社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－SE－151（3）
2006／3／23

Weblogを用いたソフトウェア共同開発プラットフォームの開発

牧　　俊男　　深海　　悟
大阪工業大学大学院　情報科学研究科
〒573-0196　大阪府枚方市北山 1-79-1
e-mail：fukami@is.oit.ac.jp

概要

開発者が分散した開発プロジェクトでは、開発者が分散しているために、開発中のドキュメント、ソースコードなどの成果物が分散し、これらのリソースに対して他の開発者が関与することが難しくなるという問題がある。これらを管理するための仕組みとして、Wiki などのコラボレーションツールや CVS などの構成管理ツールが提案されているが、Wiki は閲覧者がページを自由に編集できるため、誰が変更したかなどの管理を行うことが出来ず、仕様の変更などを全体に伝達する手段として使用するには不都合が生じる場合もある。また、Wiki や CVS は開発中に平行して使われるツールであるが、これらは相互に連携する手段を持たないため、開発者はドキュメントとソースを別々に管理する必要がある。本稿では、この問題を解決する手段として、他の支援ツールとの連携機構を備えた Weblog をコラボレーション手段として分散開発を支援させる方法を提案する。

Development of a software development platform using Weblog

Toshio Maki　　Satoru Fukami
Graduate School of Computer Science, Osaka Institute of Technology
1-79-1, Kitayama, Hirakata-City, Osaka, 573-0196, Japan

Abstract　In distributed software development project, specification documents and source codes are often kept and maintained under the developers in a different place. These situations will disturb the collaboration between the developers. Recently, CVS and Wiki are often used to improve such situations. But in Wiki, because everyone can freely edit the page, wrong edit by someone might cause serious troubles. Moreover, because Wiki and CVS are operated independently, developers must maintain documents and codes separately. To solve these problems, we propose a Weblog based integrated collaboration infrastructure to support distributed software development project.

1．はじめに

近年、インターネットの発達によりインターネットを通じて分散開発するプロジェクトが増加している。分散開発は、人員や開発現場の確保を容易に行うことができるメリットがある。その反面、開発者、ドキュメント、ソースコードなどのリソースが分散するため、集中管理する必要がある。これらを支援するツールとして、CVS、メーリングリスト・Wiki といったものがある。CVS は成果物のバージョン管理に、メーリングリスト・Wiki は開発者間のコミュニケーションに利用される。開発者はメーリングリストに投函された、または Wiki に記述された要求仕様の変更やバグ報告などの情報を元にソースコードに変更を加え、CVS を用いてコードの登録を行う。

しかし、メーリングリストで配信される情報から、要求仕様のどの部分が変更されたか等を知るためには、過去に配信されたメールと読み比べなければならず、書く側にも読む側にも注意が必要である。この点でいくつかの Wiki クローンは、コンテンツの変更履歴管理を行い、変更箇所を把握する機能を用意して、この問題を解決しているものがあるが、自分の担当以外のコンテンツも自由に書き換えることが出来、誤った開発情報が配信される可能性もある。また、メーリングリスト、Wiki のいずれも開発者各自の担当箇所を明確にする機構が備わっておらず、あらかじめ記述する際の約束事をシステム外で取り決めておく必要がある。メーリングリストや Wiki は主に議論や検討の場として利用されるが、これに加え、上記の問題を解決したシステムを用いれば、より分散開発を効率的に進めることができる。

本研究では、上記問題を解決するために開発者各自が Weblog を持ち、それぞれが担当する箇所のドキュメントを作成する手法を提案し、システムの試作を行った。

本稿では、まず提案手法について 2 章で述べる。次に 3 章で実際に作成したシステムについて述べる。4 章では提案手法と既存のメーリングリスト・Wiki と比較した結果について述べ、5 章でまとめと今後の課題について述べる。

2．提案手法

本研究で提案するシステムの概要を図 1 に示す。開発者は Weblog サーバ内に自分の専用スペースを持ち、このスペースに開発仕様、バグ報告、開発の進行状況を記録していく。Weblog にはコンテンツを閲覧した人間が、書き手に対して意見を述べるためのコメント機能やトラックバックと呼ばれる機構が備わっており、この機能を用いることで、開発者が記述した開発ドキュメントへのレビューが出来る。開発者は自分の担当する部分をドキュメントとして記述していくことで、自分の責任範囲を明確にすることができる。

また Weblog サーバには RSS と呼ばれるコンテンツのメタデータを記述した XML があり、開発者は RSS リーダによって、RSS を読み込むことにより、他の開発者の更新を知ることが出来る。

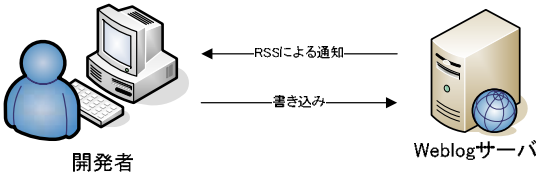

図 1　システムの構成

2.1　要求獲得支援モデル

要求獲得支援技術については、要求の表現形式や記法といった「いかに要求を正確に記述するか」に重点を置いて検討されてきた。これは「顧客が初めから明確な要求を持っており、開発者はその要求を仕様化すればよい」という前提がある上でのアプローチである。しかし、実際には顧客自身も自分の要求を完全に把握しているわけではなく、

むしろ要求は開発者やユーザ、顧客らのシステム開発の関係者が議論を繰り返し、共同して作業し、その中で練り上げられていくものである。

このような要求獲得プロセスモデルの例として要求獲得プロセスモデル（Inquiry Cycle[1]）がある。要求獲得プロセスモデルは図 2 に示した協調作業プロセスの中で、要求定義を対象とした問題評価や問題解決のためのプロセスモデルと考えることが出来る。

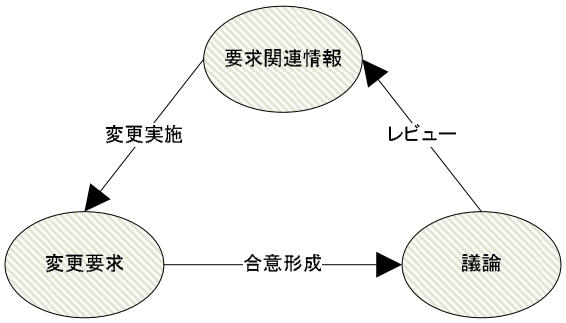

図 2　Inquiry Cycle

Inquiry Cycle では要求獲得作業を、要求仕様に関する議論、およびその結果から生じる要求仕様の変更の繰り返しとしてモデル化している。要求関連情報は、要求仕様自身、およびその背景となる情報（例えば、ドメイン知識など）からなる。議論は、質問、質問に対する回答、回答を裏付ける理由からなる。また、要求関連情報の変更には、5 つの型（明確化、洗練化、削除、統合、分割）がある。

本研究では Weblog 上で、要求獲得時に発生・参照する様々な情報（要求関連情報・要求仕様に関する議論および変更）を記録・管理・利用することで Inquiry Cycle のモデルを満たすことが出来ると考えた。すなわち、この方式ではある要求が導出されるに至った背後には、どのような議論があり、それを受けてどのような変更を行ったか等の情報を記録管理し、直感的に分かりやすい形式で他の開発者に提示することが出来る。

2.2 Weblog に対する拡張

本稿では、Weblog を開発プロジェクトで用いることで、Wiki やメーリングリストを使用するよりも効率の良い開発者の支援を行うことが出来る、と述べてきた。しかし、既存の Weblog システムを用い、開発プロジェクトの現場で使用していくには、開発中に発生する様々な要求に対応できず、十分な支援を行うことが出来ない。そこで、本研究では開発プロジェクトで使用されることを前提にし、開発中に生じる要求に対応したいくつかの機能を加えた Weblog システムを提案する。

本研究では、以下のような機能を拡張した。これらの機能が必要とされる理由と共に述べる。

- コンテンツの履歴管理・差分比較

本システムでは開発者は Weblog を用いて、仕様を記述していく。しかし、開発段階において仕様を変更しなければならない場合が時々発生する。このような場合にコンテンツを編集を行う。しかしコンテンツの変更が複数回繰り返されると、議論に途中から参加する開発者はどのような議論の経緯があって現在のコンテンツになっているのかを知るのが難しい。そこで、開発者がコンテンツを変更する際に、コンテンツの履歴を管理し、開発者が必要に応じて、これらの比較を行い、差分を可視化する機能を用意した。メーリングリストの場では 1 つの議題に対して大量のメールが投稿されることもあり、他の開発者が議論を追うことが困難になるが、本機能を用いることで、この問題を解決できる。

- コンテンツの分割・併合

コンテンツの分割・併合は Inquiry Cycle の要求関連情報の変更の型でも統合、分割として定義されている。議論が進み、コンテンツが肥大化してきた際に分離したほうがよい場合や、他のコンテンツと統合したほうがよい場合に対して使用する。

- コンテンツの関連付け・階層付け

開発用途に作成された仕様書やバグ報告書は、それ自体では完結せず、別のコンテンツを参照しなければならないことが多い。参照するコンテンツは自分の記述した場合もあれば、他の開発者が書いたコンテンツの可能性もある。しかしこれらのトピックが相互に関連のあるコンテンツだったとしても、文書内で互いのコンテンツへのハイパーリンクを用意しなければ、他の開発者が関連を把握することが出来ない。この問題を解決するために、本システムではトラックバック Ping を拡張したプロトコルでコンテンツを関連付ける方法を実装している。通常トラックバックとは受信した側から送信したページへのハイパーリンクを作成する機能であるが、拡張されたトラックバック Ping を用いると、双方向にハイパーリンクを作成するため相互に参照を行うことが出来る。

- コンテンツの委譲

メーリングリストや Wiki は開発者の責任範囲を明確にする機構が備わっていないが、提案する手法で Weblog を利用すれば、開発者の責任範囲を明確にすることが出来る。しかし、開発プロジェクトにおいては、開発者が突然脱退しなければならない場合など、責任を他の開発者に委譲しなければならないことがある。そこで、責任の委譲をシステム上で実現するためにコンテンツの委譲という機能を用意した。コンテンツの委譲を行うと、コンテンツの履歴などを含めて全て委譲先の開発者に渡される。また本機能とコンテンツの分割機能を組み合わせることで、1 つの仕様をいくつかに分割し、作業を分担するケース等にも対応することが出来る。

3． 適用例

図 3 に示すように Weblog への記録と議論機能、および差分比較機能を使用し、共同開発プロジェクトの支援を行った結果を述べる。開発クライアントには RSS リーダを導入し、30 分ごとに Weblog から RSS を取得するようにした。以降では、使用結果について述べる。

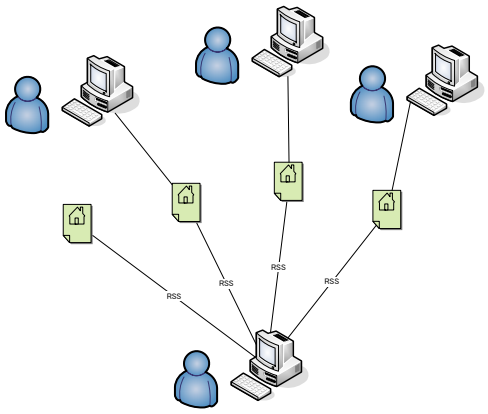

図 3　開発チームの構成

3.1 RSS 配信機能の利用

RSS はコンテンツが更新されるタイミング、コメントやトラックバックが送信され、議論が行われるタイミングで更新される。RSS リーダを導入したことで、コンテンツの変更を素早く知ることが出来た。

3.2 議論機能の利用

図 4 は、コンテンツに対して、コメントが投稿され議論が進んでいる状態を示している。

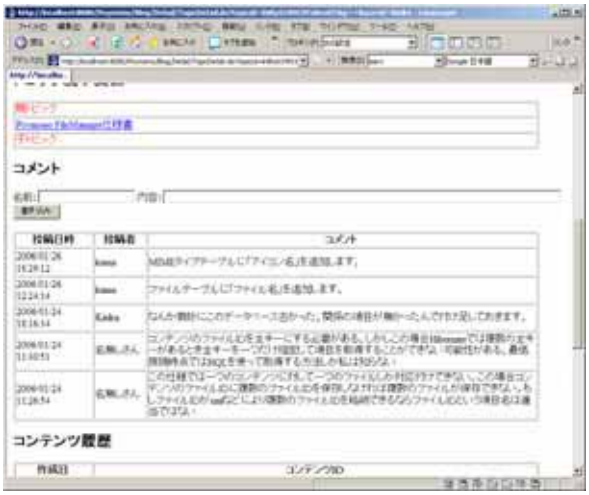

図 4　議論機能の使用結果

図 5　差分比較機能の使用結果

3.3　差分比較機能の利用

図 5 は差分比較機能の使用結果である。図中のコンテンツは、データベースのテーブルの仕様を決定するための議論が行われており、コンテンツが作成された初期の段階の仕様と、その後 6 回変更が行われた後の仕様を比較表示している。左側には変更箇所の一覧が示されており、本文中の追加・変更・削除箇所へのアンカーが用意されている。該当する本文の追加箇所は青色、変更箇所は黄色、削除箇所は赤色で表示され、開発者に対して、どの部分が変更されたかを分かりやすく可視化している。

4．既存システムとの比較

本研究で作成したシステムを、他の支援ツールと比較した結果を表 1 に示す。今回比較に用いた支援ツールは現在オープンソースプロジェクトで広く利用されているメーリングリスト（ML）と Wiki である。以降では各項目について説明を行う。

表 1　既存システムとの比較

	ML	Wiki	提案システム
開発者全員が参加すべき議論	○	○	△
1 対 1 での議論	○	△	○
変更点の比較	×	○	○
ドキュメント間の関連付け	×	△	○
図・表の表示	△	○	○
責任範囲の明確さ	×	×	○
グループ範囲の明確さ	○	○	×
検索機能	○	○	△

- 開発者全員が参加すべき議論

メーリングリストは投稿したメールが全員に配信されるため、その議論に参加したい開発者がメールを返信することで、議論を行うことが出来る。同様に Wiki も全員が目を通すべきページを作成しておけば、全員が目を通すため開発者全員を議論に参加させることが出来る。しかし、本システムでは、このような共有スペースが存在しないため、誰かの Weblog のスペースを利用するように取り決めを行わなければならない。

- 1 対 1 での議論

メーリングリストでは特定の議題に対して返信を行うことで、議論に参加することができる。Weblog でも 1 コンテンツに対してコメントを行うことで、同様のことが行える。しかし、Wiki では 1 つのページを複数人で管理するため、議論が収束する前にページが書き換えられてしまうという事態も発生する可能性があり、チーム内の秩序を保ちながら議論を進めなければならない。

- 変更点の比較

メーリングリストでは議論が進んだときに、当初の状態に比べ、どれだけ変更があったかを知るのが難しい。これに対して、本システムや一部の Wiki クローンでは差分比較機能を有しており、これを用いて、初期の状態からどれだけ変更があったかを簡単に知ることが出来る。

- ドキュメント間の関連付け

メーリングリストでは、メールを返信する形、もしくはメーリングリスト内で振られたメール ID を用いてしか、メールを参照させる手段を持たないが、本システムではトラックバックを用いて、ドキュメント間の関連付けの機能を提供している。Wiki では、誰でも自由にページの編集が出来るため、双方のページにハイパーリンクを用意することで、関連付けを表現することが出来るが、本システムを利用する場合に比べ、作業が煩雑である。

- 図・表の表示

メーリングリストはテキストが主体のコミュニティであるため、図・表などを用いて説明することが難しい。Wiki や提案システムではファイルを管理する機構を持ち、それらをドキュメント内で使用することが出来る。

- 責任範囲の明確さ

メーリングリストや Wiki では現在言及している部分の仕様が誰の担当であるかを明記しなければ分からない。しかし本システムでは仕様を管理している人間が責任者と考えることで、責任範囲を明確にすることが出来る。

- グループ範囲の明確さ

メーリングリストや Wiki は、プロジェクト単位でそれらが用意されることが多いので、メーリングリストで投稿される内容や、Wiki に書かれる情報は、そのグループ向けの情報であると判断することが出来る。しかし本システムでは分散した Weblog にその情報が書かれるため、自分のプロジェクトに関連する情報だけを抽出することが難しい。

- 検索機能

メーリングリストや Wiki ではデータベースが単純な構造になっているため、有効な検索システムがいくつか提案されているが、本システムでは分散した Weblog から情報を取得しないといけないため、有効な検索機能がまだ整備されていない。

5．まとめ

本稿では Weblog を開発プロジェクトで用いることで、従来の CVS や Wiki を用いた開発手法に比べ、効率よく開発者を支援するために必要な機能の提案を行った。

しかし、研究進めていくうちに開発プロジェクトに適用させる上で、以下のような機能が必要であると分かってきた。

● 開発グループの支援機能

本システムを用いることによって開発者間でドキュメントの公開やレビューを行うことが出来るようになる。しかし Weblog 上では開発者同士を繋ぐ方法はコンテンツの関連付け機能を用いて行う方法しかなく、開発プロジェクト単位で支援すべき機能は Weblog 上では提供することが出来ない。そこで、開発プロジェクトと開発者を関連付けるためのサブシステムを用意し、その機構を利用して開発プロジェクトを支援する仕組みを用意するべきである。この考えを実現するための機構として SNS に類似した機能を持ったものを考えている。このシステム上ではグループの作成、参加・脱退の機能の他に、仕様策定の前段階の状態での議論を行うフォーラムの提供やプロジェクト内から Weblog 上の仕様を把握する機能、スケジュール管理機能、プロジェクト内の開発者の RSS を収集して、RSS を再生成する機能などの実装を予定している。

● ソースコードの管理機構との連携

Weblog 上で管理される、仕様書やバグ報告書は、ソースコードと関連するものが多く、ソースコードを読む際に、どの仕様に関連しているものか、またはバグ報告書を読んだときに、どのソースコードに起因するものか、を参照したいという要求が発生する。オープンソースプロジェクトなどではソースコードの管理に CVS 等の構成管理ツールを用いている場合が多く、CVS と Weblog を連携させるための機構があれば、上記の要求を満たすことが出来る。このシステムは CVS を Web システムでラッピングし、CVS の通常操作やトラックバック・RSS を用いた Weblog や他のシステムと連携する機能の実装を予定している。

● 過去の類似プロジェクトの検索機能

Weblog、チーム開発の支援システム、CVS リポジトリなどの情報をシステム内に統一して管理できるため、過去の類似した開発を検索することが出来る機能があれば、その開発を担当した開発者への接点を作ることが出来たり、ノウハウを得たりすることが出来、開発者へ有用な情報の提供が行える。このシステムは各情報を収集しクラスタリングすることで実現しようと考えている。

これらのサブシステムを図にまとめると、図 6 のようなものとなる。

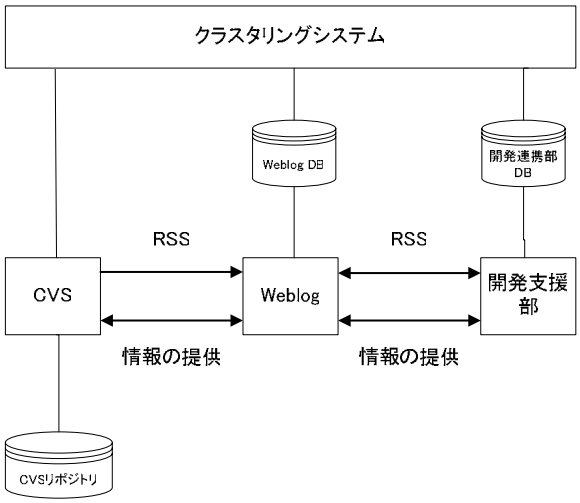

図 6 システムの全体図

今後は、これらのサブシステムを試作し、Weblog と連携させた結果などを報告していく予定である。

参考文献

[1] 長野 宏宜 : 分散ソフトウェア開発、共立出版、東京（1996）