情報処理学会
グループウェアとネットワークサービス研究会
2006年11月16日, 17日

グループによるソフトウェア開発演習における内省と協調による学習支援

櫨山淳雄　高山知佳　小林祐介　大瓶佳秀
東京学芸大学

ソフトウェア開発は知識集約的な協調作業である。ソフトウェア開発では問題解決活動が繰り返し行われる。本論文はグループによるソフトウェア開発演習において発生する問題の解決とその過程を通して獲得した知識を定着させることを目指した、内省と協調による学習プロセスモデルとその支援環境を提案している。

Learning Support by Reflection and Knowledge Collaboration in a Group-based Software Engineering Project Course

ATSUO HAZEYAMA　CHIKA TAKAYAMA　YUUSUKE KOBAYASHI　YOSHIHIDE OHGAME
Tokyo Gakugei University

Software development is a highly knowledge-intensive and collaborative activity. Problem resolution processes are performed iteratively during software development. The authors propose a learning model that is based on reflection and knowledge collaboration for problem resolution in a software engineering project course. They also describe an overview of a support system based on this model.

1. はじめに

ソフトウェア開発は協調活動を伴う知識集約的な作業である [4, 13]。ソフトウェア開発において問題解決活動が繰り返し行われる。ソフトウェア開発に従事する個々の開発者はソフトウェア開発に必要なすべての知識を有しているわけではなく、彼らは開発をしながら、同時に必要な情報の収集を行っている。あるいは、知識を有している他者に問い合わせを行うこともある[14]。葉雲文はソフトウェア開発における知識コラボレーションの重要性を指摘し、サンプルプログラムの登録・検索、過去の議論の閲覧、専門家への問い合わせを可能とするソフトウェア開発環境を提案している[14]。

一方、問題解決活動を通じて獲得した知識を定着し、学習効果を向上させるプロセスとして内省(reflection)がある。Reid は、内省を、「実践について学んだことを記述し、分析し、評価し、そしてそれを伝えるために実践経験を振り返るプロセス」と定義している[11]。

我々はグループによるソフトウェア開発演習教育に取り組んでいる [6]。この演習では、学習者がグループにより協調してソフトウェア開発を行う経験を通じて、ソフトウェア開発に必要な知識やスキルを習得することを目指している。本論文では、グループによるソフトウェア開発演習において発生する問題を解決し、問題解決により獲得した知識を定着させることを目指した、内省と協調による学習プロセスモデルとモデルに基づく支援環境を提案する。内省のための枠組として失敗学[5]の考え方を導入する。

本論文の構成を以下に示す。2 節では著者らが行っているソフトウェア開発グループ演習の概要について述べたあと、学習プロセスモデルを提案する。3 節では支援環境に対する要件を述べ、構築中の支援環境について述べる。4 節で関連研究との比較について述べ、最後にまとめと今後の課題を述べる。

2. 学習プロセスモデルの提案

2.1 ソフトウェア開発演習の概要

本研究が対象とするグループによるソフトウェア開発演習は、東京学芸大学情報教育専攻の学部 3 年生を対象とした選択科目である[6]。この科目に先立ち、ソフトウェア工学の入門に関する講義を行っている。そこでは、ソフトウェア開発プロセスモデル、要求分析・設計技法、オブジェクト指向と Unified Modeling Language (UML)、関係データベースの定義と操作、JSP/サーブレットによる Web アプリケーション開発について講義と演習を行っている。本演習では、プロジェクト管理、品質管理、構成管理に関する講義と、グループ形式によるオブジェクト指向ソフトウェア開発演習を行っている。受講者数は 20 名から 30 名であり、グループサイズは 1 グループ 4 名を標準として、3 名から 5 名で構成している。したがって、クラスに複数のグループが存在する。

各グループは教員が提示した課題の中から 1 つを選択し、選択した課題に対して、要求分析からテストまでの工程を経てシステムを完成させる。そして、グループを構成する全メンバーが全工程を経験することを求めている。

Tholander と Karlgren は学習者が学んだ知識を実際に使いこなすためには本物志向が必要と述べている[12]。本演習においても、課題によっては実務で利用する場合もある(例えば、情報処理センター職員からの要求に基づくシステムや、専攻の求人情報管理システムは実運用を行っている)。教員とティーチングアシスタント(本演習を受講した大学院生。以下 TA と記す)は上流工程（要求分析、設計）の成果物(UML ダイアグラム、画面設計書、データベース設計書)に対するインスペクションと開発されたシステムに対する受入テストを中心にグループの活動に関与している。開発期間は 3 ヶ月である。授業時間だけではすべての作業を行うには時間が足りないため、授業時間以外での活動も必要になる。したがって、分散環境下での開発形態をとることもある。そこで、分散環境下での成果物共有や非同期コミュニケーションのための支援システムを構築し、提供してきた[7,10]。

これまでも学習者が問題に直面したとき、対面でのコミュニケーション、電子掲示板 (Bulletin Board System: BBS)や電子メールによるコミュニケーションにより問題解決を行っていたが、問題解決情報は記録されていない、もしくは、BBS 等電子コミュニケーションのアーカイブ中に埋もれてしまっている。そのため、ノウハウの共有がなされておらず、他の学習者が同じような問題に直面するという状況が観測された。

2.2 学習プロセスモデル

本研究が提案するグループによるソフトウェア開発演習における問題解決の学習プロセスモデルを図 1 に示す。本学習プロセスモデルは、失敗学[5]を基盤としている。失敗学とは、失敗から学び、同じような失敗を繰り返さないことを目指した学問である。失敗学では失敗について正しく記述し、その失敗を他者に伝達するために、6 つの属性（事象、背景、経過、原因、対処、総括）を定義している。

本学習プロセスモデルは、ソフトウェア開発において、学習者が開発の各工程で直面する問題を解決する過程で、上記の各属性情報を記述させ、それらを学習者間で共有することによりソフトウェア開発における問題解決に必要なノウハウを習得することを目的としている。

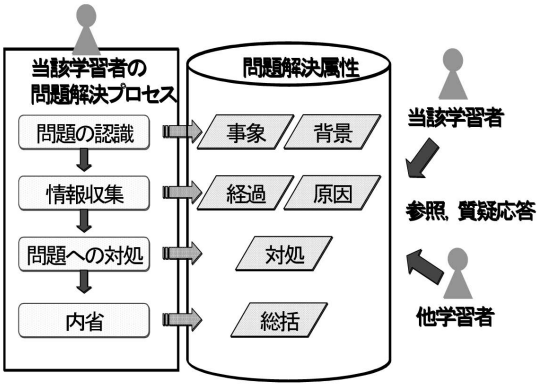

図 1　学習プロセスモデル

2.2.1 問題解決属性

本研究では、失敗学で定義された失敗情報の属性をソフトウェア開発の問題解決情報の属性として表 1 のように再定義した。以降に述べる学習プロセスにおいて、学習者は、この問題解決属性への記述を行う。

2.2.2 学習プロセス

本研究が想定する学習プロセスは、問題の認識、情報収集、問題への対処、内省のステップからなる。

(1) 問題の認識

学習者(あるいは学習者グループ)は問題に直面したとき、その事象、背景を識別する。問題としては、インスペクションやテストにより指摘された事項、プログラミングにおける不具合、開発環境構築におけるトラブル等が考えられる。

(2) 情報収集

学習者(学習者グループ)は問題解決に必要な情報を収集し、問題の原因を検討する。情報源として、本研究で構築する支援環境あるいは外部リソースに蓄積されている過去の事例や議論が考えられる。また、演習に参加している他学習者（グループメンバー、他グループのメンバー）や教授者(TA や教員)とのコミュニケーションにより情報を収集することも想定する。

(3) 問題への対処

(2)で収集した情報に基づき検討した原因を解決するために対処を行う。

(4) 内省

問題が解決できた場合には、問題解決プロセスを振り返り(内省)、問題解決情報を記録として残す。

以上のような学習プロセスを経て蓄積された問題解決情報は他の学習者にも参照可能となるよう共有される。他学習者が情報を参照した場合にフィードバック情報を提供したり、参照した情報に対して、自身の内省情報を連結することが期待される。

表 1　問題解決に関わる属性

属性	失敗学における定義	本研究による定義
事象	どのようなことがあったかに関する記述	発生した問題の内容に関する記述（エラーメッセージ等）と問題を含む成果物
背景	失敗の事象が生じた背景に関する記述	問題発生時の(開発・実行)環境・開発工程・ユースケース
経過	失敗の進行に関する記述	要求事項と、問題解決のために参照した情報
原因	失敗の原因に関する記述	問題の原因に関する記述
対処	失敗に対してどのようなことを行ったのかに関する記述	問題解決のために行ったことに関する記述と問題解決後の成果物
総括	失敗から学んだことに関する記述	問題解決から学んだ教訓に関する記述

3. 支援環境

本節では、前節で述べた枠組を実現する支援環境について述べる。3.1 項で機能要件を述べ、3.2 項で支援環境の実現について述べる。

3.1 機能要件

- 問題解決情報の記述
  開発期間の時間的制約の厳しさや開発に必要なノウハウの多様さから、学習者は問題に直面した時、さまざまな情報を参照しながら解決を試みる。その際、問題解決活動を振り返り、学んだことを記述することを通じて知識の定着を図るための仕掛けを提供する。記述を通じて当該学習者の知識の内面化につながるとともに、表出された記述は他者への共有情報（知識創造理論[9]における連結化の対象候補）となることが期待される。
- 協調活動支援
  学習者が自身のみで問題を解決できない場合、質疑応答等他者との協調活動が必要となる。問題を明らかにする習慣を身につけさせることは、グループメンバーにリスクに関するアウェアネス情報を早期に提示することにもつながり、プロジェクトの成否にとっても重要であると考える。
- 知識共有支援
  支援環境に蓄積される知識量が増加するにつれ、必要な情報を検索するコストが増大する。そこで効率的な検索を支援する仕掛けが必要となる。多くのナレッジマネジメントシステムでは投票やアクセスログのようなコンテキスト情報を提供し、情報を再利用する際の尺度としている[15]。本研究でもこれらの情報を活用する。それに加えて、問題情報を、それが発生した工程および「ユースケース」と対応付ける。ユースケースとはシステムが提供する機能単位である。類似の機能は同様の問題が発生する可能性が高いと考え、ユースケース名を検索における 1 つの手がかりとして用いることとする。
- 演習の組織構造の表現
  演習は毎年行われ、毎年の演習では複数の学習者により構成されるグループが複数存在する。また、各年度では複数の演習課題が提示され、各グループはその中から 1 つの課題を選択し、開発に取り組む。課題はユースケースを複数個含んでいる。このような組織構造を管理できる必要がある。

3.2 支援環境の実現

支援環境は特定のソフトウェアをインストールしなくても利用可能なように、JSP、サーブレットを用いた Web アプリケーションとして開発する。支援環境は、管理者に対する機能（グループ登録、受講者登録、課題登録）と学習者に対する機能群から構成されている。以下では学習者に対する主要機能である問題解決情報登録、情報閲覧（閲覧情報に対するフィードバック情報の登録を含む）、支援要請について説明する。

問題解決情報登録

図 2 は問題解決情報登録画面である。学習者が直面した問題の解決過程を振り返り、その問題解決過程について記述するものである。この画面では表 1 で示した属性に加えて、閲覧性を向上させるためのタイトルを記述する。当該問題解決情報と関連する他の問題解決情報や参照

した外部リソースへのリンク、問題解決前後の成果物を関連付けて登録する。

図 2　問題解決情報登録画面

問題解決情報閲覧

図 3 は支援環境に蓄積されている問題解決情報を閲覧するための画面である。

図 3　情報閲覧画面

情報を閲覧するためには登録情報一覧画面から、登録情報が持つタイトルをクリックすることによりこの詳細情報が表示される。この画面から、当該情報に対するフィードバック情報（有用さの評価やコメント）を記述する画面や、参照者がこの情報から学んだことを内省情報として記述する画面に推移する。この画面から別の問題解決情報を記述する画面に推移すると、タイトルが引き継がれ、関連する問題解決情報へのリンクとして参照元の ID が設定される。これが閲覧時にリンクとなり、相互参照を可能にする。この機能は組織的知識創造理論の連結化を支援するものと考えることができる。

支援要請

学習者が直面した問題に自ら解決策を見つけることができない場合に、他の学習者や教授者に支援を求める際に使用する機能である。他者に支援を求める場合には、他者に問題内容を伝えるために、問題(事象)、背景、自身が当該の問題に対して行ったことや想定される原因、当該問題に対する成果物を提供することを求める。この支援要請に対して、他の学習者が回答を記述することが可能である。

支援要請に関わる質疑応答の画面を図 4 に示す。コミュニケーションを通じて問題が解決した場合には、要請を行った学習者は問題解決情報を記述し、システムに登録する。この時、システムは当該問題解決情報とその問題解決のためになされた質疑応答の情報を関連付けて管理する。

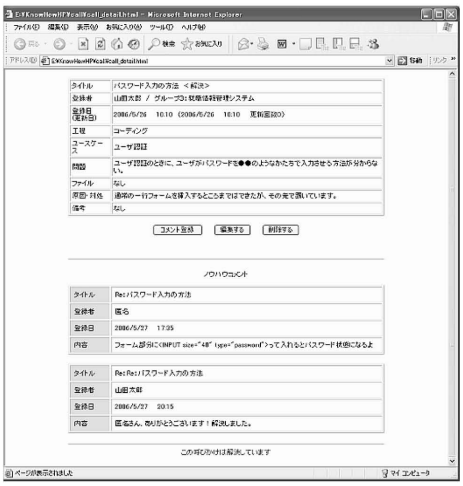

図 4　支援要請に対する質疑応答画面

4. 関連研究

本節では、本研究と関連する研究について述べ、それらと本研究の差異を述べる。本研究に関連する研究分野として、ソフトウェア開発における経験ベースの研究、バグ管理システムの研究、ソフトウェア工学教育の研究がある。

経験ベースは、V. Basili が提唱した経験工場のコンセプトの中核にあるデータベースである。経験ベースはソフトウェア開発組織で生み出される経験を蓄積し、再利用を可能にする基盤で

ある。蓄積された知識がプロジェクトに適用され、プロジェクトから収集したデータを分析し、経験ベースの中に蓄積される[3]。本支援環境は経験ベースの一種と考えることができるが、学習者に問題解決情報の記述を通じて内省を促進させることを主眼においている。

バグ管理システムは一般に、システムに含まれる不具合の報告から解決までの情報の記録、通知、共有を支援するものである(例えば、[1])。本支援環境が対象とする問題解決情報の一部はバグ管理システムに報告されるものが起点となる可能性がある。しかし、本支援環境はバグとして検出される以前の広範囲の問題とその解決に関する情報をその問題解決を通じて学んだ教訓を含めて記録、共有することを目指したものである。

また、近年オープンソースソフトウェア開発におけるツール群(CVS リポジトリ、メールアーカイブ、バグ管理データベース)を緩やかに連携させようという試みがある。Hipikat はその代表的な事例である[2]。Hipikat では、バグ管理データベースで管理されているバグ ID をそのバグの修正箇所にコメントとして埋め込むことを前提に、この関連にもとづいた検索支援を提供している。我々はこれまでに開発してきた支援環境[7, 10]と密に連携する支援環境構築を目指している。

Hazzan はソフトウェア工学教育に内省的視点を導入する重要性について述べている[8]。そこでは内省を促進するプロセスについて記述している。しかし、内省結果の表出化やそのための支援環境について述べていない。本論文では、内省と協調活動を支援する環境の構築を目指している。

5. おわりに

本論文では、グループによるソフトウェア開発演習における内省と協調活動による学習支援の枠組を提案した。そして、支援システムの機能について述べた。現在 3 節で述べた支援環境を構築中である。2006 年度後期の演習に実際に適用を行い、その有効性について評価を行う予定である。

謝辞

本研究の一部は、科学研究費補助金(C) 18500701 の助成のもとに行われている。記して謝意を表す。

参考文献

[1] The Mozilla Project's Bugzilla repository. http://bugzilla.mozilla.org/

[2] D. Cubranic, G. C. Murphy, J. Singer, and K. S. Booth, Hipikat: A Project Memory for Software Development, IEEE Transactions on Software Engineering, Vol. 31, No. 6, pp. 446-465, June 2005.

[3] T. Dingsoyr and R. Conradi, A Survey of Case Studies of the Use of Knowledge Management in Software Engineering, International Journal of Software Engineering and Knowledge Engineering, Vol. 12, No. 4, pp. 391-414, 2002.

[4] D. M. German, D. Cubranic, and M-A. D. Storey, A Framework for Describing and Understanding Mining Tools in Software Development, Proceedings the 2nd International Workshop on Mining Software Repositories (MSR 2005), pp. 95-99, ACM Press, 2005.

[5] 畑村洋太郎, 失敗学のすすめ, 講談社, 2000.

[6] A. Hazeyama, An Education Class on Design and Implementation of an Information System in a University and Its Evaluation, Proceedings of the 24th Annual International Computer Software and Applications Conference (COMPSAC2000), IEEE CS Press, pp. 21-27, October 2000.

[7] 櫨山淳雄, 中野秋子, ソフトウェア設計・開発グループ演習教育のためのコミュニケーション支援システム, 情報処理学会論文誌, Vol. 42, No.11, pp. 2550-2561, 2001 年 11 月.

[8] O. Hazzan, The reflective practitioner perspective in software engineering education, The Journal of Systems and Software, Vol. 63, pp. 161-171, 2002.

[9] I. Nonaka, and H. Takeuchi, The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation, Oxford University Press, 1995.

[10] Y. Ohgame, and A. Hazeyama, Design and Implementation of a Software Inspection Support System for UML Diagrams, IEICE Transactions on Information and Systems, Vol. E89-D, No. 4, pp. 1327-1336, April 2006.

[11] B. Reid, "But we're doing it already" Exploring a response to the concept of reflective practice in order to improve its facilitation. Nurse Ed. Today, Vol. 13, pp. 305-309, 1993.

[12] J. Tholander, and K. Karlgren, Support for Cognitive Apprenticeship in Object-Oriented Model Construction, Proceedings of Computer-Supported Collaborative Learning (CSCL2002), 2002.

[13] Y. Ye, and K. Kishida, First International Workshop on Supporting Knowledge Collaboration in Software Development (KCSD2005), in conjunction with 12th Asia-Pacific Software Engineering Conference (APSEC2005), 2005.

[14] Y. Ye, Socio-Technical Support for Knowledge Collaboration in Software Development

Tools, Proceedings of the Workshop on Integrating Software Engineering and Usability, pp. 39-51, 2005.

[15] M. Zacklad, Communities of Action: a Cognitive and Social Approach to the Design of CSCW Systems, Proceedings of the 2003 International ACM SIGGROUP Conference on Supporting Group Work, pp. 190-197, ACM Press, 2003.