ソフトウェア仕様作成会議支援ツールの設計

海谷治彦　　佐伯元司

東京工業大学 工学部 電気電子工学科

〒 152 東京都目黒区大岡山 2-12-1

実際のソフトウェア仕様作成会議の分析から得られた結果を基に会議支援ツールであるグループウェアを設計した. 実際の会議では, 会議上の有益な情報が仕様書に記述されていない場合や, 会議の進行を妨げるような議論や, 決定事項の変更による矛盾の発生などが数多く見られた. 我々のツールはこのような作業の不備な点を支援する.

和文キーワード: 協調作業, CASE, グループウェア, 要求工学, ユーザーインターフェイス

A Supporting Tool for Face-to-face Meetings to Develop Software Specifications

Haruhiko Kaiya　　Motoshi Saeki

Department of Electrical and Electronic Engineering, Tokyo Institute of Technology

Ookayama 2-12-1, Meguro-ku, Tokyo 152, Japan

E-mail address: kaiya@cs.titech.ac.jp　　saeki@cs.titech.ac.jp

In this paper, we introduce a new type of a groupware based on the results of analysis to support the cooperative tasks. In the meetings to develop software specifications, we have found many missing of items and/or explanations in the specifications, conversation to hinder the advance of the meeetings and conclusions without consistency. Our groupware will support the users to decrease such faults.

英文 key words: Cooperative Work, CASE, Groupware, Requirements Engineering, User Interface

1 はじめに

ソフトウェア開発における要求仕様作成段階では多種多様な作業者による共同作業が行なわれる. そして, その段階における代表的な作業形態である会議などでは, 口頭による会話によって作業者間の情報交換を行なうことが多い. 我々がソフトウェアの仕様作成会議を支援する目的で分析対象とした会議 [9] の多くは, 作業時間の多くを作業者間の会話に費やしている. また, Gary ら [4] が分析した設計会議の多くも作業時間の 9 割を会話に費やしていることが報告されている. 通常のソフトウェアの仕様作成は複数回の会議を通して行なわれ, 仕様書や中間生産物である文書は会議外で作成されることが多い [7] .

このようなソフトウェアの仕様作成作業支援するための協調作業モデルやツールは数多く作成されている [1], [5], [2], [8], [14], [11], [13] . しかし, それらのツールの多くは, 利用者にそれぞれのツール特有の作業規則を強いる形のものが多く, それゆえ実際の利用者はそのツールを使って彼らなりの方法で作業を進めて行くことが困難である場合が多い. そして, その事実がグループウェアの導入の障壁となっていると考えられる.

我々はこのような状況を踏まえ, コンピュータツールであるグループウェアの導入の障壁を出来る限り軽減し, 人間であるグループウェア利用者が自分流の快適な作業を成就できる方法が必要であると考えた. そのような方法を獲得するために, 計算機を導入していない “普通の” ソフトウェアの仕様作成会議の内容を分析する研究を行なってきた [9, 7] .

本報告では, それらの分析から得た結果を基に設計した全く新しい形のグループウェアを紹介する. 2章で, 我々が対象とする仕様作成会議を明らかにし, その会議支援のための方針を述べる. 3章で, 我々が実作業の観察から得た結果について述べる. 4章では, それらの結果を基に仕様作成会議を支援するためのツールについて概観する. 5章では, そのようなツールを実際にどのように利用して生産物である仕様書を効率的に作成するかについて述べる. 6章では, このツールを実作業に適用した場合の評価の方法について述べる.

2 グループウェアの設計方針

ソフトウェアの仕様作成会議とは, その名の示す通り, ソフトウェアの仕様書という文書を作成するための会議である. ソフトウェアの仕様書を記述するための言語として, 自然言語以外の形式的な言語を利用する研究や効用について数多く議論されている. しかし, 我々は利用者自身の活動を出来る限り阻害しないことを目的の一つとしているので, 自然言語で記述する仕様書を対象とする.

ソフトウェアの仕様作成会議など, ソフトウェア開発における上流工程では, 計算機の専門家だけではなく, 作成されるシステムを注文する顧客や, それを利用するユーザーなどの計算機の専門家ではない作業者が参加して議論することが有効であると考えられる [3] . 我々の対象とする会議には, 以下のような種類の作業者の内のいくつかが参加することを想定する. 本報告では, これらの作業者の種類を役割と呼ぶ.

発注者 : 仕様化するシステムの製作を依頼する作業者の種類. 通常, 製作に利用できる資源 (予算, 時間) などを制限する.

ユーザー : 仕様化するシステムを実際に利用する作業者. (本文ではユーザーと利用者は別の意味で使い分ける. 利用者は会議に参加する全ての作業者を指す.)

設計者 : 実際の仕様書を記述する作業者.

製作者 : 仕様化するシステムを計算機システムとして実現する作業者. プログラマーなど.

調整者 : 会議の進行などを管理する作業者.

保守者 : 作成されたシステムの運用する場合, 保守を行なう作業者.

我々のツールは, 1 回の会議のみで終了する作業ではなく, 複数回の会議と, その合間に行なわれる会議で得た記録を有効な生産物として情報化する作業も含めた支援を行なう. 我々のツールでは, 多くのデータを保存し, それをツール利用者に提供するが, 記録という用語でより作業者の生に行為に近いデータを表し, 情報という用語で加工されたデータを表す.

我々のツールは, 個々の会議の中をその議論の内容によって複数の段階に分けて支援を行なう. その段階とは,

発注段階 : 発注者が作成対象のシステムについての説明を行なう段階. 通常, 調整者と発注者の対話の形で議論が進む.

計画段階 : 会議のための時間配分や, 後日の会議の日程や, 作業の分担などを決める. 通常, 調整者とそれ以外の作業者の対話の形で議論が進む.

説明段階 : 設計者が決定した仕様を説明する段階.

レビュー段階 : 作成対象システムに関する決定事項をレビューする段階. 例えば, 作成対象システムの動作を追跡したり, 機能を確認したり, 構造を確認したりする.

などが上げられる. それぞれの段階における発言を発言者の役割によって分類し, その役割の時間的な変化を調べた結果, 段階毎に特徴があることが分かっている [9] . 会議の目標は最終生産物である仕様書を作成することであるが, そこに至るまでに多くの中間生産物の作成が必要とされる. 中間生産物は, 上記段階によって異なるため, それぞれの会議の段階に応じた支援が必要と思われる.

3　実作業の観察からの結果

我々は実際にソフトウェアの仕様を作成する会議をビデオカメラで記録し, その記録内容のプロトコル解析を行なった. この分析の目的は, 計算機を利用しない会議にはどのような不備な点があるかを明らかにすることである. そして, その不備な点を含む作業をツールでの支援対象とする. 以下に分析から得られた結果を示す.

1. 項目や説明の欠落: 会議では議論され, かつ否決されていないにも関わらず, 仕様書に記述が残っていない項目や説明が数多く見られた. 表 1に, 実際の欠落の回数の割合のデータを示す.

表 1: 欠落の回数の割合

	無欠落 (%)	半欠落 (%)	全欠落 (%)
プロジェクト 1	41.3	22.4	36.3
プロジェクト 2	70.5	22.8	6.7
プロジェクト 3	61.3	31.0	7.7

これは 3 つのソフトウェア仕様化のプロジェクト (それぞれ複数回の会議から成る) についての仕様書に記述が残っていない項目や説明の割合である. 半欠落とは, 仕様書に項目の記述は残っているが, 議論された説明の記述が残っていない場合の割合である. 例えば, プロジェクト 3 では, 31.0 % が半欠落している. 全欠落とは, 議論された項目も説明も全く仕様書に残っていない場合の割合である. 例えば, プロジェクト 1 では, 36.3 % が全欠落している.

2. 会議の進行を阻害すると思われる議論: 同じ内容の議論の繰り返しや, 決定事項の変更や, 結論を出さない議論などの会議の進行を阻害する議論が数多く見られた. 表 2に会議の進行を阻害する議論の時間の割合を示す.

表 2: 進行を阻害する議論の時間の割合

	繰り返しの議論 (%)	決定を変更した議論 (%)	結論のない議論 (%)
プロジェクト 1	21.7	2.5	25.0
プロジェクト 2	1.8	5.0	2.0

例えば, プロジェクト 1 では, 全会議時間の 21.7% が繰り返しの議論になっているのに対して, プロジェクト 2 は 1.8 % と少ない. ここでの繰り返しの議論と決定を変更した議論の数値は, 一番最初に議論された部分の時間 (すなわち, この部分の議論が繰り返されたり変更されたりする) も含めて数値を計算しているが, 一番最初に議論された部分の時間が長いなどの特徴は見られなかったため問題ないと考えている.

3. ある結論の変更によって再考すべき事項に関する議論が行なわれない場合が数多く見られた. それによって, 矛盾のある仕様書が作成されてしまった. この事例に関しては割合のデータはないが, そのパターンとして,
   - 組み合わせによる再考の必要性.
     例) 塗りつぶした図形の移動や変形.
   - 対称性に関する再考の必要性.
     例) 画面上の表現とプリンタ出力の対応.
   - 用語の変更に伴う再考の必要:
     用語の変更や統合が仕様書に反映されていない例.

   などが見られた.

我々はこのような点に考慮して仕様作成会議を支援するツールを設計して行く.

4　ツールの概要

観察から得た結果から, 会議の行なう作業者, すなわちツールの利用者は, 効率的な会議を行なうために以下のような作業を議論そのものに加えて行なう必要があると考えられる.

1. 項目や説明の欠落を発生させないために, 仕様書に記述されるべき項目や説明の発生源となった利用者の発言を直接記録し, その発言と項目との関係を保存する.
2. 会議の進行を妨げる会話を回避するために, 利用者は現在議論されている話題に対応する項目がすでに決定されたことであるか否かを調査しながら会議を行なう.
3. ある項目の変更に伴い同様に変更しなければならない項目や説明を調査するために, 利用者は変更の波及するであろう項目間の関係を記録する.

4.1　ツールのデータ構造

ここで仕様作成会議の概念を説明するために利用していた用語にいくつかの用語を加え, それらの用語をツールの設計のために明確化する.

項目: 会議中の “話題” に対応する概念であり, 生産物中の項目に対応するラベルである.

説明: 項目を説明するための文章や図などの記述.

事項: 行為によって言及された項目と説明の対. 説明が添付されずに項目のみが言及される場合がある. 言及の型としては,

提案 : 新しい項目, 説明を付け加える.

削除 : 既存の項目, 説明を削除する.

参照 : 既存の項目, 説明を参照する.

の 3 種がある.

生産物: 利用者に出力される生産物. 具体的には仕様書, 議事録などであり, 項目と説明の対の列から作成される.

行為: 利用者の行なう動作の計算機内での認識の単位. 発言や黒板などに図を記述する (図示) 動作やその集まりなどに対応する.

段階: 物理的に連続した作業を作業内容によって分割した断片的作業. 前述の, “発注段階”, “計画段階”, “説明段階” などに対応する. 段階は行為の列として表現される.

形態: 物理的に連続した作業に対応する. 例えば, 1 回目の会議, 3 回目の会議外のまとめ作業などに相当する. 現在の形態は後述の “同期” か “非同期” のどちらかの型を持つ. 形態は段階の列として表現される.

図 1にデータ構造の関係を図示する. 図中では, 灰色の矩形

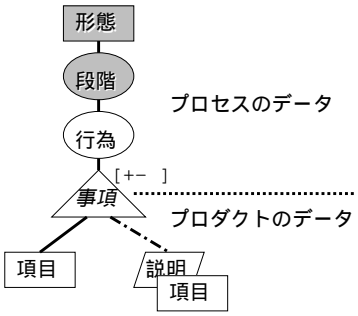

図 1: データ構造の関係

が形態, 灰色の楕円が段階, 楕円が行為, 矩形が項目, 並行四辺形が説明, 三角形が事項を表している. 事項はプロセスとプロダクトを関係付けるデータ構造である. 事項と実線で関係付けられている項目と, 破線で関係付けられている項目または説明があり, このことによって項目に関する説明, もしくは関連する項目を表現している. 事項に付加されている “[+−]” の記号は, それぞれ “提案”, “削除”, “参照” の型を表現している.

図 2に上記の構造のデータが構成される例を示す. この例では “会議 2” という形態の中で “発注”, “レビュー”, “説明” という段階が順に行なわれていることを示している. “レビュー” という段階の中では “顧客”, “ユーザー”, “顧客”, “設計者”, “顧客” の役割の作業者が順に “行為 1” から “行為 5” の行為を行なっている. それぞれの行為は, その実体が何であるかを保存しており, 例えば, 行為 1 は “発言” であるが, 行為 3 は “発言” と “図示” の組み合わせであることが表現されている. “行為 1” は “項目 1” と “事項 1” によって関係付けられている. “事項 1” は直観的には「 “項目 1” という話題が “行為 1” によって “提案” された」ことを示している. “行為 3” は “項目 1”,“説明 2” と “事項 3” によって関係付けられている. “事項 3” は直観的には「 “項目 1” に関する “説明 2” が “行為 3” によって “提案” された」ことを表現している. それに対して, “事項 5” は “説明 2” の “削除” を表現している.

図 3に, 図 2の行為 5 終了後に対応した生産物の基となるデータの例を示す. “説明 2” は行為 5 で削除されたために生産物には記述されない. “項目 2” は “事項 4” によって “項目 1” に関連することが記録されているので, 段を落して記述してある. 項目, 説明の具体的な内容は, それが提案された行為の記録 (すなわち発言/図示の内容) に従って作業者が記述する.

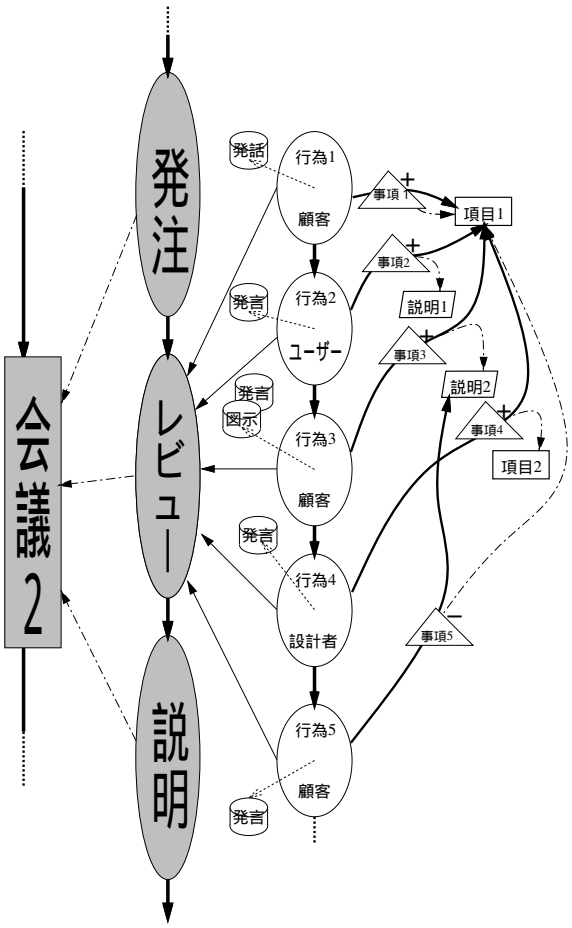

図 2: データの例

項目1：内容ボタン
説明1：内容を記述する場合に押す.
項目2：内容Window

図 3: 生産物の基となるデータの例

4.2 作業形態

実際の作業の観察/分析から得た知見として, 会議参加者が会議の進行と同時に, そこで議論された内容を十分に記録/整理することは困難であることが分かった. また, 1 つのソフトウェアの仕様作成を行なう場合, 複数回の会議を行なうのが普通である. そこで, ソフトウェアの仕様作成会議を支援するツールの利用者は, 会議中には議論の進行を阻害しない範囲で, その内容を記録/整理し, それ以外の仕様を作成するための複雑な処理は会議外で行なうべきであると考えられる. 我々のツールは, このような会議内と会議外の相異る作業形態を統合的に支援する. 我々のツールでは, 会議内において, その進行に伴い作業をするツール利用者を支援する形態を同期形態と呼び, 会議外において仕様の作成を含む複雑な処理を支援する形態を非同期形態と呼ぶ. 現段階では, この 2 つの形態のみの支援するが, 今

後, 別の形態 (例えば分散環境の形態など) の支援の追加を考慮して 4.1章で述べた形態というデータ構造を導入した.

4.2.1 同期形態

ツールの利用者は会議中において自分達の現在の議論内容を記録し, 過去議論された内容を検索することで内容の欠落や矛盾の発生を防止しなければならない. 同期形態のツールは利用者のこのような作業を支援する. ツールの利用者は, 計算機による支援を受けない通常の会議と同様に対面式の会議を行なう. それぞれの利用者は与えられたマイクロフォンを通して発言を行ない, 発言単位でその記録を計算機に蓄積する. マイクロフォンは個々の作業者に与えられているので, 蓄積された発言がどの作業者によるものかを自動的に認識できる. 同様に, 黒板などに記述された図なども, ビデオカメラを用いて直接, 計算機内に蓄積する. これらの記録には, 行為者と時刻の情報が付加される.

蓄積された発言/図示などの記録に以下のような情報を利用者自身が同期的に付加することが許されている.

1. “提案”, “削除”, “参照” のどの種類の事項になるか明記する.
2. どの “項目” に関する事項か明記する. 新たな項目の場合, 新たな項目の作成をツールに要求する.
3. どの “説明” に関する事項か明記する.
   新たな説明の場合, 新たな説明の作成をツールに要求する.
   説明が他の項目を指している場合, 項目間の関係の説明を付加することができる.
4. 項目/説明に任意に文字列によるラベルを付加する.

これらの作業は同期形態では義務ではないが, これらの作業の量が多ければ, 後述の非同期形態の作業の負担が軽減される.

利用者の行為に対して, ツールは会議を効率的に行なうための情報を利用者に提供する場合がある. 例えば,

1. 項目が削除の対象になった場合, 関連する説明/項目を利用者に提示する.
2. 既存の説明/項目に対しての言及の場合, 過去の言及の生の情報 (発言の音声/図示の図) などを提示する.

などである.

4.2.2 非同期形態

ツールの利用者は会議中に議論された内容の記録を基にその内容を整理し, 後の会議のための議事録や, 可能ならば最終生産物である仕様書を作成しなければならない. その作業は, 可能ならば会議と同時進行に行なえばよいが, 実際の作業ではそれは困難であるので, 会議外においてその作業を行なう. 非同期形態のツールは, 利用者のこのような作業を支援する.

この形態では第一に, 同期形態で成就できなかったこの作業を完全に行ない, それぞれの記録に情報を付加する.

次に, 生産物の作成を行なう. 4.1章の図 3で例示したように削除されていない項目と説明を箇条書きにすることで生産物である文書を作成する. 作業者は, それぞれの文書を段階別に作成し, それぞれの段階が出力すべき生産物として利用できる. 例えば, 計画段階に属する行為から作成された生産物は, 仕様書ではなく, 作業を進めるために必要な決定事項を記述した議事録として用いられる.

5 仕様作成支援のシナリオ

4章で, このツールがどのようなデータをどのような形態で蓄積するかについて述べた. この章では, ツールの利用者が有効な作業を成就するための戦略としてのツール利用のインターフェイスと利用手法について述べる.

5.1 インターフェイス

図 4にツール利用の大まかな流れを示す. ツール利用者は同期形態で基本的には通常の会議と同様に議論を行ない, 必要ならば計算機との情報のやりとりを行なう. 同期形態で蓄積された記録は非同期形態で必要な情報を添付することで整理し, 続く同期形態の作業に必要な仕様書や議事録を作成し, 可能ならば最終的な仕様書を作成する.

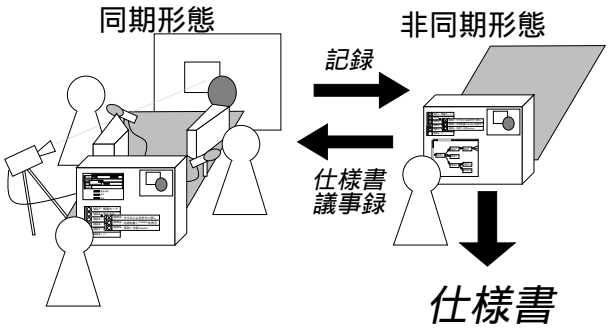

図 4: 大まかの作業の流れ

次に個々の作業者の利用するインターフェイスを紹介する. 初めに, 同期形態で利用されるインターフェースから紹介する.

- 段階リスト:
  図 5に作業者が行なった段階を図示するインターフェイスの例を示す. 2 次元の表中に段階が表示されている. 横軸が時間の経過を左から右に表し, 縦軸は段階を示している. 黒い矩形で表現されている段階を指定することで, その段階での項目/説明リスト (後述) を表示することができる.
- 行為リスト:
  図 6に作業者が行なった行為を図示するインターフェイスの例を示す. 2 次元の表中に作業者の行為が表示されている. 横軸が時間の経過を左から右に表し, 縦

図 5: 段階リスト

軸は役割を示している．この図では, 役割 2-役割 3-役割 4-役割 2 の順番に行為が行なわれ, 最後の役割 2 による行為は喋りながら図で説明をしていることを示している．個々の行為の内容を再生したり, その行為の言及している項目や説明や言及の仕方 (提案, 削除, 参照など) を表示することができる．この例では, 役割 3 の行為の詳細が表示されている．

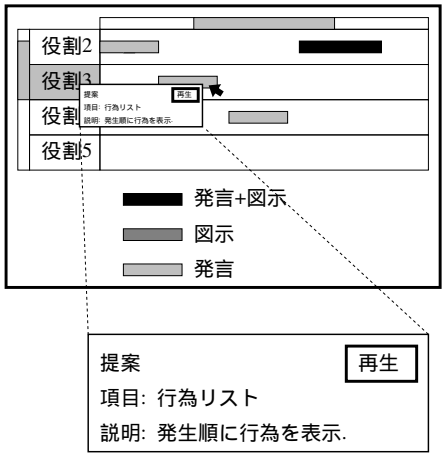

図 6: 行為リスト

- 項目/説明リスト:
  図 7に提案された項目/説明を表示するインターフェイスの例を示す．項目を表形式に表示し, ある項目を選ぶことでその項目の説明の表を表示できる．この例では, “項目 5:内容ボタン” に関する説明が 3 つ表示されている．それぞれの枠の左にある○ × は, その項目/説明が削除されているか否かを示している．また, それぞれの欄を指定することで, その決定が行なわれた行為の生データを再生することができる．

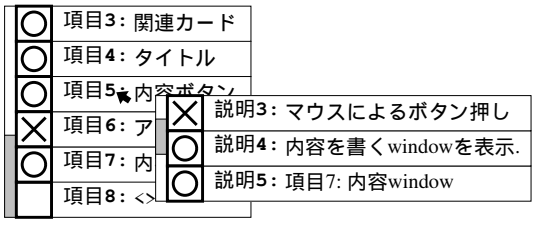

図 7: 項目/説明リスト

ある行為を行なう場合に, この項目/説明リスト上で項目/説明と行為の関係を保存することができる．

- 項目変化パターン表:
  話題となる項目の会議中における (出現順序を基にした) 変化のパターンを (一次元) 項目変化パターンと呼ぶことにする．例えば 100 個の項目出現していれば 100^2個の変化のパターンがある．この項目変化パターンを図 8のような項目変化パターン表に表示する．こ

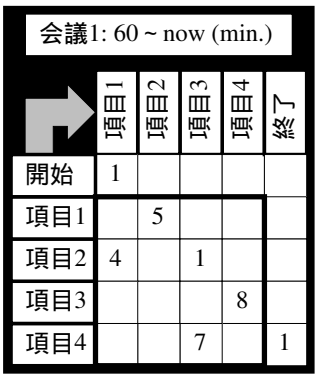

図 8: 項目変化パターン表

の例は, 会議 1 のある時点 (now) において, 開始から 60 分からその時点までの項目変化パターンを示している．項目 1 から項目 2 への話題の変化は, この時間範囲において 5 回あったことが示されており, この時間範囲の開始に議論された項目は “項目 1” であり, 最後に議論された項目, すなわち現在議論している項目は “項目 4” であることが分かる．また, 項目 1 と 2, 項目 3 と 4 が交互に変化していることが分かるので, 項目 1 と 2, 項目 3 と 4 は意味的に関係が深い可能性がある．

- ビデオ画面:
  黒板などの図示を計算機にとりこむ便宜のため, モニターを画面に表示する．実際に図示を行なっている作業者が, その図示という行為を計算機に記録するのは困難なので, それ以外の作業者が図示を行なっている作業者の行為として記録を行なう．
- 音声インターフェイス:
  発言の単位は自動的に計算機によって認識するのではなく, 利用者自身が区切りをつける．例えば, 「 発言中は計算機のマウスボタンを押す 」などのルールを決める．

次に非同期形態で使われるインターフェイスを紹介する．同期形態で利用したものは全て利用可能である．

- 項目/説明エディタ:
  項目/説明に関する文字によるラベルがない部分の補間を行なう．インターフェイスは図 7に近い表現となる．
- 生産物エディタ:
  項目/説明の蓄積情報より生産物である仕様書や議事録を作成する．図 9に生産物エディタの例を示す．図 7

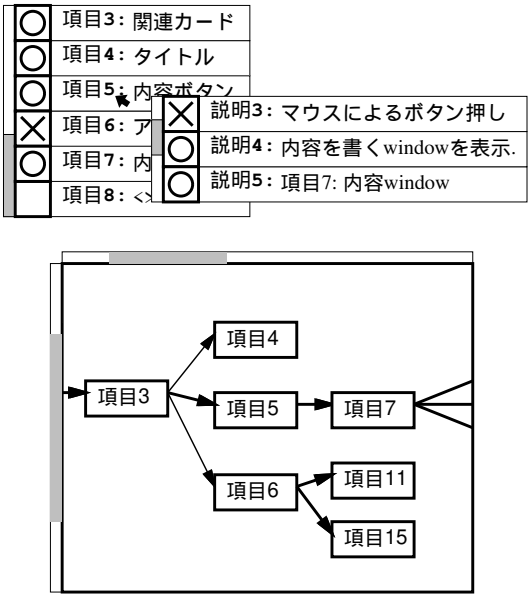

図 9: 生産物エディタ

の項目/説明リストに加えて, それらの関係を表示するブラウザを持つ. これを用いて利用者が生産物中における項目間の関係を決定してゆくことができる.

5.2 作業手法

実際にツールを利用して効率的に仕様書を作成するためにはいくつかの作業上の発見的な手法が必要である. この章では, いくつかの作業上の手法を紹介する.

- 行為リストの表示領域の制限をする. 個人の仕様作成作業の分析結果では後戻り作業はほとんど 7 つ程度前の塊までしか及ばないことが報告されている [12] . 共同作業でも同様の性質があると考えられ, その数に従って行為リストなどの表示領域を制限することで議論の発散を防止することが可能となる.
- 生産物エディタでブラウザ上の構造を作成する場合, 原則的には同期形態における明示的に関係を主張する行為に基づき関係を構成する. これに加えて非同期形態における作業者が項目間の関係を付け加えることを許す. この場合, 項目変化パターンを参考にする.
- 実際の会議の分析から, 作成対象とするソフトウェアが既知のシステムの場合, その項目変化パターンの数が少なく (すなわち, 全体の構造がわかっているので順序良く話しが進むということ), そうでない場合は, 多くの項目変化パターンが見られる (すなわち, 総当り的に話題を変えて可能性を探る) ことがわかっている. 項目変化パターン表では, この事実を表中の数値の埋まり具合で視覚的に表現している. 利用者は自分達の作業が整理された状態で行なわれているか否かの判断にこの表を利用することができる.
- 項目変化パターン表において, 全く変化を持たない項目間の関係を利用者にチェックさせることにより, 内容の矛盾や欠落を防ぐ支援を行なう. この関係の中には内容的に本当に関係のない関係もあるので, 可能性のあるものを非同期形態で洗い出す必要がある.
- 段階毎の生産物間の関係を調査して矛盾や欠落点を検出する. 例えば, このツールの段階リストを用いて発注段階の項目/説明を表示することにより, 発注段階で発注者から提案された項目をレビュー段階で全て調査し直すことが可能となる. また, 計画段階の作業分担を利用してシステムの構造などを決定することも同様に可能である.

6 今後の課題

我々は現在 UNIX を基盤とした網環境において, このツールのプロトタイプを試作中である. 今後, このプロトタイプを用いて, 我々の設計の性能と実作業への適用性の評価を行なう. 評価の基準としては,

- 議論された事項が十分に仕様書に反映されているか?
- 議論の繰り返し, 結論の変更, 結論のない議論などの, 能率的でない作業の割合がどのくらいあるか?
- 決定事項間に矛盾はないか?
- ある決定事項の変更に伴い必要となる修正が行なわれているか?

などが考えられる.

ツールで獲得した記録によって, 利用者にどのような有効な情報を提供すればよいかは, まだまだ不明な点が多いので, 過去の計算機を使わないソフトウェア作成会議の記録から必要とされる機能を分析/抽出する研究が必要とされている. 例えば, 段階毎の作業者の行為のパターンを収集し再利用することが考えられる. すなわち, 「 次の発言は顧客の立場で行なって下さい」のように続く発言内容を制約することで効率的な議論の展開を支援するのである. これについては, 実際の仕様化会議やツールの運用の記録を分析することで優良なパターンを蓄積する必要があるが, 永田ら [6, 10] の会議の文脈を追跡する手法が参考になると思われる.

また, ツール自身の拡張として, 会議で決定された仕様と, その後で作成される設計書, コード (プログラム), テスト項目 などを関係付けて管理する機能も考案中である.

謝辞

本研究を進めるにあたり, 富士通国際情報社会科学研究所での社会科学アプローチセミナーを通して貴重な議論及び助言を戴いた同研究所 ユーザー指向ソフトウェアプロセスグループのメンバーに深く感謝致します.

参 考 文 献

[1] Jeff Conklin and Michael L. Begeman. gIBIS: A hypertext tool for exploratory policy discussion. In *CSCW'86 Proceedings*, Dec. 1986.

[2] J. Lee. Extending the Potts and Bruns Model for Recording Design Rationale. In *13th International Conference on Software Engineering*, pp. 114–125, 1991.

[3] Linda Macaulay. Requirements capture as a cooperative activity. In *IEEE International Symposium On Requirements Engineering*, pp. 174–181. IEEE Computer Society, Jan. 1993.

[4] Gary M. Olson, Judith S. Olson, Mark R. Carter, and Marianne Storrosten. Small group design meetings: An analysis of collaboration. *Human-Compter Interaction*, Vol. 7, No. 3, pp. pp.347–374, 1992.

[5] C. Potts and G. Bruns. Recording the Reasons for Design Decisions. In *10th International Conference on Software Engineering*, pp. 418–427, 1988.

[6] 永田守男, 高木晴夫, 竹内昇一. コミュニケーション文脈の追跡システム. 日本経営情報学会誌, Vol. 1, No. 1, pp. 39–44, Jan. 1991.

[7] 海谷治彦, 佐伯元司. プロダクトを基にしたソフトウェア設計者会議の分析法. 情報処理学会ヒューマンインターフェース研究会, pp. 47–16, Mar. 1993.

[8] 中谷三江. 劇場モデルに基づいたソフトウェア意図伝達支援ツール COMICS. 情報処理学会論文誌, Vol. 31, No. 1, pp. 124–135, Jan. 1990.

[9] 西, 海谷, 佐伯. ソフトウェアの発注者-開発者会議におけるインタラクションの分析. 情報処理学会ヒューマンインターフェイス研究会, Mar. 1992.

[10] 竹内晃一, 永田守男, 植竹朋文, 高木晴夫. コミュニケーションの分析を通した会議の支援システムの構築 (1)(2). 情報処理学会第 45 回全国大会講演論文集 (6), pp. 251–254. 情報処理学会, Oct. 1992.

[11] 中島毅, 田村直樹, 藤岡卓, 上原憲二, 高野彰. PPK 法 : ソフトウェア設計プロセスの記録と分析の手法. ソフトウェア工学研究会, Vol. 67, No. 2, Jul. 1989.

[12] 渡辺, 海谷, 佐伯. 仕様記述作業のモニタリングツールとその記録分析. 情報処理学会 ソフトウェア工学研究会, Jul. 1991.

[13] 尾上裕子, 桑名栄二. 設計者間のコミュニケーション構造モデルの一考察. 情報処理学会グループウェア工学研究会, Vol. 3, No. 1, Dec. 1992.

[14] 落水ほか. ソフトウェア開発における協調支援環境 Vela. 情報処理学会第 41 回全国大会 5, pp. 149–162, Sep. 1990.