4.3 節で述べたように，本論文では分析を行う観点として OSS プロジェクトを OSS 開発プロジェクト，企業と混合の OSS 開発プロジェクト，モジュール毎に分

表 1 OSS プロジェクトの分類

OSS 開発プロジェクト	企業と混合の OSS 開発プロジェクト	モジュールに分かれた OSS 開発プロジェクト
	Java development tools (JDT)	
Python	Plugin Development Environment (PDE)	Platform
PostgreSQL	Business Intelligence and Reporting Tools (BIRT)	SWT
Apache HTTP Server (Apache)	Eclipse Modeling Framework (EMF)	RCP
	C/C++ Development Tooling (CDT)	
	Eclipse Web Tools Platform Project (WTP)	

かれた OSS 開発プロジェクトの 3 つのタイプに分類する．表 1 に各プロジェクトと OSS プロジェクトの分類を示す.

5.3 基本統計量

分析対象データとしては，主に開発に関する話題（新規機能，バグ報告，メンテナンス，リリースなど）を取り扱う ML またはニュースグループのアーカイブを用いる．各プロジェクトに対する分析対象 ML/ニュースグループの一覧および，分析期間，全メッセージ数，ML/ニュースグループ参加者数，送信・返信の対の数，返信時間が 24 時間未満のデータ数に関するそれぞれの統計量を表 2 に示す.

本研究では，送信日時，送信場所，どのメッセージへ返信したか（返信情報）が含まれていないメッセージを対象外とした．また，表 2 における開発者数は，ML やニュースグループに一度でも登場した人物の総数とする.

表 2 基本統計量

分析対象プロジェクト			分析対象 ML/ ニュースグループ	分析期間	全メッセージ数（件）	開発者数（人）	送信・返信の対（件）	返信 24 時間未満のデータ数（件）
Python			python-dev@python.org	1999/04〜2009/04	89,301	2,150	56,707	51,830
PostgreSQL			pgsql-hackers@postgresql.org	1997/01〜2009/09	184,492	4,731	130,239	114,514
Apache HTTP Server (Apache)			dev@httpd.apache.org	1995/03〜2009/08	119,673	2,736	63,293	55,459
Eclipse	Java development tools (JDT)		eclipse.tools.jdt	2003/05〜2009/09	24,691	4,717	7,894	5,951
	Plugin Development Environment (PDE)		eclipse.platform.pde	2008/05〜2009/09	2,238	578	847	627
	Business Intelligence and Reporting Tools (BIRT)		eclipse.birt	2004/09〜2009/09	35,108	3,886	9,058	7,798
	Eclipse Modeling Framework (EMF)		eclipse.tools.emf	2002/10〜2009/09	44,216	3,337	22,654	20,259
	C/C++ Development Tooling (CDT)		eclipse.tools.cdt	2002/01〜2009/09	19,088	4,290	4,727	3,296
	Eclipse Web Tools Platform Project (WTP)		eclipse.webtools	2003/07〜2009/09	19,160	3,658	6,193	4,445
	Eclipse Platform	Platform	eclipse.platform	2003/05〜2009/09	7,628	1,502	1,152	917
		SWT	eclipse.platform.swt	2003/05〜2009/09	45,490	6,963	10,377	8,340
		RCP	eclipse.platform.rcp	2004/09〜2009/09	37,583	4,796	13,614	10,243

6. 分析結果

本章では3つの仮説を検証するために行った分析結果について報告する．6.1節では情報交換のタイムラグが発生する要因に関する仮説1の分析結果について報告し，6.2節では時差が開発者の情報交換に与える影響に関する仮説2，仮説3の分析結果について報告する．

6.1 情報交換のタイムラグが発生する要因

仮説 1: 地域によって情報交換を行う時間帯が異なる

本論文では，各プロジェクトに対する分析対象地域，各地域のメッセージ数と開発者数を表3に示す．なお，各タイムゾーンにおけるメッセージ数の分布は付録Aの図5，図6より，メッセージ数の多い2地域を分析対象地域とし，横軸は各タイムゾーンを表し，縦軸はメッセージ数を表す．以降，本論文ではUTC-8〜UTC-4に該当する地域を**米大陸地域**，UTC+0〜UTC+3に該当する地域を**欧・アフリカ地域**とする．

分析対象地域における各時刻のメッセージ数を図3，図4に示す．図3，図4の縦軸はUTC+0の時刻を表し，横軸は各時刻のメッセージ数を表す．なお，各図の左側は米大陸地域を表し，右側は欧・アフリカ地域を表す．

OSS開発プロジェクトに着目すると，図3よりPythonでは，米大陸地域と欧・アフリカ地域の両地域で情報交換を活発に行っている時間帯が同じであることを確認できる．これはPostgreSQL，Apacheでも同様の結果がみられ，米大陸地域の開発者と欧・アフリカ地域の開発者が同じ時間帯に情報交換を活発に行っていることがわかる．

企業と混合のOSSプロジェクトに着目すると，図3，図4よりJDT，EMFでは米大陸地域と欧・アフリカ地域の両地域で情報交換を活発に行っている時間帯が同じであるが，PDE，BIRT，CDT，WTPでは情報交換を活発に行っている時間帯が異なることを確認できる．しかしながら，企業と混合のOSSプロジェクトはそれぞれの現地時刻の朝9時頃（米大陸地域はUTC+0の13時頃，欧・アフリカ地域はUTC+0の7時頃) からメッセージ数が多くなっている．また，米大陸

地域では情報交換が活発に行われている時間帯が OSS 開発プロジェクトに比べて短く，UTC+0 の 23 時から 13 時頃（米大陸地域の 19 時から 9 時頃）にかけてメッセージ数が極めて少なくなっていることがわかる．

モジュール毎に分かれた OSS 開発プロジェクトに着目すると，図 4 より米大陸地域と欧・アフリカ地域で情報交換を活発に行っている時間帯が異なることを確認でき，それぞれの現地時刻の朝 9 時頃（米大陸地域は UTC+0 の 13 時頃，欧・アフリカ地域は UTC+0 の 7 時頃) からメッセージ数が多くなっている．また，企業と混合の OSS 開発プロジェクトと同様に，米大陸地域では情報交換が活発に行われている時間帯が，UTC+0 の 0 時から 12 時頃（米大陸地域の 20 時から 8 時頃）にかけてメッセージ数が極めて少なくなっていることがわかる．

図 3，図 4 より，全てのプロジェクトにおいて，メッセージ数が最小となる時間帯はそれぞれの現地時刻の深夜から早朝にかけての時間帯であった．このことより，OSS 開発者は深夜から早朝となる時間帯に情報交換をあまり行わないことがわかる．

メッセージが最大となる時間帯や最小となる時間帯はそれぞれで異なるため，各地域でメッセージ数が最大となる時間帯と最小となる時間帯を表 4 にまとめる．図 3，図 4，表 4 より，情報交換を活発に行っている時間帯が同じになるプロジェクトもあるが，メッセージ数の分布は一致しておらず，情報交換を行う時間帯は地域によって異なることが読み取れるため，仮説 1 は支持されたといえる．

表 3 分析対象地域と統計量（米: 米大陸地域，欧: 欧・アフリカ地域）

分析対象プロジェクト			分析対象地域		メッセージ数（件）		開発者数（人）	
			米	欧	米	欧	米	欧
Python			UTC-8〜UTC-4	UTC+0〜UTC+3	28,574	17,748	679	575
PostgreSQL			UTC-5〜UTC-4	UTC+0〜UTC+3	61,875	27,318	537	795
Apache HTTP Server (Apache)			UTC-8〜UTC-4	UTC+0〜UTC+2	41,302	12,840	598	451
Eclipse	Java development tools (JDT)		UTC-5〜UTC-4	UTC+1〜UTC+2	2,468	2,547	226	320
	Plugin Development Environment (PDE)		UTC-5〜UTC-4	UTC+1〜UTC+2	321	193	28	68
	Business Intelligence and Reporting Tools (BIRT)		UTC-5〜UTC-4	UTC+1〜UTC+2	4,042	1,401	152	287
	Eclipse Modeling Framework (EMF)		UTC-5〜UTC-4	UTC+1〜UTC+2	13,807	5,219	165	510
	C/C++ Development Tooling (CDT)		UTC-5〜UTC-4	UTC+1〜UTC+2	872	1630	119	258
	Eclipse Web Tools Platform Project (WTP)		UTC-5〜UTC-4	UTC+1〜UTC+2	2,302	1,225	171	285
	Eclipse Platform	Platform	UTC-4	UTC+2	431	398	64	100
		SWT	UTC-8〜UTC-4	UTC+1〜UTC+2	3,248	4,517	410	613
		RCP	UTC-5〜UTC-4	UTC+1〜UTC+2	4,269	4,354	202	570

図 3 情報交換を行う時間帯 - 1

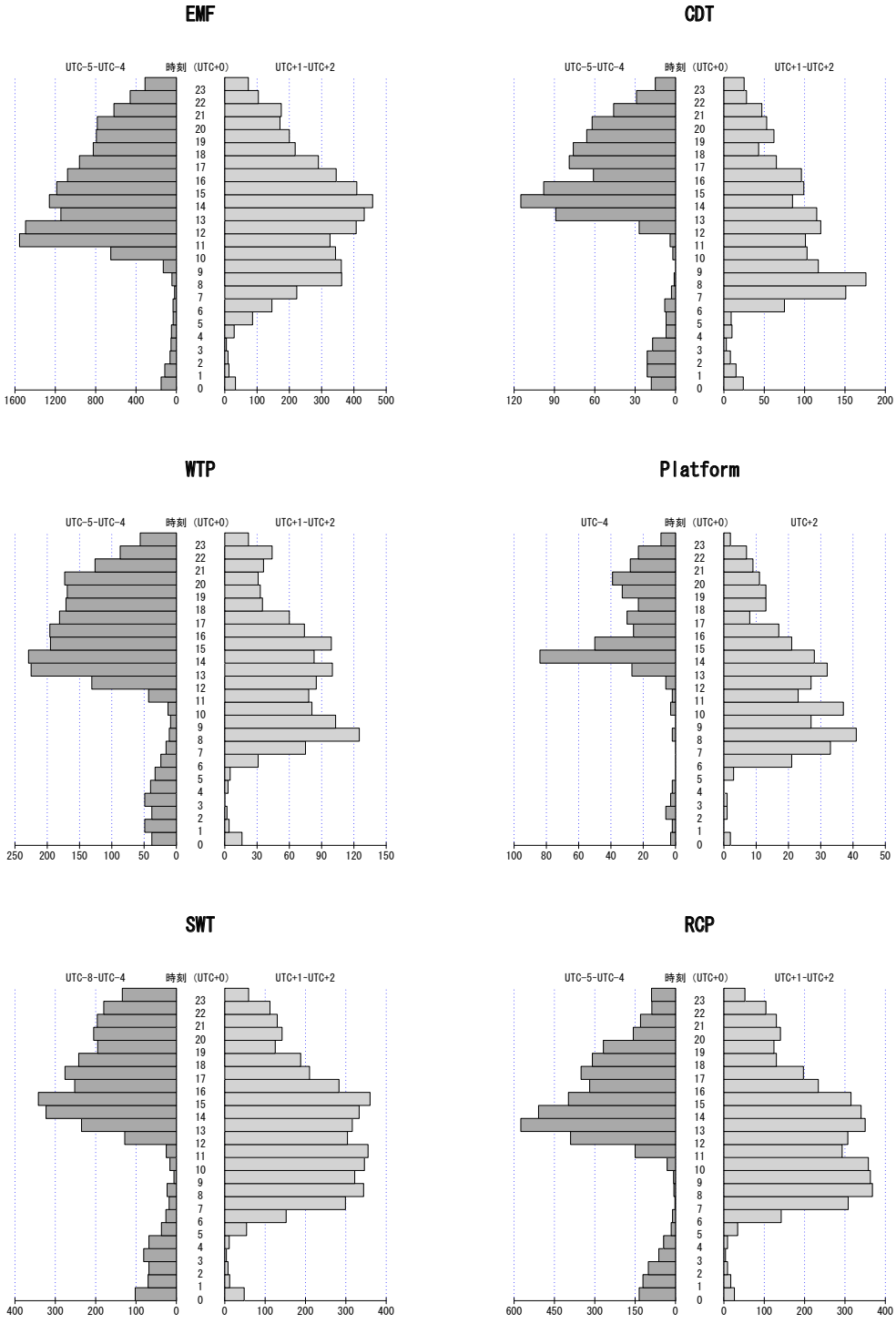

図 4 情報交換を行う時間帯 - 2

表 4 メッセージ数が最大/最小となる時間帯（米: 米大陸地域，欧: 欧・アフリカ地域）

分析対象プロジェクト			最大となる時間帯（UTC+0）		最小となる時間帯（UTC+0）	
			米	欧	米	欧
Python			17（時）	21（時）	9（時）	3（時）
PostgreSQL			15	15	9	3
Apache HTTP Server (Apache)			18	15	10	3
Eclipse	Java development tools (JDT)		14	14	8	5
	Plugin Development Environment (PDE)		15	16	8〜9	2〜4，23
	Business Intelligence and Reporting Tools (BIRT)		15	7	8〜9	2〜3
	Eclipse Modeling Framework (EMF)		11	14	7	3
	C/C++ Development Tooling (CDT)		14	8	9	3
	Eclipse Web Tools Platform Project (WTP)		14	8	9	3
	Eclipse Platform	Platform	14	8	5〜7，9	1，4
		SWT	15	15	9	3
		RCP	13	8	7	3

6.2 タイムラグが情報交換に与える影響

仮説 2: 時差のある地域間の情報交換は返信時間が長くなる

時差のない地域間と時差のある地域間における返信時間の中央値と有意水準5%のウィルコクソンの順位和検定による P 値を表 5 に示す．ただし，返信時間に有意差がみられなかった P 値を太字で表す．

OSS 開発プロジェクトに着目すると，表 5 より，Python の欧→欧では返信時間の中央値が 1.59 時間であり，欧→米では 1.80 時間であることを確認できる．欧→米の方が返信時間は長いが，これらの返信時間に有意差はみられなかった．これは PostgreSQL でも同様の結果を確認できる．これらの結果より OSS 開発プロジェクトでは，欧→欧と欧→米の返信時間の差に有意差がみられないことがわかる．米大陸地域の開発者が送信した場合は，情報交換のタイムラグに時差が影響し，欧・アフリカ地域の開発者が送信した場合は，情報交換のタイムラグに時差が影響しないといえる．

企業と混合の OSS 開発プロジェクトに着目すると，表 5 より，JDT，EMF，WTP においても，欧→欧と欧→米の返信時間の差に有意差がみられなかった．また，PDE では米→米と米→欧の返信時間の差に有意差がみられなかった．企業と混合の OSS 開発プロジェクトでは，米大陸地域の開発者が送信した場合は，情報交換のタイムラグに時差が影響しているといえる．しかしながら，欧・アフリカ地域の開発者が送信した場合は，プロジェクトによって結果が異なるため，情報交換のタイムラグに時差が影響しているとは必ずしもいえない．

モジュール毎に分かれた OSS 開発プロジェクトに着目すると，表 5 より，全てのプロジェクトにおいて返信時間の差に有意差がみられ，情報交換のタイムラグに時差が影響しているといえる．

これらの結果より，米大陸地域の開発者が送信した場合は時差が影響して情報交換にタイムラグを発生させているといえ，欧・アフリカ地域の開発者が送信した場合は時差が影響して情報交換にタイムラグを発生させているとは限らないといえる．したがって，仮説 2 は弱くではあるが支持されたといえる．

表 5 地域別の返信時間（米: 米大陸地域，欧: 欧・アフリカ地域）

分析対象プロジェクト			米→米（時）	米→欧（時）	P 値*	欧→欧（時）	欧→米（時）	P 値*
Python			1.24	2.07	0.00	1.59	1.80	**0.57**
PostgreSQL			0.76	1.88	0.00	1.19	1.42	**0.95**
Apache HTTP Server (Apache)			0.87	1.75	0.00	1.30	1.70	0.00
Eclipse	Java development tools (JDT)		1.91	6.88	0.00	2.75	3.39	**0.91**
	Plugin Development Environment (PDE)		2.10	3.86	**0.17**	1.69	5.12	0.00
	Business Intelligence and Reporting Tools (BIRT)		2.33	10.41	0.00	1.48	4.72	0.00
	Eclipse Modeling Framework (EMF)		0.93	1.72	0.00	0.97	1.00	**0.69**
	C/C++ Development Tooling (CDT)		2.16	8.56	0.00	2.42	4.70	0.02
	Eclipse Web Tools Platform Project (WTP)		2.06	6.13	0.00	3.04	4.27	**0.15**
	Eclipse Platform	Platform	1.88	4.28	0.03	2.12	3.56	0.00
		SWT	2.42	3.25	0.00	0.96	3.42	0.00
		RCP	2.90	3.54	0.01	1.52	4.38	0.00

*P 値: 有意水準 5%によるウィルコクソンの順位和検定

仮説 3: 返信者の地域によって素早い返信を期待できる時間帯が異なる

各地域の素早い返信が期待できる時間帯を表 6 に示す．各時間帯は送信地域の現地時刻で表す．各時刻における返信時間の分布は付録 B の図 7 から図 18 に示す．図 7 から図 18 の横軸は送信地域の現地時刻を表し，縦軸は各返信時間におけるメッセージ数の割合を表す．Eclipse はメッセージ数が少なかったため，横軸の時刻を 3 時間毎で表す．“0 - 0.5” は返信時間が 0.5 時間未満を表し，“8 - 24” は返信時間が 8 時間以上 24 時間未満を表す．

表 6，図 8 より，PostgreSQL の米大陸地域の 2 者間で情報交換を行う場合（米→米），10 時から 19 時頃にメッセージを送信すると約 60%の確率で送信してから 1 時間未満に返信される．一方，米大陸地域から欧・アフリカ地域に対してメッセージを送信する場合（米→欧），米大陸地域の 5 時から 12 時頃にメッセージを送信すると約 50%の確率で送信してから 1 時間未満に返信される．よって，米→米で素早い返信が期待できる時間帯は米大陸地域の 10 時から 19 時頃であり，米→欧では米大陸地域の 5 時から 12 時頃であり，それぞれ返信者の地域が異なると素早い返信が期待できる時間帯も異なることがわかる．同様に，PostgreSQL の欧→欧で素早い返信が期待できる時間帯は欧・アフリカ地域の 10 時から 18 時頃，欧→米では欧・アフリカ地域の 16 時から 0 時頃であることを確認でき，返信者の地域によって素早い返信を期待できる時間帯が異なることがわかる．

プロジェクトによって素早い返信が期待できる時間帯は異なるが，その他のプロジェクトにおいても同様の結果がみられた．OSS 開発プロジェクトでは，図 7 から図 9 より，返信地域の多くの開発者が就寝している可能性の高い深夜から早朝にかけた時間帯にメッセージを送信した場合に返信が遅くなる傾向がわかる．企業と混合の OSS 開発プロジェクト，モジュール毎に分かれた OSS 開発プロジェクトでは，図 16 から図 18 より，返信地域の夕方から早朝にかけた時間帯にメッセージを送信した場合に返信が遅くなる傾向がわかる．これらの結果から，返信者の地域が異なると返信の早い時間帯も異なり，仮説 3 は支持されたといえる．

表 6 素早い返信が期待できる時間帯（米: 米大陸地域，欧: 欧・アフリカ地域）

分析対象プロジェクト			返信の早い時間帯（現地時刻）			
			米→米	米→欧	欧→欧	欧→米
Python			9〜20 (時)	7〜16 (時)	14〜23 (時)	16〜2 (時)
PostgreSQL			10〜19	5〜12	10〜18	16〜0
Apache HTTP Server (Apache)			10〜19	5〜13	9〜19	18〜1
Eclipse	Java development tools (JDT)		9〜21	3〜15	9〜18	15〜21
	Plugin Development Environment (PDE)		6〜12	6〜12	9〜18	15〜21
	Business Intelligence and Reporting Tools (BIRT)		6〜15	3〜12	6〜21	12〜21
	Eclipse Modeling Framework (EMF)		6〜18	3〜12	6〜18	12〜0
	C/C++ Development Tooling (CDT)		6〜18	6〜12	9〜18	15〜21
	Eclipse Web Tools Platform Project (WTP)		6〜18	6〜12	9〜18	12〜21
	Eclipse Platform	Platform	9〜18	3〜12	6〜18	15〜0
		SWT	9〜18	3〜12	9〜18	15〜0
		RCP	6〜15	3〜12	6〜18	12〜21

7. 議論

本章では仮説を検証するために行った分析結果を基に考察と制約について議論を行う．7.1 節では 3 つに分類した OSS プロジェクトに関してそれぞれ考察を行い，7.2 節では本論文の制約について述べる．

7.1 考察

本節では 3 つに分類した OSS プロジェクト毎に考察を行う．7.1.1 項では OSS 開発プロジェクトに関する考察を行い，7.1.2 項では企業と混合の OSS 開発プロジェクトに関する考察を行う．次に 7.1.3 項ではモジュール毎に分かれた OSS 開発プロジェクトに関して考察を行う．

7.1.1 OSS 開発プロジェクト

仮説 2 より，時差のない地域間（米→米）と時差のある地域間（米→欧）の返信時間の差には有意差があったが，返信時間の差は 1 時間程度であった．米大陸地域と欧・アフリカ地域に 6 時間以上の時差があることに対して，返信時間の差は短い．この要因の一つとして，図 3 より欧・アフリカ地域の開発者が米大陸地域の開発者の開発に従事する時間帯に合わせて情報交換を行っている可能性が高いと考えられる．そのため，米大陸地域の開発者と欧・アフリカ地域の開発者の両者が情報交換を活発に行う時間帯が同じになり，タイムラグの少ない情報交換が可能になったと考えられる．

表 5 より，PostgreSQL と Apache が Python より返信時間の中央値が短い要因として，PostgreSQL と Apache はデータベース管理システムと Web サーバといった信頼性の高いシステムが求められ，重大な不具合や脆弱性などの問題が起こると早急な対応が求められるというように，ソフトウェアの特徴も情報交換のタイムラグに影響していると考えられる．

仮説 3 より，OSS 開発プロジェクトでは返信地域の深夜から早朝にかけた時間帯にメッセージを送信すると返信が遅くなるため，この時間帯を避けてメッセー

ジを送信すれば，情報交換のタイムラグが解消される．また，米大陸地域の開発者は 10 時から 12 時頃，欧・アフリカ地域の開発者は 18 時頃にメッセージを送信すると，どの地域からも素早い返信が期待できるため，情報交換の所要時間を短縮化することが可能と考える．

7.1.2 企業と混合の OSS 開発プロジェクト

仮説 1 より，米大陸地域では情報交換が活発に行われている時間帯が OSS 開発プロジェクトに比べて短く，UTC+0 の 23 時から 13 時頃（米大陸地域の 19 時から 9 時頃）にかけてメッセージ数が極めて少なくなった要因の一つとして，企業の開発者が多く存在していることが考えられる．Eclipse は IBM で開発が開始されたプロジェクトであることから，米大陸地域の開発者には企業の開発者が多く存在する可能性がある．

本研究では，企業の開発者がどのくらい存在しているかを調査するために，メールアドレスの “@” 以降のドメイン情報[2]から企業の開発者であるかを調査した．その結果を表 7 に示す．表 7 は各地域において最も多かったドメインが企業であるかどうかをまとめたものであり，括弧の中の数字は各地域のメッセージ数のうち，トップドメインが占める割合を表す．なお，企業以外のドメインは “その他” とする．表 7 より，米大陸地域には企業の開発者が多く存在することを確認できた．

一方，欧・アフリカ地域では米大陸地域のような特徴はみられなかったため，企業の開発者は少ないと考えられる．表 7 からも，欧・アフリカ地域には企業の開発者が少ないことを確認できた．

仮説 2 より，PDE の米→米と米→欧の返信時間の差に有意差がみられなかった要因の一つとして，付録 C の表 9 より他のプロジェクトに比べてデータ数が少なかったことが考えられる．PDE に関する検定結果は偶然出た可能性が高く，この結果の信頼性は低い可能性がある．また，EMF を除いた返信時間の中央値が OSS 開発プロジェクトと比べて長くなった要因としては，図 3，図 4 より米大陸地域の開発者が情報交換を活発に行っている時間帯が短く，米大陸地域の開発者と欧・アフリカ地域の開発者が情報交換を行っている時間帯が異なっているため，

[2]is.naist.jp や gmail.com などのドメイン

表 7 各地域のトップドメイン（企業と混合の OSS 開発プロジェクト）

分析対象プロジェクト	米大陸地域	欧・アフリカ地域
Java development tools (JDT)	企業 (22.4%)	企業 (23.6%)
Plugin Development Environment (PDE)	企業 (28.0%)	その他 (15.5%)
Business Intelligence and Reporting Tools (BIRT)	企業 (67.8%)	その他 (13.2%)
Eclipse Modeling Framework (EMF)	企業 (73.1%)	その他 (13.8%)
C/C++ Development Tooling (CDT)	企業 (35.8%)	企業 (15.3%)
Eclipse Web Tools Platform Project (WTP)	企業 (31.2%)	その他 (11.4%)

情報交換にタイムラグが発生する可能性が高くなったと考えられる．EMF では米大陸地域の開発者と欧・アフリカ地域の開発者の両者が互いに情報交換を行う時間帯を合わせている可能性が高く，タイムラグの少ない情報交換が可能となったと考えられる．

仮説 3 より，企業と混合の OSS 開発プロジェクトでは返信地域の夕方から早朝にかけた時間帯にメッセージを送信すると返信が遅くなるため，この時間帯を避けてメッセージを送信すれば，情報交換のタイムラグが解消される．また，米大陸地域の開発者は 9 時から 12 時頃，欧・アフリカ地域の開発者は 15 時から 18 時頃にメッセージを送信すると，どの地域からも素早い返信が期待できるため，情報交換の所要時間を短縮化することが可能と考える．

7.1.3 モジュール毎に分かれた OSS 開発プロジェクト

仮説 1 より，企業と混合の OSS 開発プロジェクトと同様の傾向がみられたことから，モジュール毎に分かれた OSS 開発プロジェクトにおいても，米大陸地域の開発者には企業の開発者が多く存在し，欧・アフリカ地域には企業の開発者が少ないと考えられる．企業の開発者がどのくらい存在しているかを調査した結果を

表 8 各地域のトップドメイン（Eclipse Platfrom）

分析対象プロジェクト		米大陸地域	欧・アフリカ地域
Eclipse Platform	Platform	企業 (31.6%)	その他 (22.9%)
	SWT	企業 (11.8%)	企業 (31.8%)
	RCP	企業 (55.7%)	企業 (16.1%)

表 8 に示す．表 8 より，米大陸地域には企業の開発者が多く存在することを確認できた．しかし，欧・アフリカ地域においては想定していた結果とは異なり，企業の開発者が多く存在していることを確認した．

仮説 2 より，モジュール毎に分かれた OSS 開発プロジェクトにおいても，返信時間の中央値が OSS 開発プロジェクトと比べて長くなった要因は，図 4 より米大陸地域の開発者が情報交換を活発に行っている時間帯が短く，米大陸地域の開発者と欧・アフリカ地域の開発者が情報交換を行っている時間帯が異なっているため，情報交換にタイムラグが発生する可能性が高くなったと考えられる．

仮説 3 より，モジュール毎に分かれた OSS 開発プロジェクトにおいても，返信地域の夕方から早朝にかけた時間帯にメッセージを送信すると返信が遅くなるため，この時間帯を避けてメッセージを送信すれば，情報交換のタイムラグが解消される．また，米大陸地域の開発者は 9 時から 12 時頃，欧・アフリカ地域の開発者は 15 時から 18 時頃にメッセージを送信すると，どの地域からも素早い返信が期待できるため，情報交換の所要時間を短縮化することが可能と考える．

7.2 本論文の制約

OSS 開発は開発に従事する時間帯に束縛がないため，開発者個々人の都合の良い時間帯に開発を行うことが可能である．そのため，時差のない地域間の情報交換であっても，開発者個々人の開発に従事する時間帯が異なることで，開発者間の情報交換にはタイムラグが発生する可能性が高くなる．本論文では，開発者間の時差に着目して情報交換のタイムラグの実態を解明したが，開発者個々人の開発に従事する時間帯の違いまでは考慮していない．開発者個々人に着目して情報

交換のタイムラグの実態を解明することで，情報交換所要時間の更なる短縮化が可能となり得る．

本論文では，情報交換のタイムラグが解消される方法として，素早い返信が期待できる時間帯にメッセージを送信することを述べた．しかしながら，送信するメッセージの内容によってすぐに返信できない場合も考えられる．このような場合，情報交換のタイムラグを解消できる時間帯にメッセージを送ったとしても，素早い返信が行われずタイムラグが発生し，情報交換のタイムラグを解消できない可能性がある．

8. 関連研究

OSS 開発における情報交換のタイムラグや遅延に関する問題として，OSS プロジェクト内のバグトラッキングシステムを用いた不具合修正プロセスに関する研究が今までに行われている [20][22][34]．例えば，Wang ら [34] は OSS の進化を計測するための適切な指標の必要性を挙げ，OSS の進化の特徴を考慮した指標を提案している．提案指標の一つとして，ソフトウェア中に存在している不具合の数や修正された不具合の数などが挙げられている．Wang ら [34] の研究では，提案指標のケーススタディとして Linux ディストリビューションである Ubuntu を対象とした分析が行われている．その結果，報告されている不具合のうち，修正されている不具合は 20%程度に過ぎず，修正されていない不具合の中には開発者の割り当てが行われていない不具合が多く存在していることが明らかにされている．これらの調査結果は，バグトラッキングシステムに報告された全ての不具合が解決するまでの時間が長くなり，不具合の修正を開始する時間も長くなることが指摘されている．しかし，これらの研究は不具合を解決するまでの時間や情報交換のタイムラグが明らかにされていない．

Mockus ら [24] と Herraiz ら [19] は，OSS 開発において不具合を解決する平均時間を報告した．Mockus ら [24] は OSS 開発が成功する理由を調査するために，2 つの有名な大規模なプロジェクトである Apache プロジェクトと Mozilla プロジェクトの分析を行った．ケーススタディとして，ユーザにとってソフトウェアの不具合は迅速に解決されることが求められるため，不具合が解決されるまでの時間を分析した．分析の結果，カーネルやプロトコルに関するモジュールや広い範囲で利用される機能を持つモジュールに存在する障害は不具合が解決されるまでの時間が短いことがわかった．また，優先度別の不具合が解決されるまでの時間は，P1 と P3 の 50%の不具合が 30 日以内，P2 の 50%の不具合が 80 日以内，P4 と P5 の 50%の不具合が 100 日以内に解決されることがわかった．これらの研究は，OSS 開発において不具合修正プロセスの正確な知識に焦点が当てられているが，不具合修正プロセスと開発者間の情報交換のタイムラグが明らかにされていない．

分散開発における情報交換のタイムラグの実態把握を目的とした研究が今までに行われている [3][4][21][23][31]．例えば，Herbsleb ら [18] は企業のオフショア開

発を対象とし，開発者が各拠点に点在している開発プロジェクトは，各拠点に点在していない開発プロジェクトと比べて開発スピードに遅延が生じているかを調査している．調査の結果，開発者が点在している分散開発は，対面のコミュニケーションを欠落させ，開発スピードに遅延をもたらすことが明らかにされている．一方で，Nguyen ら [26] は OSS 開発を対象とし，従来報告されているように分散開発が地域差により情報交換のタイムラグをもたらしているかを調査している．Eclipse Jazz プロジェクトを対象とした調査の結果，開発者間の地域差は情報交換のタイムラグに影響は与えているが，それほど大きなものではないという結論を出している．これらの研究では開発者間の地域差に着目してはいるものの，開発者が開発に従事する時刻までは考慮されていない．

本論文では，メッセージの送信時刻と返信時間を細分化し，開発者間の時差と送信時刻がそれぞれ情報交換のタイムラグにどういった影響を与えているかを分析した点，および，情報交換のタイムラグを解消する時間帯を示した点が異なる．

9. おわりに

本論文では OSS プロジェクト内における情報交換のタイムラグの実態を解明し，情報交換の所要時間を短縮化するための指針を得ることを目的として，OSS 開発における情報交換においてタイムラグが発生する要因に関する仮説と，タイムラグが開発者間の情報交換に与える影響に関する仮説について分析と検証を行った．分析対象プロジェクトを OSS 開発プロジェクト，企業と混合の OSS 開発プロジェクト，モジュール毎に分かれた OSS 開発プロジェクトの 3 つに分類して分析を行った結果，得られた主な知見は以下の通りである．

- OSS 開発プロジェクトでは，返信地域が深夜から早朝となる時間帯にメッセージを送信すると，日中に比べて開発者間の情報交換にタイムラグが発生する可能性が高いことがわかった．また，時差の影響を受けずに情報交換を行うことができる時間帯は，米大陸地域では 10 時から 12 時頃，欧・アフリカ地域では 18 時頃であることがわかった．

- 企業と混合の OSS 開発プロジェクト，モジュール毎に分かれた OSS 開発プロジェクトでは，OSS 開発を仕事としている企業の開発者が多く存在するため，返信地域が夕方から早朝となる時間帯にメッセージを送信すると，日中に比べて開発者間の情報交換にタイムラグが発生する可能性が高いことがわかった．また，時差の影響を受けずに情報交換を行うことができる時間帯は，米大陸地域では 9 時から 12 時頃，欧・アフリカ地域では 15 時から 18 時頃であることがわかった．

OSS 開発において，本論文で示唆したタイムラグが解消できる時間帯に情報交換を行うと，情報交換の所要時間を短縮化することが可能であると考える．しかし，重大な不具合や脆弱性の問題が発生するなど送信するメッセージの内容によっては，早急に返信を得たい状況も考えられる．この状況においては，問題解決のために早急な対応が求められるため，素早い返答が期待できる時間帯以外であってもメッセージを送信する方が望ましい．

本論文の貢献として，ML やニュースグループにおける情報交換活動だけでなく，情報交換活動の媒体を変更するときにおいて，本論文で示した素早い返信の

期待できる時間帯が利用可能であると考える．例えば，リアルタイム性の低いMLからリアルタイム性の高いIRC媒体に移行するときは，情報交換のタイムラグをできるだけ解消するために情報交換の最適なタイミングの理解が必要であると考えられる．その際に，素早い返信の期待できる時間帯が必要となる．

今後，本研究の知見が効率的な情報交換の実現へのきっかけになることを期待する．

謝辞

本研究を遂行するにあたり多くの方々に御指導，御助言，御協力を賜りました．この場を借りてこれまでお世話になった方々に感謝の意を表したいと思います．誠にありがとうございました．

主指導教員であり本論文の審査委員を務めて頂いた，奈良先端科学技術大学院大学 情報科学研究科 松本 健一 教授に対し，厚く御礼申し上げます．論文執筆やプレゼンテーションの作法など研究に対する直接的な指導に限らず，研究に取り組む上での心構えから，研究者としての姿勢などありとあらゆる面で熱心な御指導を賜りました．私が本研究を成し遂げることができたのは，先生の御理解と温かい御助言に励まされたおかげです．先生の御尽力に敬意を表し，心より感謝致します．

副指導教官であり本論文の審査委員を務めて頂いた，奈良先端科学技術大学院大学 情報科学研究科 関 浩之 教授には，本研究を進めるに当たり，貴重な御指導を賜りました．学内発表において多数の御質問，御指摘を頂き，心より感謝致します．

副指導教官であり本論文の審査委員を務めて頂いた，奈良先端科学技術大学院大学 情報科学研究科 門田 暁人 准教授には，本論文に対して多くの建設的なアドバイスを賜りました．研究以外の様々な活動においてもお世話になり，心より感謝致します．

副指導教官であり本論文の審査委員を務めて頂いた，奈良先端科学技術大学院大学 情報科学研究科 大平 雅雄 助教には，私の研究生活の全過程において熱心な御支援を賜りました．研究の進め方から，論文執筆，プレゼンテーション技術，全てに渡り，熱意ある御指導を賜りました．私の研究は先生の御指導がなければ成し得ませんでした．大学院入学前から様々な温かい御助言を頂き，心より感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 森崎 修司 助教には，本論文において一歩離れたところから有益な御指摘，御意見を頂きました．また，ソフトウェアレビューの研究を通じて，物事を進める上で段取りの重要さや，締め切りを守ることへの心構えなど，数多くの御指導を賜りました．研究以外の様々な活

動においても，しばしばお気遣いの言葉を賜り，心より感謝致します．

奈良先端科学技術大学院大学 情報科学研究科 角田 雅照 特任助教には，本論文について一歩離れたところから有益な御指摘，御意見を頂きました．研究以外の様々な活動においても，しばしばお気遣いの言葉を賜り，心優しく接して頂きました．心より感謝致します．

奈良先端科学技術大学院大学 ソフトウェア工学講座 亀井 靖高 博士には，論文執筆に対するアドバイス，プレゼンテーションの作法等，適切かつ建設的なアドバイスを頂きました．また，研究会発表や学内発表の練習にも数多く付き合って頂きました．研究生活においても，研究以外の場においても御指導して頂きました．氏からの励ましがあったからこそ，二年間諦めずに研究を続けてくることができました．大学院入学前から様々な温かい御助言を頂き，心より感謝致します．

奈良先端科学技術大学院大学 ソフトウェア工学講座 柗本 真佑 氏には，論文執筆に対するアドバイス，プレゼンテーションの作法等，適切かつ建設的なアドバイスを頂きました．また，研究会発表や学内発表の練習にも数多く付き合って頂きました．氏の研究に対する真摯な姿勢に心打たれ，研究意欲を駆り立てられました．心より感謝致します．

奈良先端科学技術大学院大学 ソフトウェア工学講座 伊原 彰紀 氏には，論文執筆に対するアドバイス，プレゼンテーションの作法等，適切かつ建設的なアドバイスを頂きました．また，論文の体裁チェック，研究会発表や学内発表の練習に数多く付き合って頂きました．研究生活においても，研究以外の場においても昼夜問わず付きっきりで御指導して頂きました．氏からの励ましがあったからこそ，二年間諦めずに研究を続けてくることができました．心より感謝致します．

奈良先端科学技術大学院大学 ソフトウェア工学講座ならびにソフトウェア設計学講座の皆様には，日頃より多大な御協力と御助言を頂きました．研究発表の準備や研究の過程で賜った御助力，励ましの言葉により，私は研究を完遂することができました．研究のみならず日々の生活においても，皆様の明るさ，優しさにより非常に充実した二年間を過ごすことができました．深く感謝致します．

最後に，大学院において研究するにあたり，日頃より私を励まし，温かく見守ってくれた家族に心より感謝致します．

参考文献

[1] Ban Al-Ani and H. Keith Edwards. A comparative empirical study of communication in distributed and collocated development teams. In *Proceedings of the 3rd International Conferenceon Global Software Engineering (ICGSE'08)*, pp. 35–44, 2008.

[2] Apache Software Foundation. Apache HTTP Server Project. ⟨http://httpd.apache.org⟩, accessed 2010-02-04.

[3] Christian Bird, Nachiappan Nagappan, Premkumar Devanbu, Harald Gall, and rendan Murphy. Does distributed development affect software quality? an empirical case study of windows vista. In *Proceedings of the 31st International Conference on Software Engineering (ICSE'09)*, pp. 518–528, 2009.

[4] Erran Carmel. *Global Software Teams: Colloborating Across Borders and Time Zones*. Prentice Hall, Upper Saddle River, USA, Jan. 1999.

[5] Marcelo Cataldo and Sangeeth Nambiar. On the relationship between process maturity and geographic distribution: an empirical analysis of their impact on software quality. In *Proceedings of the the 7th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE'09)*, pp. 101–110, 2009.

[6] Jorge A. Colazo. Following the sun: Exploring productivity in temporally dispersed. In *Proceedings of the 14th Americas Conference on Information Systems (AMCIS'08)*, 2008.

[7] Paul A. David, Andrew Waterman, and Seema Arora. FLOSS-US: The Free/Libre/Open Source Software Survey for 2003. available from ⟨http://www.stanford.edu/group/floss-us/⟩, accessed 2010-02-04.

[8] Chris Dibona, Danese Cooper, and Mark Stone. *Open sources 2.0*. O'Reilly Media, Oct. 2005.

[9] Eclipse Foundation. Business Intelligence and Reporting Tools (BIRT). ⟨http://www.eclipse.org/birt/phoenix/⟩, accessed 2010-02-04.

[10] Eclipse Foundation. Eclipse C/C++ Development Tooling (CDT). ⟨http://www.eclipse.org/cdt/⟩, accessed 2010-02-04.

[11] Eclipse Foundation. Eclipse Modeling Framework (EMF). ⟨http://www.eclipse.org/modeling/emf/⟩, accessed 2010-02-04.

[12] Eclipse Foundation. Eclipse Platform. ⟨http://www.eclipse.org/platform/⟩, accessed 2010-02-04.

[13] Eclipse Foundation. Eclipse Project. Eclipse.org home ⟨http://www.eclipse.org/⟩, accessed 2010-02-04.

[14] Eclipse Foundation. Java development tools (JDT). ⟨http://www.eclipse.org/jdt/⟩, accessed 2010-02-04.

[15] Eclipse Foundation. Plugin Development Environment (PDE). ⟨http://www.eclipse.org/pde/⟩, accessed 2010-02-04.

[16] Eclipse Foundation. Web Tools Platform (WTP) Project. ⟨http://www.eclipse.org/webtools/⟩, accessed 2010-02-04.

[17] Karl Fogel. *Producing Open Source Software: How To Run Sucessful Free Software Project*. O'Reilly Media, Nov. 2005.

[18] James D. Herbsleb, Audris Mockus, Thomas A. Finholt, and Rebecca E. Grinter. An empirical study of global software development: Distance and speed. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE'01)*, pp. 81–90, 2001.

[19] Israel Herraiz, Daniel M. German, Jesus M. Gonzalez-Barahona, and Gregorio Robles. Towards a simplification of the bug report form in eclipse. In *Proceedings of the 5th International Working Conference on Mining Software Repositories (MSR'08)*, pp. 145–148, 2008.

[20] Akinori Ihara, Masao Ohira, and Ken ichi Matsumoto. An analysis method for improving a bug modification process in open source software development. In *Proceedings of the joint International Workshop on Principles of Software Evolution (IWPSE) and the ERCIM Workshop on Software Evolution (EVOL) (IWPSE-EVOL'09)*, pp. 135–144, 2009.

[21] Dale Walter Karolak. *Global Software Development: Managing Virtual Teams and Environments*. Wiley-IEEE Computer Society Press, Los Alamitos, USA, Jan. 1999.

[22] Sunghun Kim and Jr. E. James Whitehead. How long did it take to fix bugs? In *Proceedings of the 5th International Working Conference on Mining Software Repositories (MSR'06)*, pp. 173–174, 2006.

[23] Allen E. Milewski, Marilyn Tremaine, Richard Egan, Suling Zhang, Felix Kobler, and Patrick O'Sullivan. Guidelines for effective bridging in global software engineering. In *Proceedings of the 3rd International Conferenceon Global Software Engineering (ICGSE'08)*, pp. 23–32, 2008.

[24] Audris Mockus, Roy T. Fielding, and James D. Herbsleb. Two case studies of open source software development: Apache and mozilla. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, Vol. 11, No. 3, pp. 309–346, 2002.

[25] Sandro Morasca, Davide Taibi, and Davide Tosi. Towards certifying the testing process of open-source software: New challenges or old methodologies? In *Proceedings of the 31st ICSE Workshop on Emerging Trends in*

Free/Libre/Open Source Software Research and Development (FLOSS'09), pp. 25–30, 2009.

[26] Thanh Nguyen, Timo Wolf, and Daniela Damian. Global software development and delay: Does distance still matter? In *Proceedings of the 3rd International Conferenceon Global Software Engineering (ICGSE'08)*, pp. 45–54, 2008.

[27] PostgreSQL Global Development Group. PostgreSQL. ⟨http://www.postgresql.org/⟩, accessed 2010-02-04.

[28] Python Software Foundation. Python Programming Language. Official Website ⟨http://www.python.org/⟩, accessed 2010-02-04.

[29] Eric S. Raymond. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary.* O'Reilly Media, Oct. 1999.

[30] Gregorio Robles and Jesus M. Gonzalez-Barahona. Geographic location of developers at sourceforge. In *Proceedings of the 5th International Working Conference on Mining Software Repositories (MSR'06)*, pp. 144–150, 2006.

[31] Raghvinder Sangwan, Matthew Bass, Neel Mullick, Daniel J. Paulish J., and Juergen Kazmeier. *Global Software Development Handbook (Auerbach Series on Applied Software Engineering Series).* Auerbach Publications, Boston, USA, Sep. 2006.

[32] Emad Shihab, Nicolas Bettenburg, Bram Adams, and Ahmed E. Hassan. On the central role of mailing lists in open source projects: An exploratory study. In *Proceedings of the 3rd International Workshop on Knowledge Collaboration in Software Development (KCSD'09)*, pp. 34–48, 2009.

[33] Ran Tang, Ahmed E. Hassan, and Ying Zou. A case study on the impact of global participation on mailing lists communications of open source projects.

In *Proceedings of the 3rd International Workshop on Knowledge Collaboration in Software Development (KCSD'09)*, pp. 63–76, 2009.

[34] Yi Wang, Defeng Guo, and Huihui Shi. Measuring the evolution of open source software systems with their communities. *SIGSOFT Software Engineering Notes*, Vol. 32, No. 6, p. 7, 2007.

[35] Steven Weber. *The Success of Open Source*. Harvard University Press, Apr. 2004.

[36] 独立行政法人情報処理推進機構. オープンソース情報データベース OSSiPedia. 入手先 ⟨http://ossipedia.ipa.go.jp/⟩, 参照 2010-02-04.

付録

A. 地域別メッセージ数の分布

各タイムゾーンにおけるメッセージ数の分布を図 5，図 6 に示す．各図の横軸は各タイムゾーンを表し，縦軸はメッセージ数を表す.

B. 各時刻における返信時間の分布

地域毎の各時刻における返信時間の分布を図 7〜図 18 に示す．各図の横軸は送信地域の現地時刻を表し，縦軸は各返信時間におけるメッセージ数の割合を表す.

C. 各地域における基本統計量

各地域のデータ数を表 9 に示す.

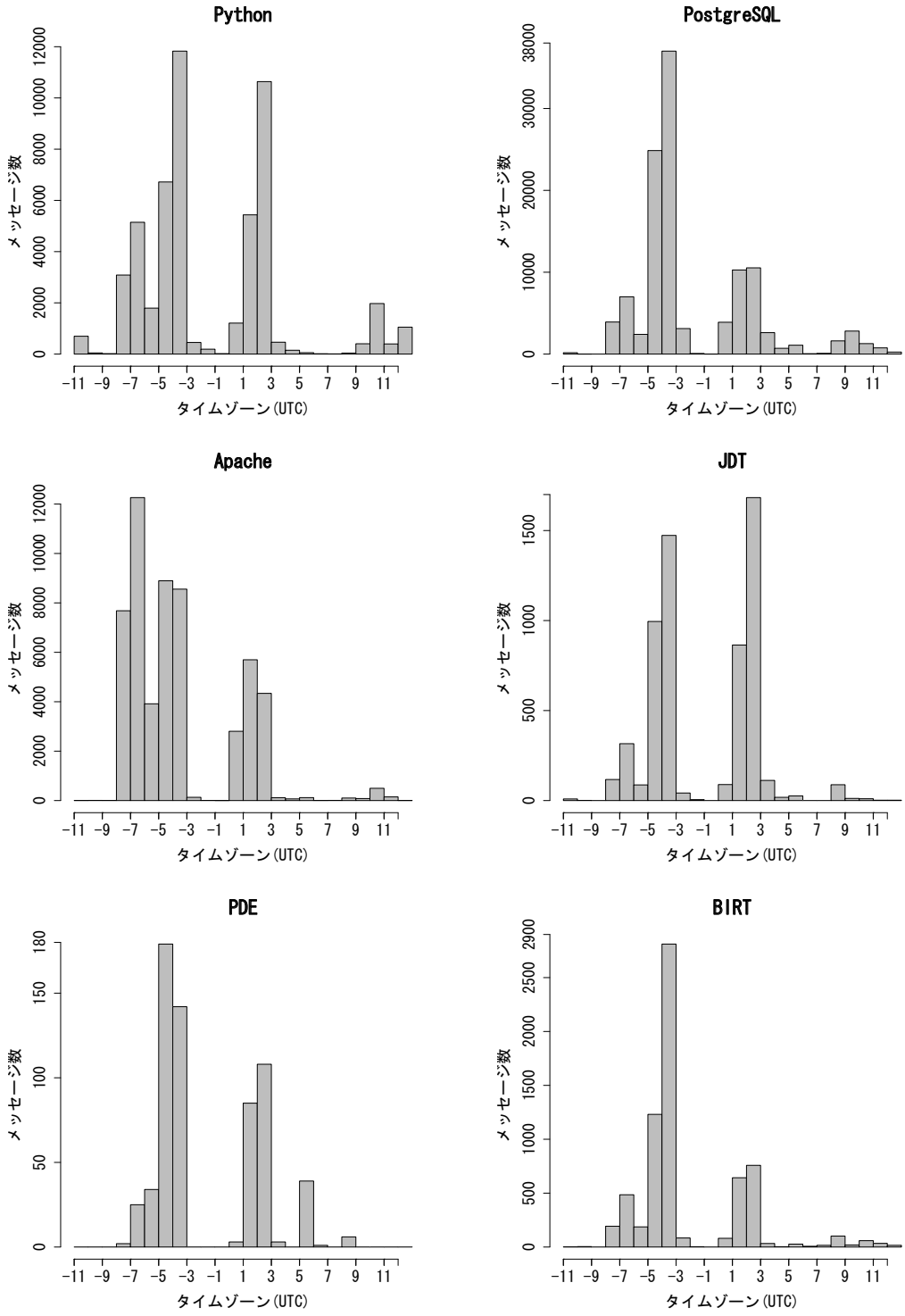

図 5 地域別メッセージ数の分布 - 1

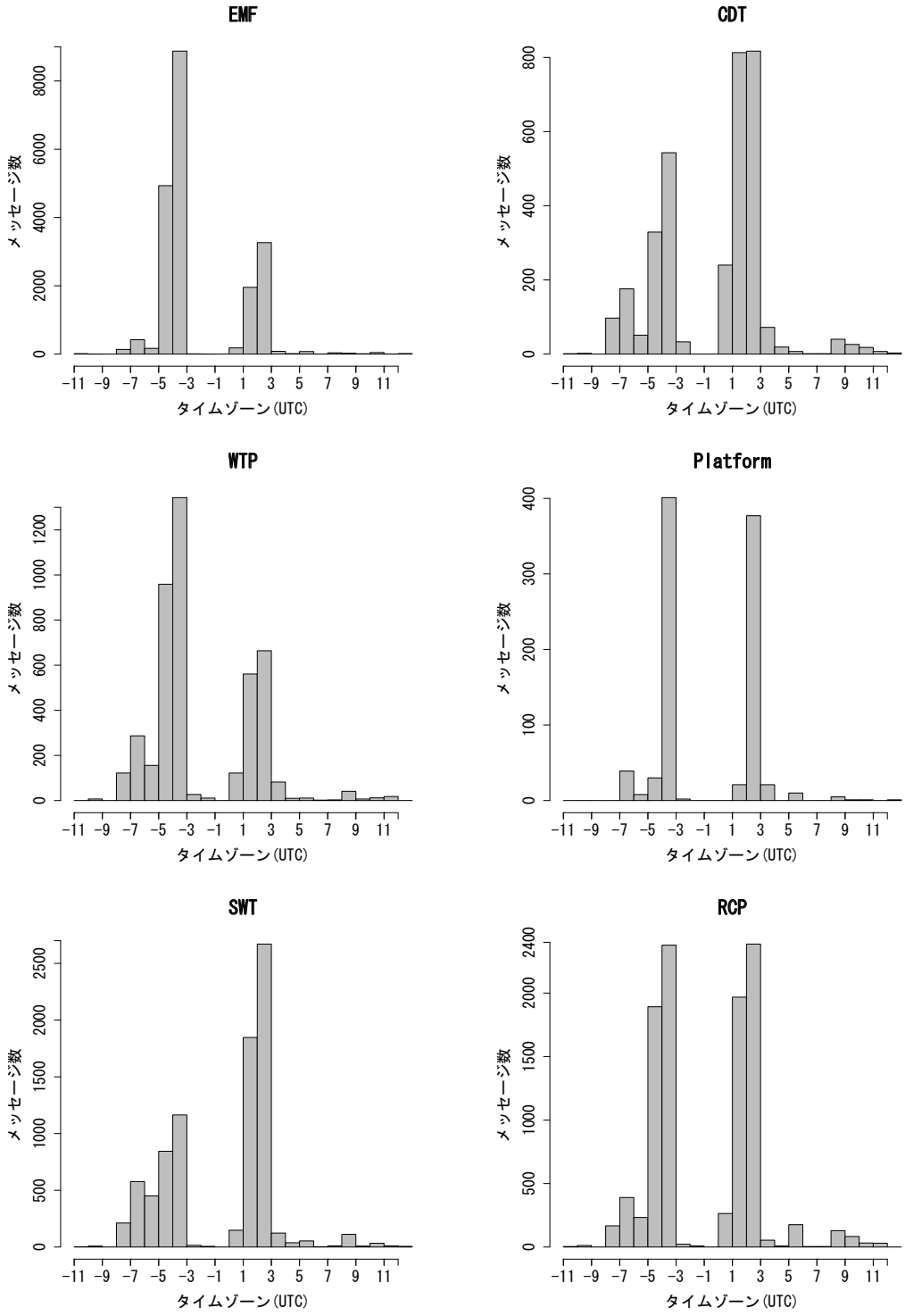

図 6 地域別メッセージ数の分布 - 2

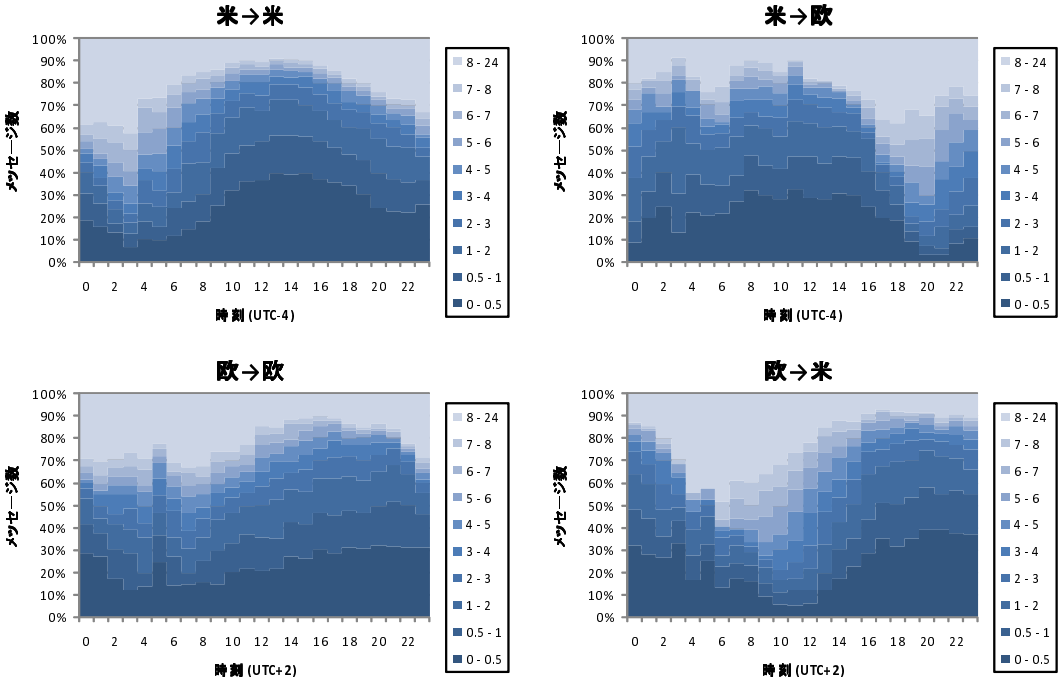

図 7 時刻別の返信時間の分布 (Python)

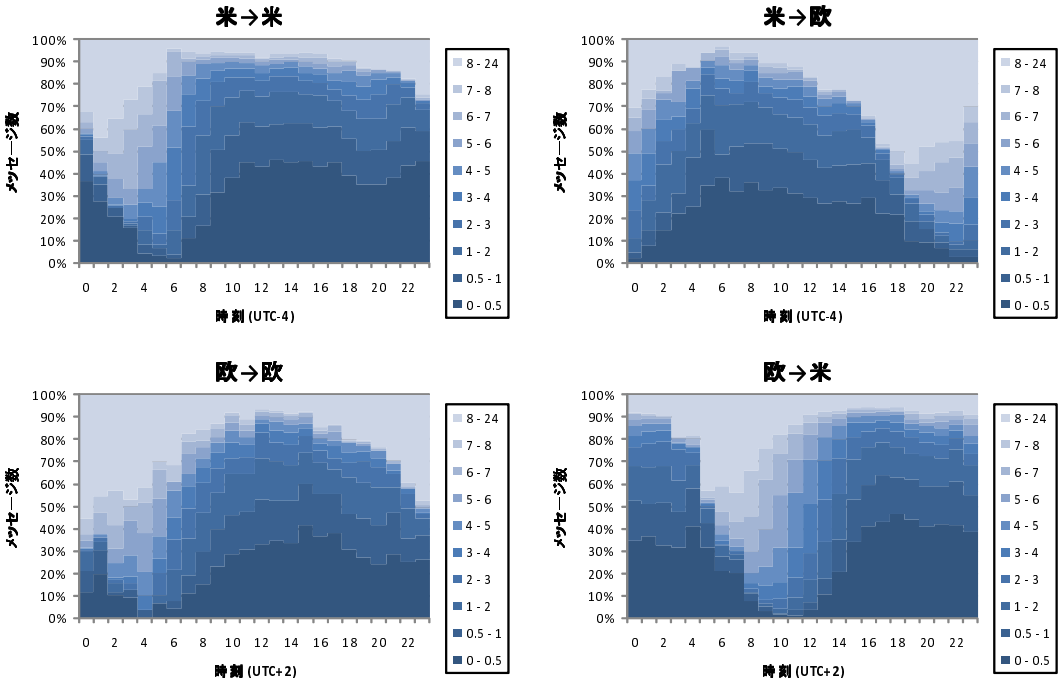

図 8 時刻別の返信時間の分布 (PostgreSQL)

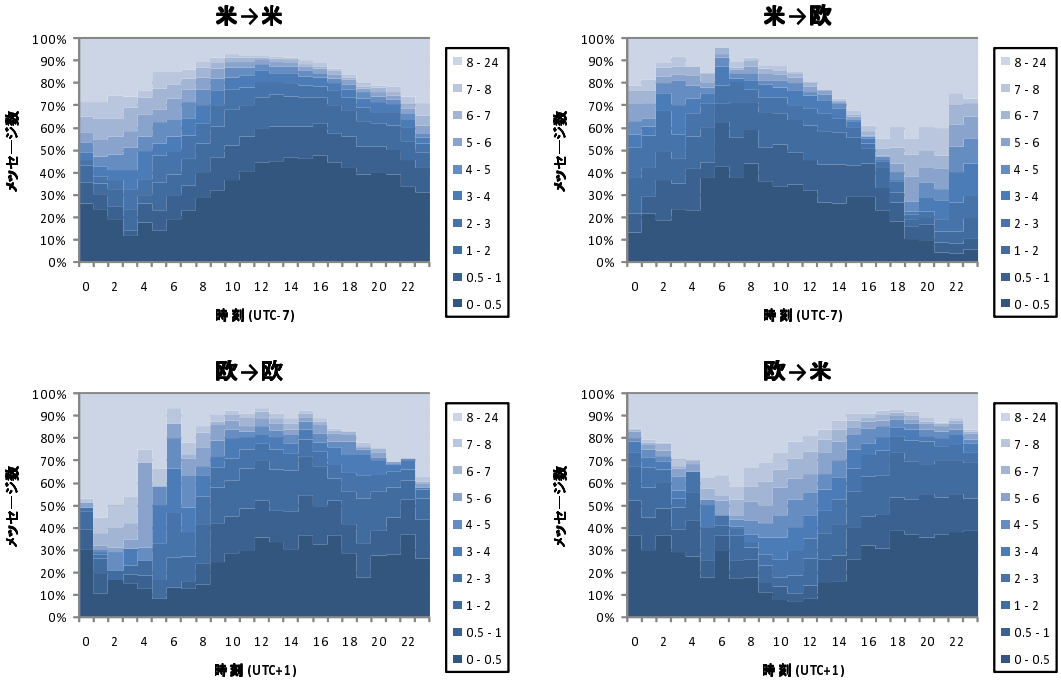

図 9 時刻別の返信時間の分布 (Apache)

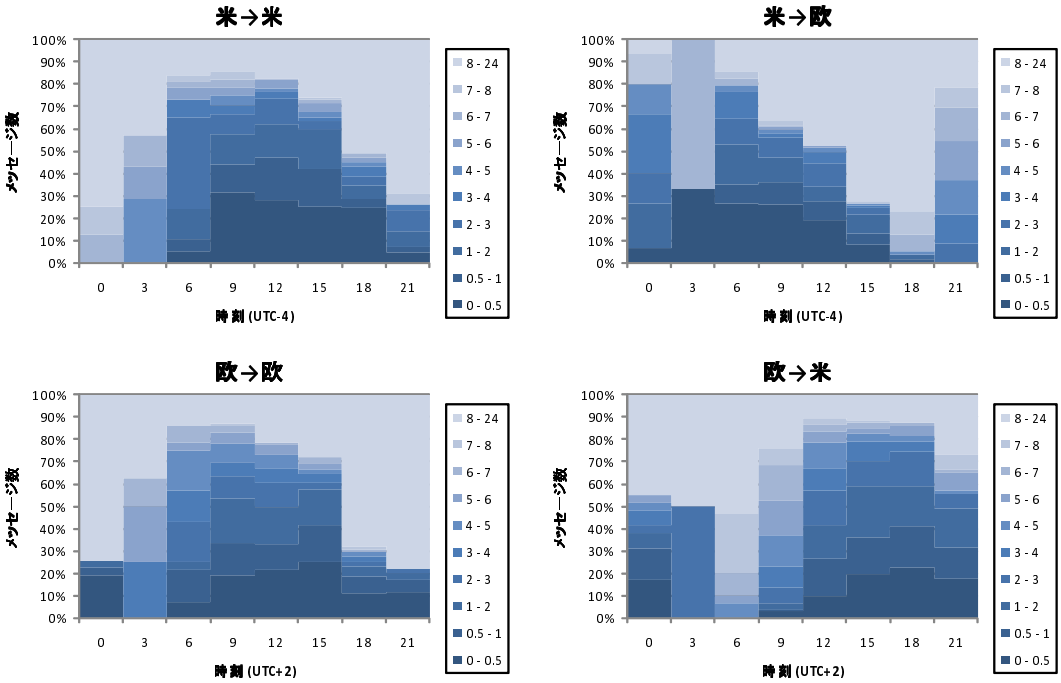

図 10 時刻別の返信時間の分布 (JDT)

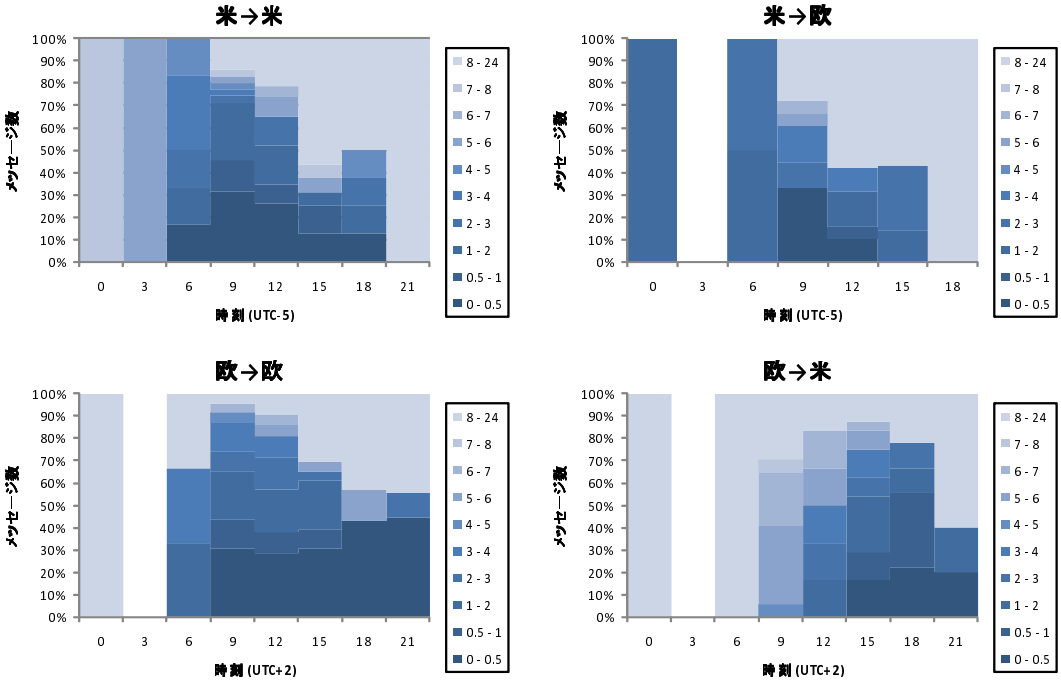

図 11 時刻別の返信時間の分布 (PDE)

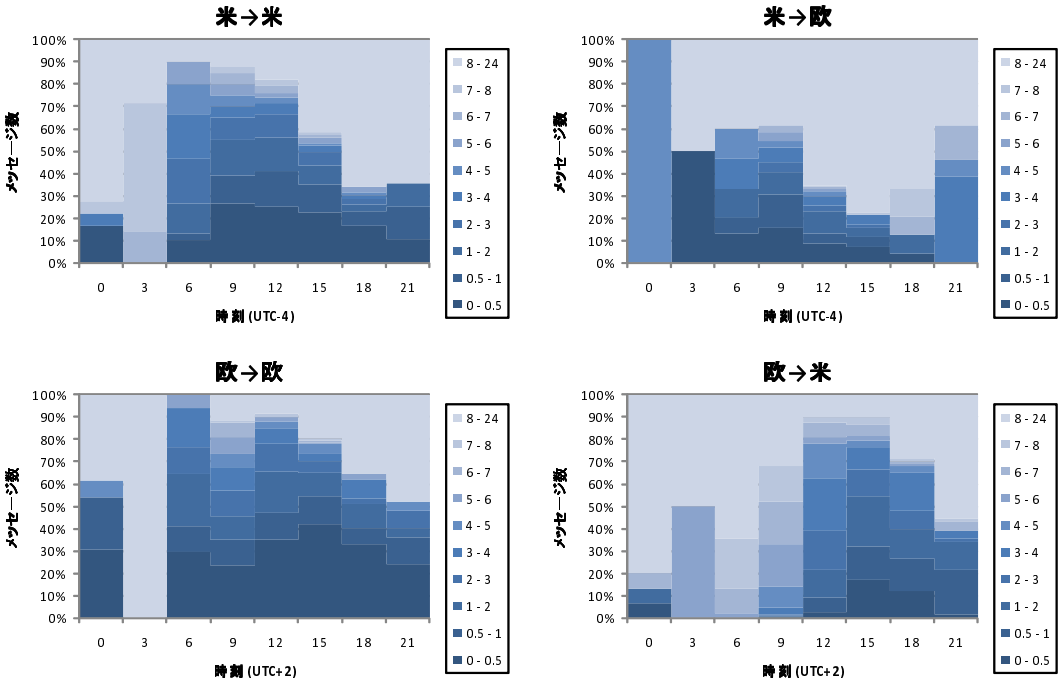

図 12 時刻別の返信時間の分布 (BIRT)

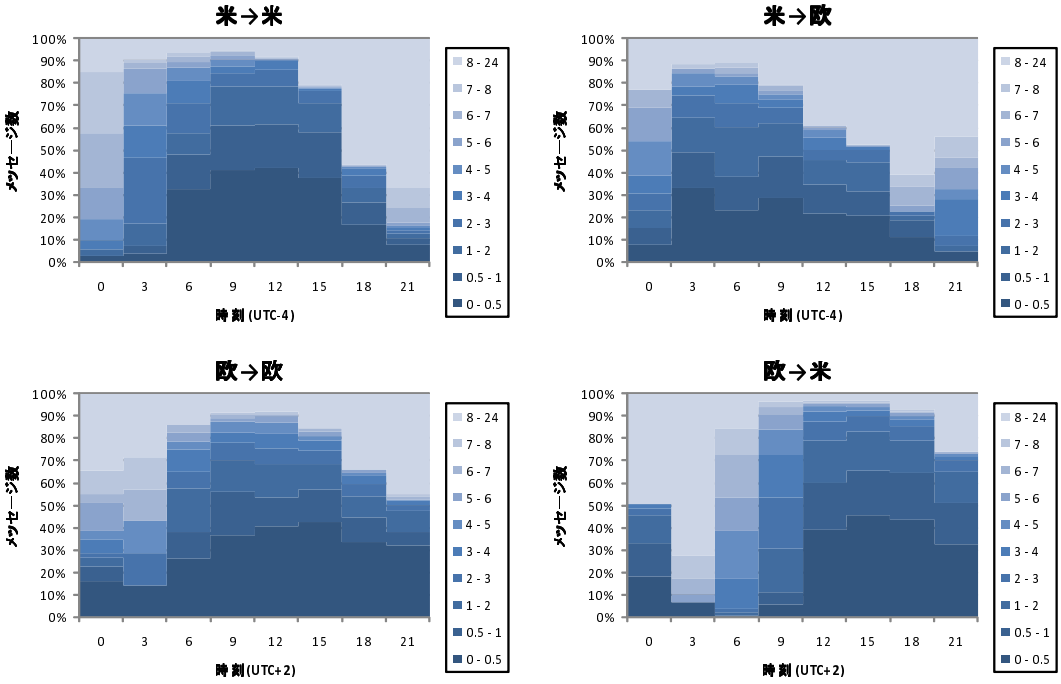

図 13 時刻別の返信時間の分布 (EMF)

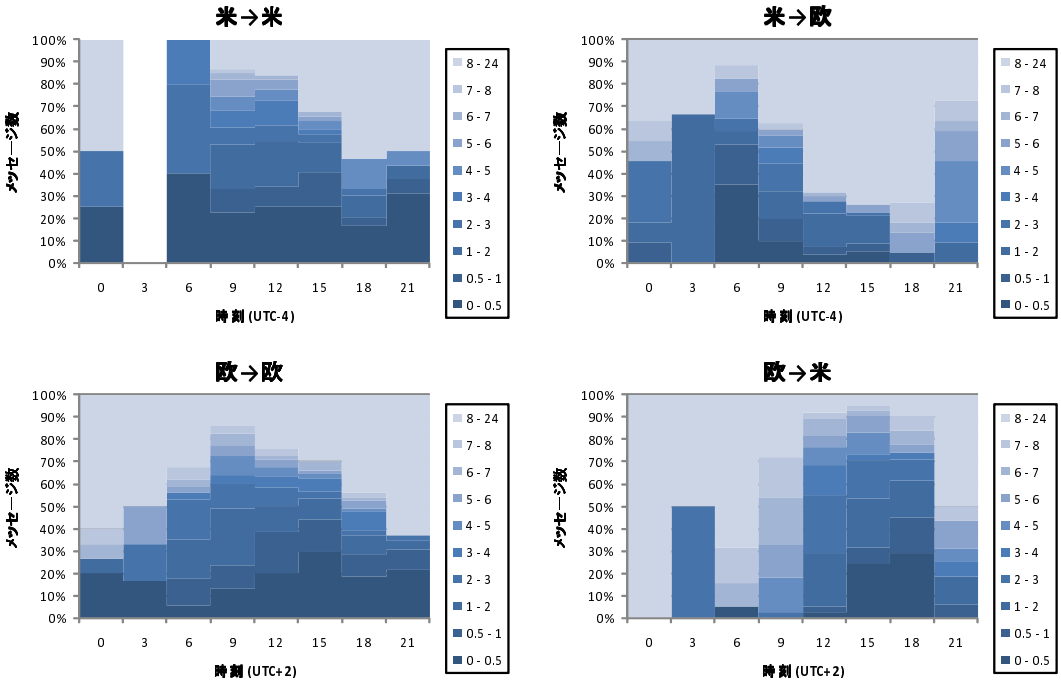

図 14 時刻別の返信時間の分布 (CDT)

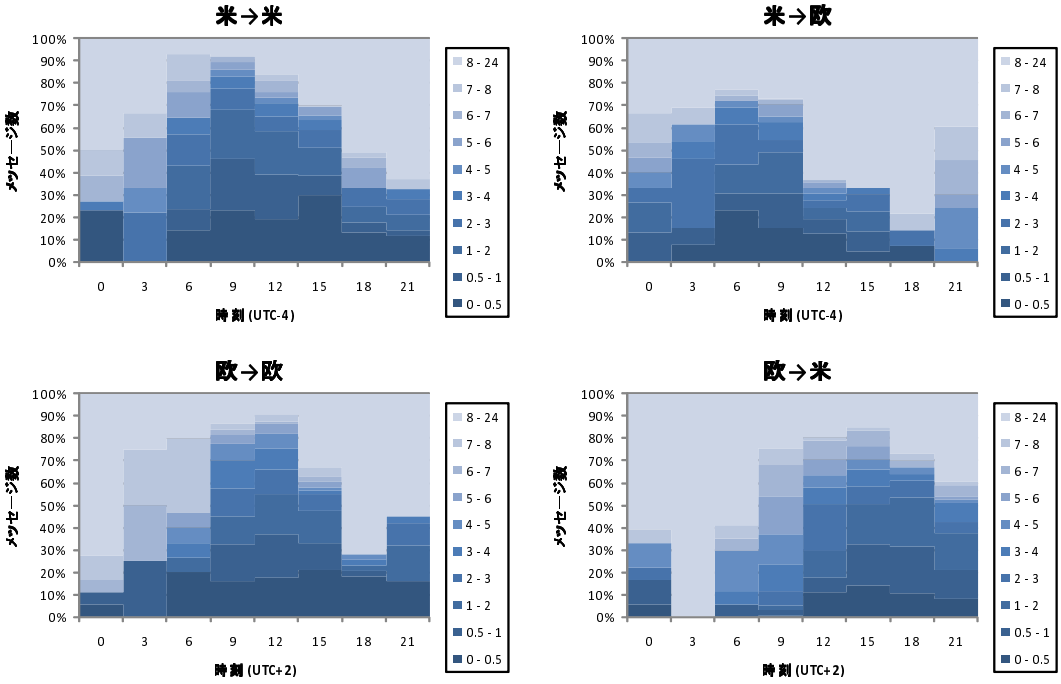

図 15 時刻別の返信時間の分布 (WTP)

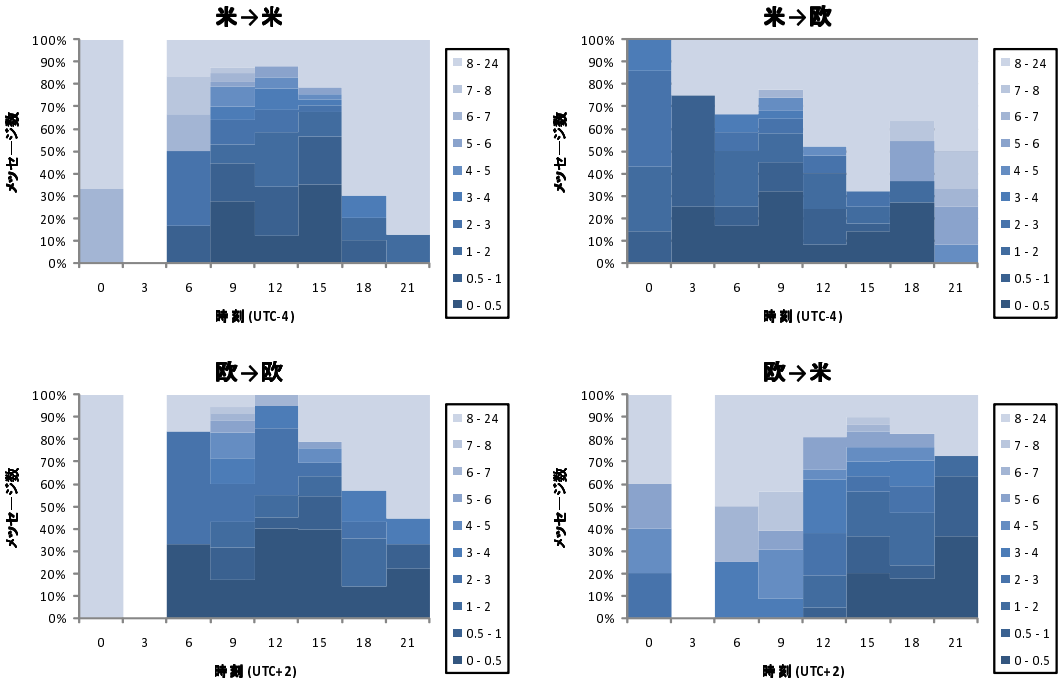

図 16 時刻別の返信時間の分布 (Platform)

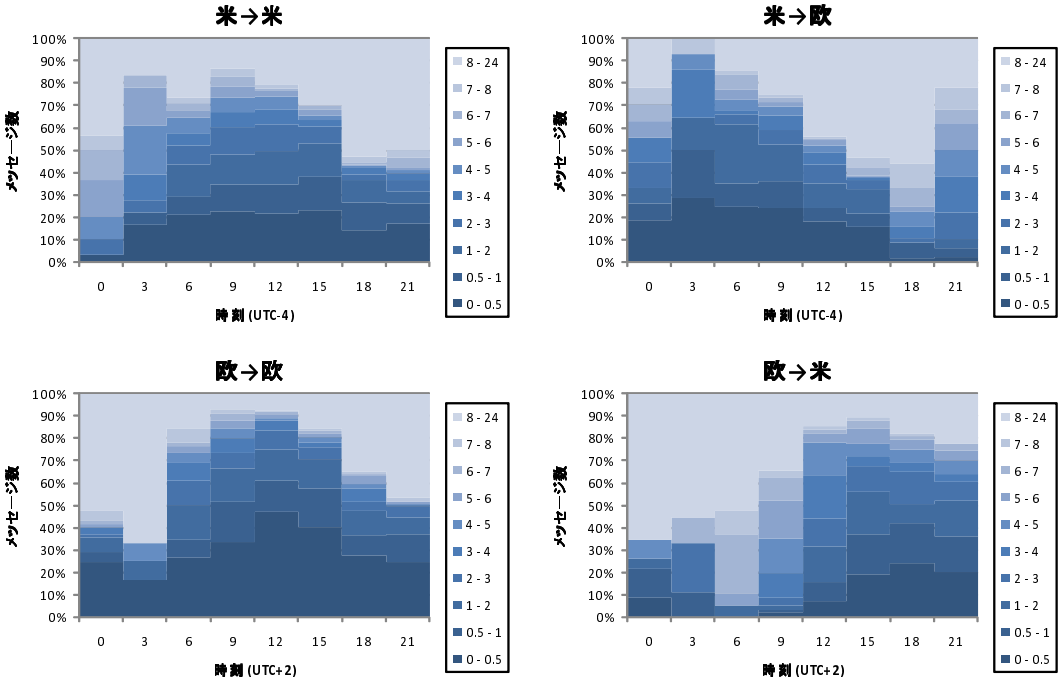

図 17 時刻別の返信時間の分布 (SWT)

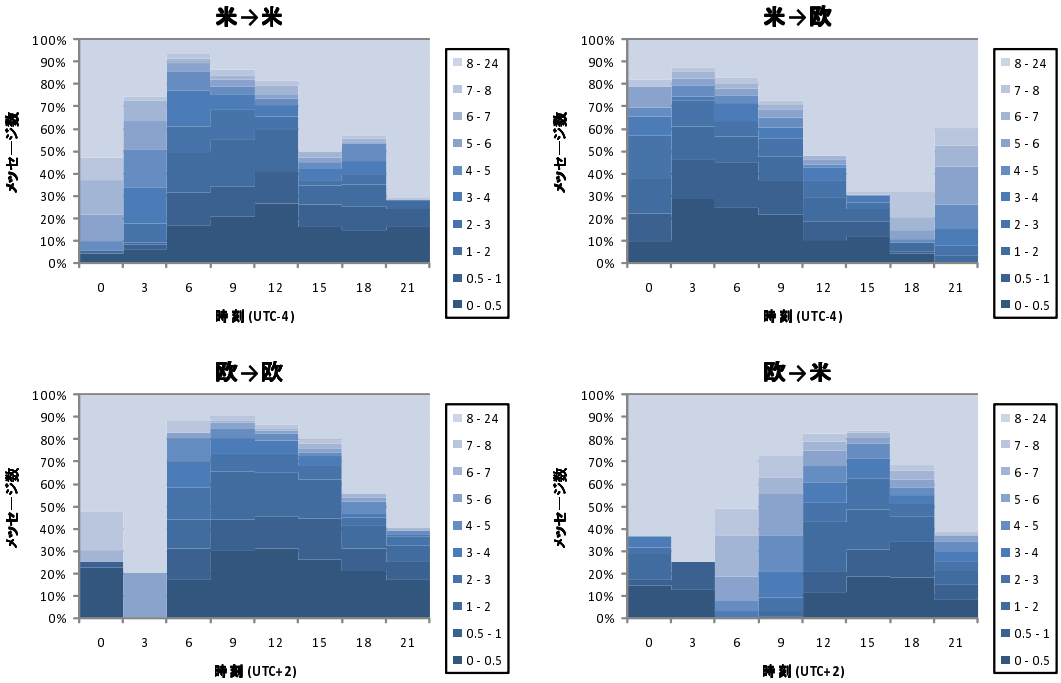

図 18 時刻別の返信時間の分布 (RCP)

表 9 地域別の基本統計量（米: 米大陸地域，欧: 欧・アフリカ地域）

分析対象プロジェクト			米→米（件）	米→欧（件）	欧→欧（件）	欧→米（件）
Python			18,901	6,942	9,426	7,215
PostgreSQL			29,016	13,790	8,249	17,109
Apache HTTP Server (Apache)			30,459	7,642	4,826	9,879
Eclipse	Java development tools (JDT)		584	510	790	580
	Plugin Development Environment (PDE)		93	49	90	72
	Business Intelligence and Reporting Tools (BIRT)		829	437	431	712
	Eclipse Modeling Framework (EMF)		3,236	2,216	1,932	4,642
	C/C++ Development Tooling (CDT)		217	258	543	192
	Eclipse Web Tools Platform Project (WTP)		543	371	336	519
	Eclipse Platform	Platform	152	130	119	111
		SWT	1,328	1,101	1,687	702
		RCP	1,230	1,224	1,606	1,181