ソフトウェア工学 99－7
（1994．7．7）

協調的仕様化作業を支援するツール
－ 作業分担者の支援

佐伯元司† 松村欣司‡ 郭文音‡*

† 東京工業大学 工学部 情報工学科
‡ 東京工業大学 理工学研究科 電気電子工学専攻

本論文では，ソフトウェアの仕様化・設計段階における協調作業を支援するツールの一例について述べる．作業分担が決まった後の分担作業の支援を行うことを目的とする．本ツールは，Method Base と呼ばれる個人の設計作業を支援する部分と，作業者間のコミュニケーションを支援する Product Lens とからなる．前者は，種々の仕様化・設計法を蓄積しておき，作業分担者が自分の担当部分に応じて，適切な設計法を選択できるようになっている．また，異なる設計法で作成されたプロダクトを統合する機構も有している．Product Lens は，電子メールとハイパーテキストをもとにした支援ツールで，テンプレートによる電子メールの作成，プロダクトの変更箇所の関係者への自動通知，受信メールを構造的に蓄積し検索する機能などを有している．

A Tool for Supporting Collaborative Specification Development

Motoshi Saeki† Kinji Matsumura‡ Kuo Wenyin‡*

† Department of Computer Science, Tokyo Institute of Technology
‡ Department of Electrical and Electronic Engineering, Tokyo Institute of Technology

This paper discusses a supporting tool for developing specifications in collaboration of a team — for collaborative specifications development processes. It consists of two parts — one is *method base* for individual use, and another is *product lens* for team use. Method base has various kind catalogued specification & design methods, so worker can select suitable methods for his problem domain from it. It has also the facility to integrate the products developed with multiple methods into one. Product lens system is based on E-mail system plus Hyper text, and it has following functions to compose and edit E-mails by using templates, to notify automatically the modification of the specifications to the relevant team members, to store and retrieve the received E-mails in structured way, and so on.

*現在 日立製作所

1 はじめに

ソフトウェアの仕様化・設計段階は，開発過程で初期段階に位置するため，高品質のソフトウェアを効率良く開発するうえで，その支援が重要である．ソフトウェア開発の多くは，複数の作業者（発注者やユーザも含む）の協調作業によって行なわれる．従って，1人の作業者の支援を目的とした従来のCASEツール[6]では，協調作業が行われる状況では，不十分であろう．ソフトウェア開発での協調作業がどのように行われているかを調べ，それに即した支援を行うことが効果的な支援につながる．特に，協調作業中にどのようなコミュニケーションが行なわれているかを調べ，それらの特徴を活かした支援ツールを作成することが，効果的な支援を行なう上で重要である．本稿では，ソフトウェア開発の中の仕様化・設計段階でどのような協調作業が行われているかを分析し，それらの特徴を活かしたソフトウェア仕様化・設計作業の支援ツールについて議論する．種々の形態の協調作業を支援するために，Speech Act Theoryやハイパーテキスト等を用いた支援システムも開発されているが[7, 9]，仕様化・設計作業の特徴を活かしたシステムとすることにより，より効果的な支援が期待できる[3]．

実際の典型的な協調的仕様化作業は，まず，作業分担やスケジュールを決めた開発計画が立てられ，それに従って個々の作業者が担当部分の作業を行う．担当作業が終了すると，その作業結果がレビューされ，1つの作業結果としてまとめられる．本稿では，個々の作業者が行う分担作業の支援に焦点を当てる．

2 分担作業支援の要件

作業計画を立てたり，作業結果をレビューしたりする作業は，主に対面式の会議で行われ，そこでのコミュニケーション量も多い．自分の担当箇所を開発する分担作業では，作業者は自分の作業のみに専念し，他の作業者とのコミュニケーションは少なくなる．問題の発生や進捗状況を報告したり，作業分担の境界を確認したりするコミュニケーションが主になる．また，作業場所が遠隔地に離れている場合，対面式会議のように他の作業者との同期をとって，コミュニケーションを行うのは不便である．この段階のコミュニケーション量がそれほど多くないことから，電子メール等の電子的手段を用いたほうが効果的と思われる．本支援ツールでは，コミュニケーション手段として，電子メールを用いる．従って，仕様化・設計方法論に基づいたCASEツールと電子メール機能を連結した支援ツールが必要である．電子メールをレビューに使用するための支援ツールも開発されているが[11]，本ツールでは仕様化・設計作業段階での電子メールの使用を支援する．

上記のような設定のもとで，支援ツールに要求される機能をあげると，以下のようになる．

1. 作業計画に従った，分担者の作業の支援．
2. 方法論に従った個人の仕様化・設計作業の支援．作業の成果物を構造的に蓄積・管理したり，作業履歴を蓄積したり，方法論に従って作業者をナビゲートしたりする．さらには，図式表現などの取り扱いといった親和性のよいユーザインタフェースの提供．
3. 種々の方法論の支援．担当する箇所に応じて各担当者が異なる方法論を用いる可能性があるため，各種の方法論を支援し，成果物を統合する機能．
4. 電子メールを他の作業者へ送信したり，受信したりする機能．電子メールの作成を支援する機能．
5. 他の作業者が作成中のプロダクトを参照する機能．

支援ツールは，方法論に従った支援を行う部分(Method Baseと呼ぶ)と電子メール等のたの作業者の通信を支援する部分(Product Lensと呼ぶ)の2つから成る．

3 Method Base

Method Baseシステムは，オブジェクト指向分析・設計法[1, 2]や構造化分析・設計法[4, 10]といった種々の方法論を蓄積した一種のデータベースシステムで，ユーザはこれらを適宜選択して支援を受けることができる．方法論以外に，作成中のプロダクトや作業履歴が関係付けられて蓄積される．また，異種の方法論で作成されたプロダクトの統合化機能も有している[8]．本システムのユーザは，Method Baseシステムへのアクセスを，Overview WindowとEdit Windowと呼ばれる2種類の画面に表示されるWindowを介して行う．Overview Windowは，作業計画に従って作業を進めることを支援するためのもので，作業者の作業分担，使用する方法論，進捗状況等の一覧が表示される．Edit

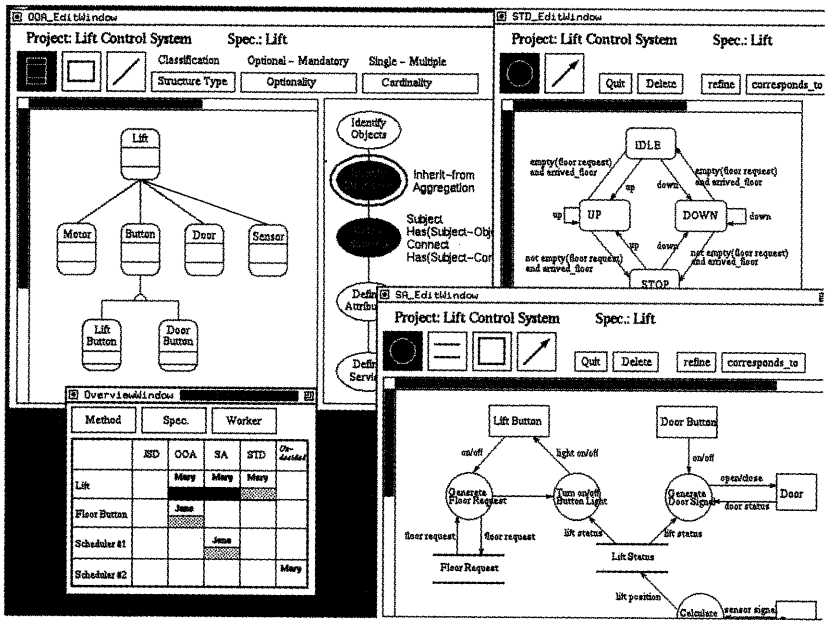


図 1: Method Base の画面例

Window は，1 つの方法論に従って作業を進めることを支援する．すなわち，プロダクトの入力・編集作業を支援したり，作業手順をガイドしたりする．この様子を図 1に示す．図の例は，Lift Control System を設計する例で，作業者は Mary と Jane の 2 人が割り当てられている．図中には，OOA，STD(状態遷移図)，SA(データフロー図) 用の 3 つの Edit Window と，左下に Overview Window が表示されている．表の行は分担された作業名を，列は使用する方法論名が表示されており，表の要素には担当者名が書き込まれる．例えば，Lift の設計は OOA，SA，STD の 3 つの方法論を用いて行われ，すべて Mary が担当することになっている．

1 つの Edit Window は 2 つに仕切られており，その左側部分はプロダクトの入力・編集用，右側部分は作業手順を表示する領域である．入力・編集用にどのようなコマンドが用意されているかは方法論ごとに異なっており，Window の上方にコマンドメニューが表示される．どの方法論にも共通なコマンドは，プロダクト要素 (例えば，OOA ではオブジェクトなど) の入力，消去，移動，サイズの変更 (図表現の場合) などである．ユーザは右領域の作業手順表示をコマンド入力として使用することもできる．表示されている作業手順を選択すると，その手順の出力要素を入力するコマンドが入力されたのと同じことになる．例えば，図 1 中の作業手順 Identify Objects を選択すると，オブジェクトの入力を行うことができる．

異なる方法論で作成されたプロダクトは図 2に示すように，各方法論に共通な概念要素を介して統合される．OOA で作成されたプロダクトと SA を統合することを考えてみよう．OOA のオブジェクト概念は，SA では Source&Sink もしくは Data Store に対応する．従って，図 1の OOA の Edit Window に出現している Lift Button と SA に出現している Lift Button は同じものを表していると考えることができる．OOA と SA で作られた 2 つの異なるプロダクトは，Lift Button のような共通に出現する要素を介して連結される．

各方法論は，実体関連モデルに基づいて記述され，そのうちの汎用部分がプロダクトの統合に使用されるため，図 3のように Generic Model としてあらかじめ Method Base システムに用意されている．新しい方法論を登録する場合，汎用部分以外の部分，汎用部分との関係を実体関連モデルを用いて定義する必要がある．これ以外にも方法論固有の表記法を扱うためのエディタを作成する必要がある [12].

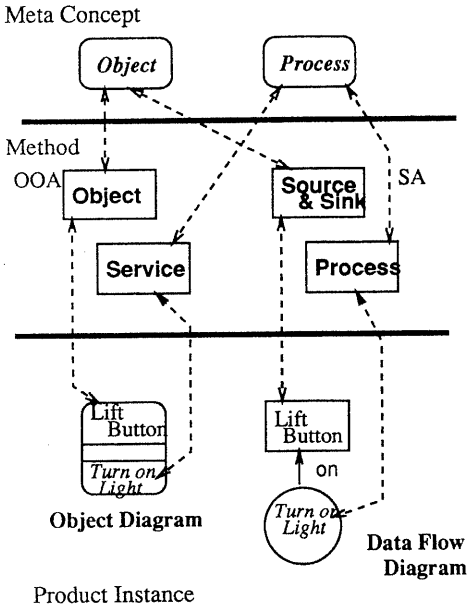


図 2: 共通概念によるプロダクトの統合

4 Prodcut Lens

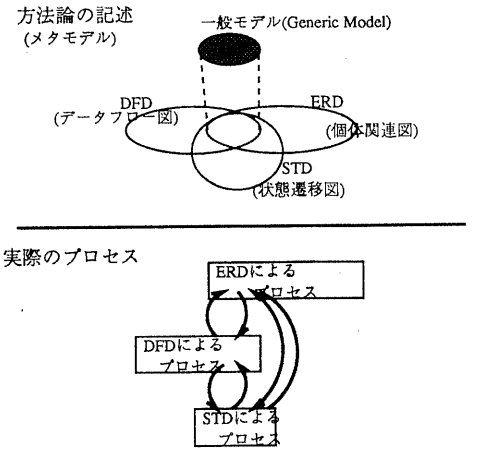
作業者間のコミュニケーションは，電子メールを用いて行う．実際の設計作業においてどのような内容が電子メールでやりとりされているかの例を表 1 に挙げる．

表 1: 電子メールによる通信内容

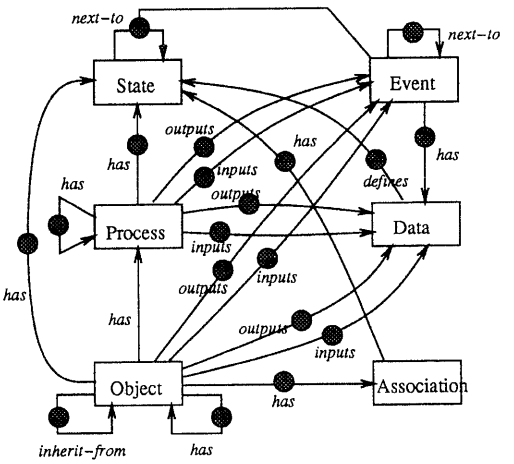
特徴	メールの数
進捗状況報告・スケジュール調整	8
用語の定義	4
未決事項の審議	11
他の作業者のプロダクトへのコメントと返答	16
決定事項の確認	4
問題提起と決定事項の変更	18
その他	10
合計	64(71-7)

開発対象:スケジュール管理システム
作業者:2 名　　期間:14 日　　メール総数:64
作業中に対面式会議 1 回開催
7 通が 2 つのカテゴリ分類される内容を含んでいた

電子メールは，対面式会議のような同期通信方



(a) 共通モデルの考え方



(b) 共通モデルのスキーマ

図 3: 共通モデル

式ではなく非同期通信であるため，従来の電子メールシステムをそのまま利用するには，問題点があると思われる．実際に使用した作業者へのインタビューをもとに電子メールに対する使い勝手をまとめると，以下のようになる．

1. 同期式通信と異なり，相手からのメッセージの受信中で，インタラプトをかけることができず，到着した電子メールをまず最後まで読まなければいけない．

2. 文中に未定義語があったり，意味解釈が難しい部分が含まれていることがある．

3. 結論がどうなったのかわからないことがある．

4. 電子メールでの議論はある人が述べた意見に一人が返答を返すだけという1対1の議論になりがちで，参加者全体に議論が発展していかない．

5. 実際に頭の中で考えている時と，関連した話題の電子メールを読む時にずれがあり，それが内容の理解を妨げることがある．

6. 電子メールが送られて来た時刻と実際に読む時刻とに大きな遅れが出ることがあり，全体の議論に1人取り残されることがある．

上記のような問題を解決するためには，メールの作成者に内容が提案であるか，報告であるかといったメールの目的や主旨を指定させ，読者に作成者の意図が明確に伝わるようなメールを迅速に作成できるようにし，他人から来た電子メールに対しては受信した電子メールを内容に応じて整理し，検索が容易に行えるような支援が必要である．本システムでは，表1に示したような通信内容を考慮し，以下の機能を用意した．

1. テンプレートを用いて電子メールを作成する機能．メールの目的や主旨に応じて，提案，質問，返答，要求，通知，問題，補足(以前に出した電子メールの補足・修正)の7種類のテンプレートを用意した．受信者は，テンプレートの種類により，自分がどのようなレスポンスを要求されているのかが直ちにわかる．

2. 電子メールの本文とプロダクトを関連付ける機能．電子メール中で作成中のプロダクト部分を参照したいとき，Edit Window 中に表示されたその部分にリンク機能を用いて電子メールを張りつけることができる．

3. 受信した電子メールをその話題と種類に応じて，構造的に整理蓄積する機能．Subject 欄に書かれている話題，言及しているプロダクトごとにフォルダを用意し，受信したメールを自動的に蓄積する．各メールは，何を参照・引用しているか，どのメールに対する返答か，どのメールに対する補足かといった関係も自動的に保存される．

4. 進捗状況などの定時報告の自動送信，変更があった場合の関係者への自動通知，作業計画に基づくプロダクトの自動配布といった自動送信機能．

テンプレートを用いた電子メールの作成手法や電子メールの自動送信機能，メールが到着したときの処理の定義機能を備えた電子メールシステムも開発されているが[5]，汎用を目指したものである．本システムでは，仕様化・設計作業に特化することにより，例えばテンプレートの種類を絞ることなどにより，より効果的な支援を目標としている．

図4に実際の使用例を示す．この例は，Lift 部分を OOA を用いて作業している Mary のプロダクトを，参照したもう一人の作業者 Jane がコメントを送ろうとしているところである．まず，メール作成コマンドをクリックすると，一般的なメールのテンプレートが表示される．このテンプレートは，宛先をしめす TO:，送信者を示す FROM:，参照しているものを指す REGARDING:欄などから成る．この部分は，通常の郵便の封筒に該当する部分である．メールの本文は，TEXT:欄に記述する．定型的な文の作成を迅速に行なえるように7種類のテンプレートが用意されている．図4の例では，受信者の Mary に作成中のプロダクトに問題があることを伝えるため，問題提起を表すテンプレート Problem を呼び出して使用している．1つのテンプレートには，1つの話題しか記述できないようになっており，これにより読者も送信者の意図を理解しやすいと思われる．1つのメールに複数のテンプレートを入れて送ることができる．メールの本文で，プロダクトのある部分を参照したい場合は，そのメールと参照部分とをリンクで関連付ける．図4の例では，Mary が作成中のプロダクトの Button 部分を参照しているため，Edit Window 中の Button 部分をクリックし，作成中のメールとの間にリンクを貼ることにより，メールの TO:欄，REGARDING:欄が自動的に埋められる．

図5のように，受信者の作業ディレクトリの下には，REGARDING:欄，SUBJECT:欄(話題を表

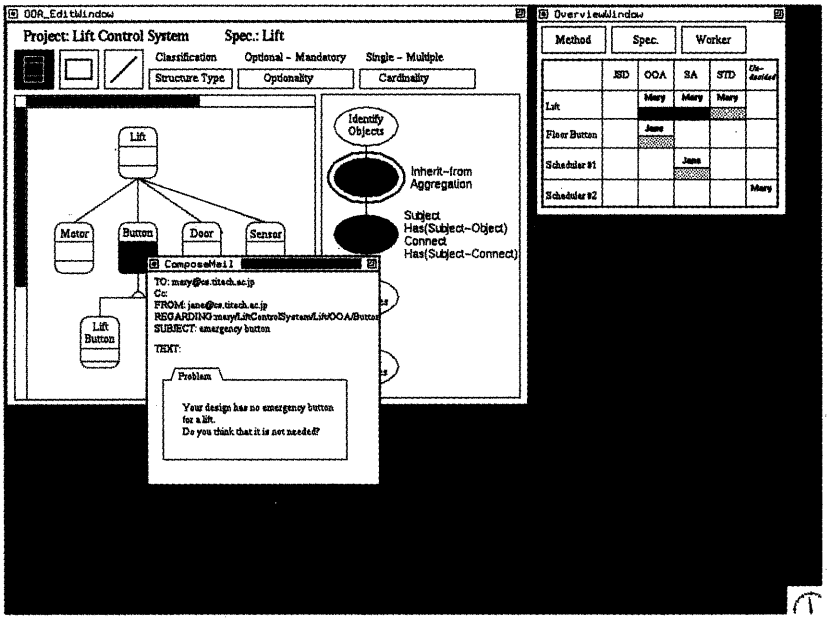


図 4: 電子メールの作成例

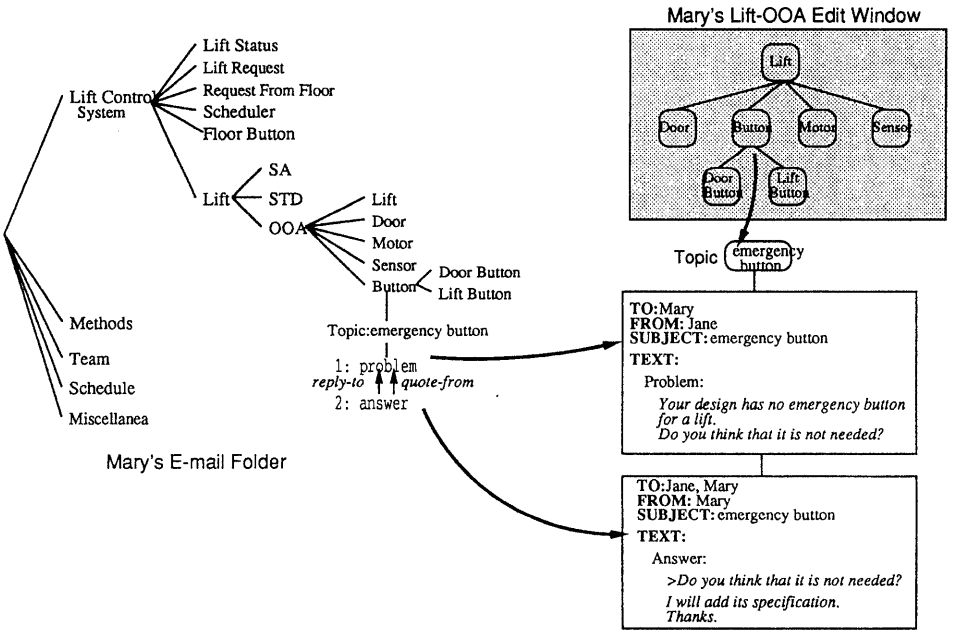


図 5: 電子メールのフォルダと仕様書の生成物との対応構造

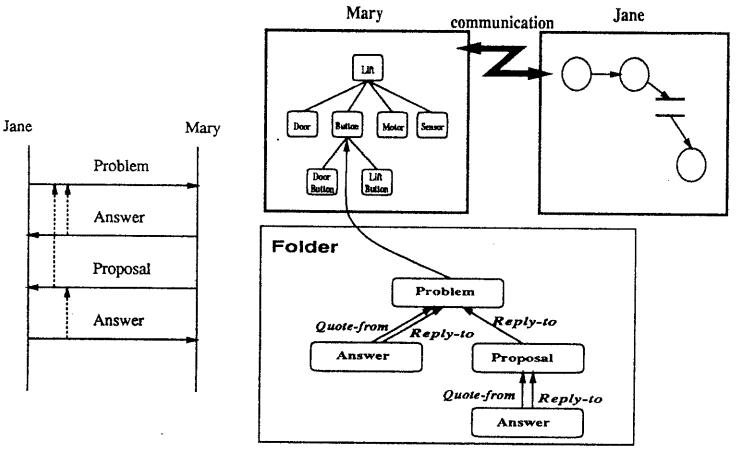


図 6: 電子メールが交わされた時のメールフォルダ内の構造例

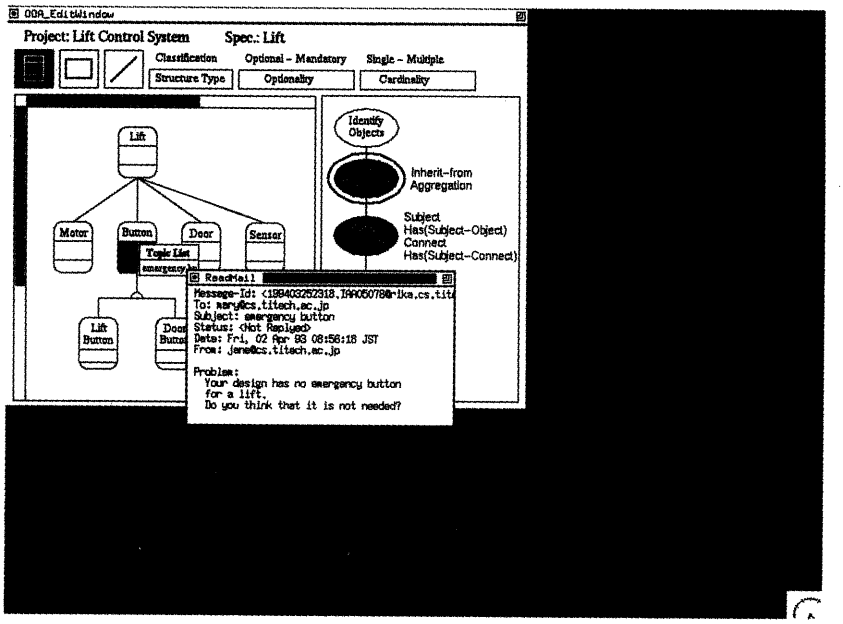


図 7: 受信メールの表示

す) の内容に応じて階層的にフォルダが生成され，同じ内容を持つメールが格納される．メール間には，どのメールに対する返答か (Reply-to)，どのメールを引用しているか (Quote-From) しているかの関係がある．フォルダに格納されたメールは，これらの関係に基づいてネットワーク構造で関係付けられ，ハイパーテキストとして検索できるようになっている．図6は，メールが交わされた時の，メールフォルダ内の構造を示している．Mary が作成中のプロダクトである Button について，図6の左側に示したような流れでメールがやり取りされた場合，フォルダ内は図のような階層構造になる．

受信した電子メールを検索して読むには，Browser を用いるやり方と，参照しているもの (REGARDING 欄の内容)，話題 (SUBJECT 欄の内容) のリストをメニューとして表示させ，選択するやり方がある．後者の場合，特に参照しているものがプロダクト要素のときは，Edit Window 中に表示されているプロダクト要素をフォルダ選択のメニューとして使用することもできる．図7は，Mary が受信した Jane からのメールを，Edit Window 中の Button 部分をクリックし，表示された話題リストから Emergency Button に関する話題を選択し，メールを読んでいる様子である．逆にプロダクト要素を参照しているメールを読もうとしたとき，参照しているプロダクトも Edit Windowt 中に表示される．また，1つのメールから，リンクによって関連付けられた他のメールも呼び出すことも可能である．これは，ノードがメールであるようなネットワーク構造でメールが格納されており，ハイパーテキストとして，ネットワークに沿って辿ることができるからである．メールを格納するための論理スキーマを図8 に示す．

5 おわりに

この論文では，ソフトウェア仕様化・設計の段階で，作業分担者の作業を支援するツールを考案した．対面式の会議などの作業形態の支援法およびそのツールとの統合を行うことにより，協調作業の継ぎ目のない (Seamless) 支援が可能である．このようなツールとの統合手法，運用・評価は今後の課題である．また，仕様化・設計段階では，発注者とのヒアリングを行ったり，作業分担が議論されたり，レビューが行なわれたりする．従って，発注者からの要求獲得法，作業分担の方法論やレビュー法とのコミュニケーション支援の組み合わせも考える必要があろう．

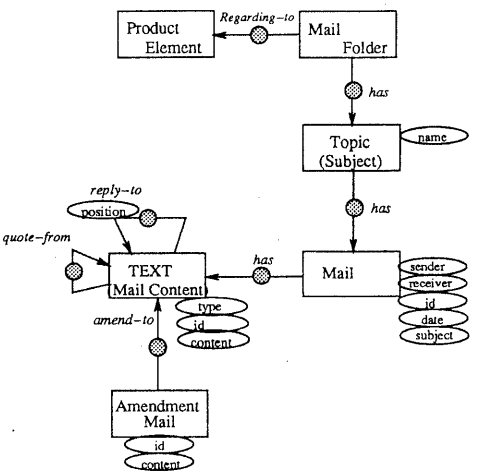


図 8: メールの論理スキーマ

参考文献

[1] P. Coad and E. Yourdon. *Object-Oriented Analysis.* Prentice Hall, 1990.

[2] P. Coad and E. Yourdon. *Object-Oriented Design.* Prentice-Hall, 1991.

[3] B. Curtis. Implication from Empirical Studies of the Software Design Process. In *Proc. of Int. Conf. by IPSJ to Commemorate the 30th Anniversary.*

[4] T. DeMarco. *Structured Analysis and System Specification.* Yourdon Press, 1978.

[5] K. Lai, T. W. Malone, and K. Yu. Object Lens : A Spreadsheet for Cooperative Work. *ACM Trans. on Office Information Systems*, Vol. 6, No. 4, pp. 332–353, 1988.

[6] T.G. Lewis. *CASE : Computer-Aided Software Engineering.* Van Nostrand Reinhold, 1991.

[7] G. L. Rein and C. A. Ellis. rIBIS : A Real-Time Group Hypertext System. *Int. J. Man-Machine Studies*, No. 34, pp. 349–367, 1991.

[8] M. Saeki, K. Iguchi, K. Wen-yin, and M. Shinohara. A Meta-Model for Representing Software Specification & Design Methods. In *Information System Development Process*, pp. 149–166. North-Holland, 1993.

[9] T. Winograd. Where the action is. *BYTE*, Vol. 13, No. 13, pp. 256–260, 1988.

[10] E. Yourdon and L.L Constantine. *Structured Design : Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice-Hall, 1979.

[11] 金政, 松本, 垂水, 田渕. 電子メール基盤「め組」を利用したソフトウェアプロセス管理–究仙–. 情報処理学会ソフトウェアプロセスシンポジウム, pp. 87–96, 1994.

[12] 篠原, 井口, 佐伯. ソフトウェア仕様化・設計法のデータベースの試作. 情報処理学会ソフトウェア工学研究会, Vol. 90, No. 14, pp. 105–112, 1993.