ソフトウェアの仕様化過程における協調作業に関する研究

東京工業大学 理工学研究科
電気電子工学専攻 知識システム講座
指導教官: 佐伯元司 助教授

M9371 海谷 治彦

１９９１年2月

目 次

目次 **2**

1 はじめに **3**

2 目的と方法 **5**

2.1 目的 5

2.2 実験的手法 6

2.2.1 実験計画 7

2.2.2 実験環境 8

3 実験記録の整理 **9**

3.1 整理方法 9

3.2 実験 1 , 2 の共通点 9

3.3 実験 1 10

3.3.1 実験内容 10

3.3.2 作業者の役割による分類 12

3.4 実験 2 13

3.4.1 実験内容 13

3.4.2 作業者の役割による分類 17

4 実験データの分析と協調作業のモデル **27**

4.1 実験 1 27

4.2 実験 2 28

4.3 分析法と協調作業モデル 30

4.3.1 分析法 30

4.3.2 協調作業モデル 34

5 議論 **37**

5.1 モデルの事例としての実験 37

5.1.1 実験 1 37

5.1.2 実験 2 40

5.2 分析法の評価 40

5.3 協調作業モデルの評価と改良 41
5.4 協調作業の支援について 41

6 おわりに **45**

謝辞 **47**

参考文献 **50**

第 1 章

はじめに

作成対象となるソフトウェアが複雑になるにつれ，複数の人間が仕様化 / 設計作業に携わるようになった現状では，協調作業としての人間の活動を明らかにする必要がある．我々は実際の仕様化過程の分析の方法を提案することで，人間の協調作業を明らかにする足掛かりとする．

近年，ソフトウェア開発も含め，グループによる知的生産活動を支援する技術，グループウェアに関する研究が盛んに行なわれている．

グループウェアを従来の支援技術と区別する重要な点として，Winograd は人間の協調構造に基づいて設計されていることを上げている．

グループウェアの設計においてはグループ作業の構造を的確にとらえ，システムに反映させる必要がある．そのために作業のモデル化技術が重要な役割を演じることとなる．

例えば，Wingrad らの The Coodinator では，言語行為理論に基づく Conversation Model を開発し，その状態遷移に基づいて会話の進行管理を行なっている [6] [7] [8] [9] ．しかし，The Coodinator はシステムで規定されている会話規約に従うための作業の煩わしさが指摘されている．さらにこのシステムは一般的な用途に対して作られているので，ソフトウェアの仕様化過程の協調作業を支援するには不十分な点がある．

Marca らは，グループウェアは実際の協調作業に再利用できることが重要であると主張している．彼らは，初期のグループウェアである Monster や XCP のように作業の相互関係を定義することにより作業をモデル化したものは，作業過程の少しの変化にも対応できないことを紹介し，作業における時間に依存しない規則を定義することが再利用可能なモデルを構築する鍵となると述べている [4]．

また協調作業のモデルを基にした協調作業支援系もいくつか提案されている．

Marca らは，自分達の提案した考えを基に，Winograd の考えを利用した CONTRACT というシステムを構築しているが，彼ら自身の提案していた構築の鍵を生かしているようには思えない．

Conklin らの gIBIS では IBIS(Issue Based Information System) モデルと呼ぶ議論の状態遷移モデルに基づいて設計討論用のハイパーテキストシステムを設計している [1]．

ソフトウェア開発の分野では，岸本 [11] [10] [13] らが，劇場における脚本家から観客への意思伝達方法をヒントにした劇場モデルという意思伝達のモデルを構築し，システム設計者からプログラマへの意思伝達を支援するツール，COMICS を実現した．彼女らのモデルの着想は大変良い点をついているが，システムへの実装の時点でそのモデルが反映されいてるとはおもえない．

落水 [16] らは，初心者から熟練者への技術移転の促進，作業者間の意思伝達，合意形成，情

報交換の促進などの協調支援機能の実現を目標としたプロトタイプ Vela の開発を進めている．

Bill Curtis[2] は，従来のソフトウェアプロセスのモデル化がライフサイクルモデルや個人レベルの活動や理想化されたグループ作業にかたよっていることを批判し，グループ作業の効率を改善するソフトウェア開発過程のモデルが必要であることを主張している．彼は，従来の協調作業を支援する電子会議室 ([5] など) などがグループ作業の効率を改善することができず中止になったことを紹介し，その原因は作業者が効果的に振舞えるような支援をしなかったことにあると主張している．また彼は，大型のシステム開発においての根本的な問題として，グループ上の要求や設計の共通理解の構築をあげ，異なった分野の専門家が一緒になって知識の統合をする過程や，対象とするソフトウェアの共通モデルを作り上げる必要があるコミュニケーション過程を調査することが重要であると主張している．

本研究では，ソフトウェア開発における複数の作業者の仕様化作業を分析するための枠組み— 分析方法 — を提案し，それをもとに協調作業モデルを構築した [12] ．

人間の協調作業の構造は，形式化された数学モデルを出発点としては，Curtis らの批判するような実際のグループ作業の効率を高めることのできない構造となる危険が大きい．我々は実際の協調作業を分析することで協調作業の構造を明らかにする方法をとった．

そのために重要なのは，実作業を分析するための指針— 分析方法 — を構築することである．分析方法は現在のところまだ定説が存在しないため，作業の分析を行ないながら分析方法を構築する方法をとった．

我々はまず始めに，ビデオカメラを用いて実際のグループ作業の様子を記録し，それを繰り返し観察することにより分析のための指針を少しずつ決定していった．

分析の中で得た指針を整理することにより，分析方法を構築し，その分析方法を用いて，記録したグループ作業の整理を行ない，ソフトウェアの仕様化作業における協調作業の構造のモデル化を試みた．

本論文の構成は以下のようになっている．

2章では，研究目的と研究作業のシナリオについて述べる．

3章では，実験記録から実験データを抽出する過程を示す．

4章では，3章の実験データの分析を行ない，その分析結果に基づいた分析方法を構築する．そして，その方法を反映させた協調作業モデルを提案する．

5章では，分析方法と作業モデルの評価を行ない，それぞれのグループの作業をモデルの事例として整理，評価を行なう．さらに，ソフトウェアの仕様化作業における協調作業構造について考察を行なう．

最後に 6章で，まとめについて述べる．

第 2 章

目的と方法

我々の最終目的の 1 つはソフトウェア開発における人間の協調作業のモデル化技術を確立し，協調の仕組みを明示的に組み込んだソフトウェア開発環境を構築することである．

本章では，本研究が最終目的の中でどのような位置を占めているかを説明し，その目的を達成するための我々のとったアプローチ— 実験的手法 — について述べる．

2.1 目的

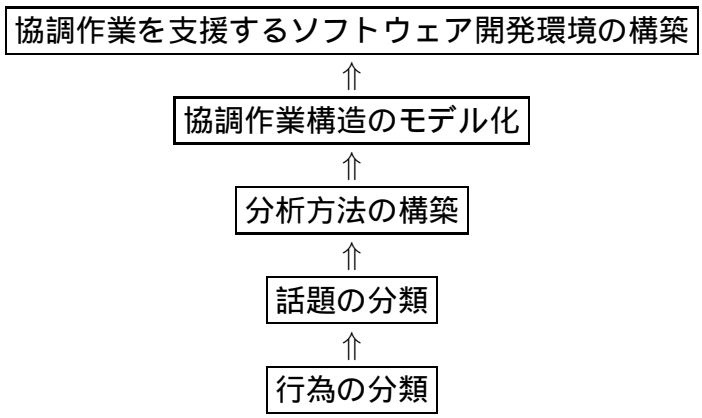


図 2.1: 研究目的

図 2.1 に本研究の流れを示す．

協調作業を支援するソフトウェア開発環境の構築をするためには，協調作業構造のモデル化を行なうことが必要である．

我々は協調作業構造のモデルを特定の数学的構造を出発点とせず，実際のグループ作業のプロトコル解析を行ない，それの分析を行なうことによって協調作業モデルを段階的に構築する方法— 実験的手法 — を選択した．これは，1章で述べた Curtis[2] の示唆を考慮している．

しかし，複数の作業者の分析の指針が明らかでないため，実際の仕様化作業の観察を行なうことによって指針を決める方法をとる．

定義 1 (分析方法) 仕様化作業を分析するための指針を 分析方法 と呼ぶ．

分析方法を構築する過程においていくつかの 仮説 を定め，いくつかの 結論 を導くことにより研究を進める．

ここで第 1 の仮説をおく．

仮説 1 (行為) ソフトウェアの仕様化過程における協調作業の構造において大きな役割を果たしているのは，作業者の 行為 である．

ただし，

定義 2 (行為) 作業者が行なう他の作業者に影響を与えてる発言や，行動を行為とする．

この 仮説 1 を基に，実験計画を立てた．

話題とは，実験の分析をした結果として考え出した行為を束ねる概念であり，次の章で説明する．

2.2 実験的手法

我々は実験を行なって，その結果から協調作業をモデル化するための基準が発見できると考えている．我々が最初に持っている仮説は 行為 が協調作業において重要であることだけである．

そこで，実験的手法の研究計画を 図 2.2 のように立てた．

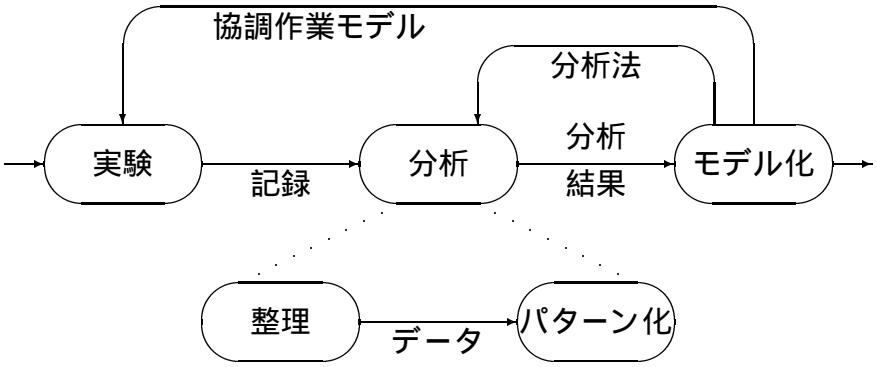


図 2.2: 研究計画の流れ

楕円は作業を示し，矢印は作業によって得られるプロダクトの流れを示している (実験) によって得られた 記録 に対して (分析) を行なう．

分析は具体的には，記録のなかで意味のある物を (整理) することにより データ を作成する段階と，データの特徴を考察するすることによる (パターン化) し 分析結果 にまとめる段階とに分かれる．

分析結果を用いて，協調作業のモデル と，分析のための指針である 分析方法 をまとめる．具体的には，顕著に見られたパターンを整理して記述することで，協調作業モデルを構築し，

そのパターンを分析の重点項目とすることで分析方法を決定していくのである．これが（モデル化）の作業である．

分析方法 を（分析）に，協調作業のモデル を（実験）に帰還させることで，それぞれを精錬することが期待できる．具体的には，協調作業モデルのある特徴を際だたせるような実験設定を行なうことにより，そのモデルの実作業での有効性を計ることができる．そして，実験結果を分析する時点において，分析方法に従って分析を行なうことで，効率的な分析が期待できる．

2.2.1 実験計画

図 2.3 に具体的に今回行なった実験の手順図を示す．図中の四角が研究の作業の 1 単位であり，矢印がそれぞれの作業の依存関係を示している．

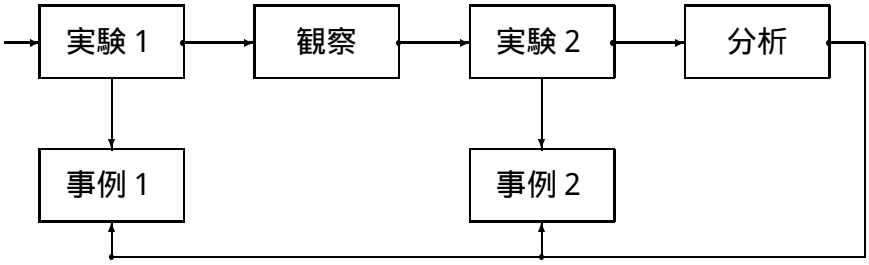


図 2.3: 実験の手順

それぞれの作業段階について説明する．

実験 1: 予備実験

複数の作業者に対して実際のソフトウェアの仕様化作業を行なってもらい，その作業過程をビデオテープに記録する．

観察

実験 1 の実験結果の観察を行ない，実験記録に対するある程度の分析基準を与える．

実験 2

観察 で得られた分析基準を考慮した制約を作業者に加えて，実験 1 同様に実験を行なう．

分析

実験 1 ，2 に対する実験結果を整理することにより，協調作業のモデル化のための指針 — 分析方法 — と，協調作業のモデルをまとめる．

事例 1, 事例 2

作業モデルを用いて実験 1 ，2 の実験記録を実際の協調作業の事例として記述する．

2.2.2 実験環境

我々が複数の作業者にたいして仕様化作業の実験を行なうにあたり，以下のような点が重要であると考えた．

- 作業者の作業に対して，できるだけ束縛を与えない．
- 実験の記録をすることが作業者に対して影響をあたえない．

以上のような点を留意して，実験１，２は以下のような環境で作業を行なわせた．

作業形式

会議室を使って討論形式で作業を行なう．作業者は 1 つのテーブルを囲んで，お互い対面する形で自由な討論が可能である．

さらにそれぞれの作業者は，メモ，黒板の利用が可能である．これによって作業者は個人の作業のための領域と，作業者全体に伝えるための領域を確保することが可能である．

実験の記録方法

１台のビデオカメラを使い，実験中の作業者を撮影することで記録を作った．特に，作業中に発言や黒板やメモなどに何かを書くというような行為をした作業者をできる限り追跡するような撮影をおこなった．

これによって，作業者は自分の必要な作業以外の作業，例えば，分析を行なうための考えたことをメモをしなければならないなどの制約を作業者は負う必要がなくなる．

第 3 章

実験記録の整理

本章では ,どのような実験記録が残り , その記録をどのようなデータに加工したかを紹介する .

3.1 整理方法

実験 1 , 2 において , 作業を録画したビデオテープと作業者の記述した仕様書 , およびメモが記録として得られた .

仮説 1 より , 実験記録から行為を取り出すことにより整理を行なう .

具体的には ,

1. 記録より , 行為を行なった作業者とその行為を時系列順に列挙する .
2. 行為を 適当な基準 を用いて分類する .
3. 行為間の依存関係を整理する .

のような作業を進める .

我々は適当な基準として ,

- 時系列的基準: 作業者の行為を時間軸にそって整理する .
- 内容的基準: 作業者の個々の行為の内容にそって整理する .

の 2 つの基準を使って記録を整理する .

初めにもっとも客観的なデータとして , 作業過程においての行為を , その行為がどの作業者が行なったかで分類し , 時系列にそってグラフ表現する .

次に , 作業者の行為を作業者に与えた役割を指標として分類し , 同様にグラフ表現する .

定義 3 (役割) 作業者の行為を束縛する作業上の性質を役割とする .

3.2 実験 1 , 2 の共通点

2 つの実験で共通している結果を示す . これは , 2.2.2 で述べた実験環境を留意し , 利用できる資源を検討した上で決定した .

作業者

本学，情報工学科４年生の学生５ ~ ６人

問題の与え方

ソフトウェアを発注する顧客にあたる作業者を 1 人参加させている．他の作業者にたいして問題に対する事前の知識は与えない．

問題の性質

対話型のソフトウェア．既存のソフトウェアが存在し，ワークステーション（ＷＳ），パーソナルコンピュータ（ＰＣ）上で稼働している．顧客はそれに対しなんらかの不満を抱いている．

3.3 実験 1

実験 1 において以下のような仮説をおく．

仮説 **2** ソフトウェアの仕様化過程において作業分担は重要な要素である．

この仮定に基づき，実験 1 ではそれぞれの作業者がプログラミングを担当するモジュールの分担を行なうように作業者に対して指示をしたうえで実験を行なった．

仮説 **3** 実験 1 での役割は作業分担である．

3.3.1 実験内容

以下に，作業に対する入力，作業過程，出力を示す．

作業に対する入力

問題 グラフィックエディタ
論文などの図を書くツール

制約 5 人のプログラマに作業を割り当てるための，作業分割を行なうように指示．

作業過程

会議は合計 3 回行なうことになり，それぞれの日時と時間は以下の通りである．

第 1 回 5 月 15 日 2 時間 12 分
第 2 回 6 月 6 日 1 時間 15 分
第 3 回 6 月 12 日 1 時間 16 分

作業後の出力

8 ページ程度の自然言語 + 図で書かれた仕様書．図 3.1 に実際の仕様書に例を示す．

作業はモジュール分割の段階（第 1 回）と，モジュール間の調整の段階（第 2 ，3 回）に別れた．

描画関係
基本機能
直線関係：直線，長方形，平行四辺形，連続直線（多角形を含む）
曲線関係：円（円弧等を含む），自由曲線，ひっかき線
その他：塗りつぶし，点
各機能の入力（[ ]内は指定がなければデフォルト値を使用）
1. 直線… 始点，終点，[線種，線サイズ]
2. 長方形… 始点，終点（対角線上にとる），[線種，線サイズ]
3. 平行四辺形… 始点，中間点，終点，[ ]
4. 連続直線… 始点，中間点，…，終点（ダブルクリック）[ ]
5. 円 …・中心，半径，[始点，終点（円弧のみ），線種，線サイズ]
・円周上の3点，[ ]
・直径の2端点，[ ]
いずれかの方法で入力
6 自由曲線…（連続直線と同様）
7. ひっかき線… 始点，ひっかき線の方法，終点，[線種，線サイズ]
8. 塗りつぶし… 点指定（その点を含む領域を塗りつぶす）[塗りつぶしパターン]
9. 点…… 点指定 [点の大きさ]
始点，終点を指定
直線に対して，
どちらの方向に
マウスを動かしたか
checkする
または
内部の処理
・各機能の種類と，それに必要なデータをメモリにストアし，描画を行う。

図 3.1: 実験 1 の仕様書

3.3.2 作業者の役割による分類

実際に役割分担名とその内容と役割を与えられた作業者の対応を 表 3.1 に示す．

役割	内容	作業者
入出力	ファイルやプリンタなどとのやりとりの管理をする．	作業者 1
画面	メニューレイアウトや画面表示をする．	作業者 2
文字	文字入力をする．	作業者 3
編集	図形の編集をする．	作業者 4
描画	特定の図形を描く．	作業者 5

表 3.1: モジュールと作業者

3 回の実験から，作業者を分類基準としたデータと役割を分類基準としたデータを作成した．表 3.2 に実験 1 のデータの対応を示す．

図 3.2 ，図 3.4 ，図 3.6 は，縦軸に上から下へ作業者の行為の順番を時系列にとり，横軸は実際の行為を行なった作業者をとっている．横軸の作業者 1 は入出力，作業者 2 画面，作業者 3 は文字，作業者 4 は編集，作業者 5 は描画の担当になっている．

図 3.3 ，図 3.5 ，図 3.7 は第 1 ，2 ，3 回それぞれにおいて，作業分担に対応した行為を時系列的に表現したものである．縦軸が上から下に作業者の行為の順番を時系列的に表しており，横軸がそれぞれの行為が言及している役割分担を表している．

また，顧客に相当する作業者が会議の進行の調整も兼ねていることが分かり，以下の結果が得られ，それは実験 2 における仮説のヒントとした．

結果 **1** (調整者) 仕様化過程においては調整者が大きな役割を果たしている．

図 3.3 では，描画 と 編集 に言及する行為が多く，その実際の内容は，どのような 描画 ができて，それにたいしてどのような 編集 がしたいかの羅列であった．

	作業者	役割
第 1 回	図 3.2	図 3.3
第 2 回	図 3.4	図 3.5
第 3 回	図 3.6	図 3.7

表 3.2: 分類によるデータの対応

図 3.5 からは，画面，編集，文字，入出力，画面 の順番で作業者の行為の言及しているモジュールは移っている．

仮説 2 と 表 3.2 のデータの対応より，仕様化過程をソフトウェアの作業分担を基準にした，ある程度まとまった行為の列を用いて整理することが考えられる．

定義 **4** (話題) ソフトウェアの特定の作業分担について言及している行為の列を 話題 とする．

仮説 **4** (話題) 話題の列を仕様化過程とする．

第 2 回以降の行為は主に 画面 を中心に言及が行なわれている．このソフトウェアにおいては，利用者界面である画面が各モジュールの接点となっており，そのために画面に言及する行為が多いと考えられる．

3.4 実験 2

実験 2 の作業者に対しては，表 3.3 のような作業に関わる立場の違いによって役割を与えた．

役割	内容
顧客，利用者	ソフトウェアに対する要求を意識する行為を代表
製作者	ソフトウェア製作を意識する行為を代表
調整者，書記	作業の調整を意識する行為を代表

表 3.3: 実験 2 における役割

定義 5 (立場) 作業過程において，異なった作業段階の意見を支持することにより分類される役割を 立場 とする．

3.4.1 実験内容

以下に，作業者に対する入力，作業過程，出力を示す．

作業に対する入力

問題: ハイパーテキスト型のツール
ソフトウェアを仕様化する人間の思考過程の履歴をとるためのツール [15]

制約: 以下に示すような役割を与える．

顧客 ..... 要求を提案する立場．
利用者 ..... ソフトウェアを利用する立場．
製作者 ..... 技術的な助言をする立場．
書記 ..... 記録をとり，仕様書を書く立場．
調整者 ..... 作業進行を調整する立場．

プロダクトとして，

1. 基本調査書
   このソフトウェアがどんなものかを明確にする．
2. 要求仕様書
   ソフトウェアが何をすれば良いのかを明らかにする．
3. 設計仕様書
   ソフトウェアをどのように実現すれば良いのかを明らかにする．

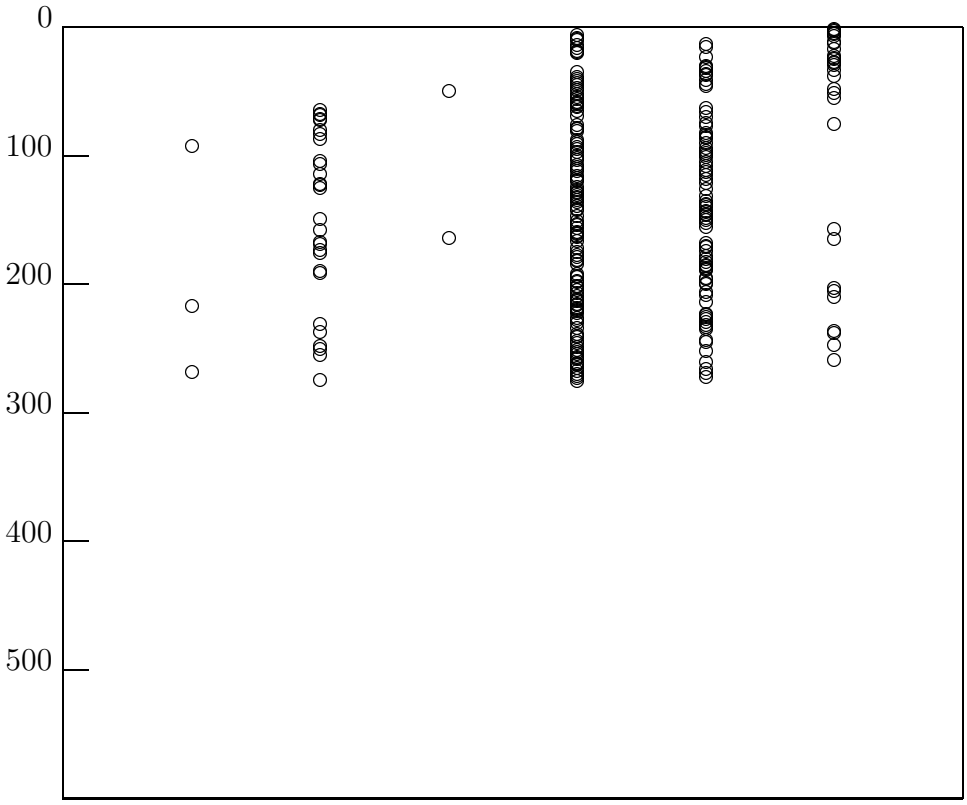


図 3.2: 実験 1 第 1 回の作業者別の行為の推移 (289 行為)

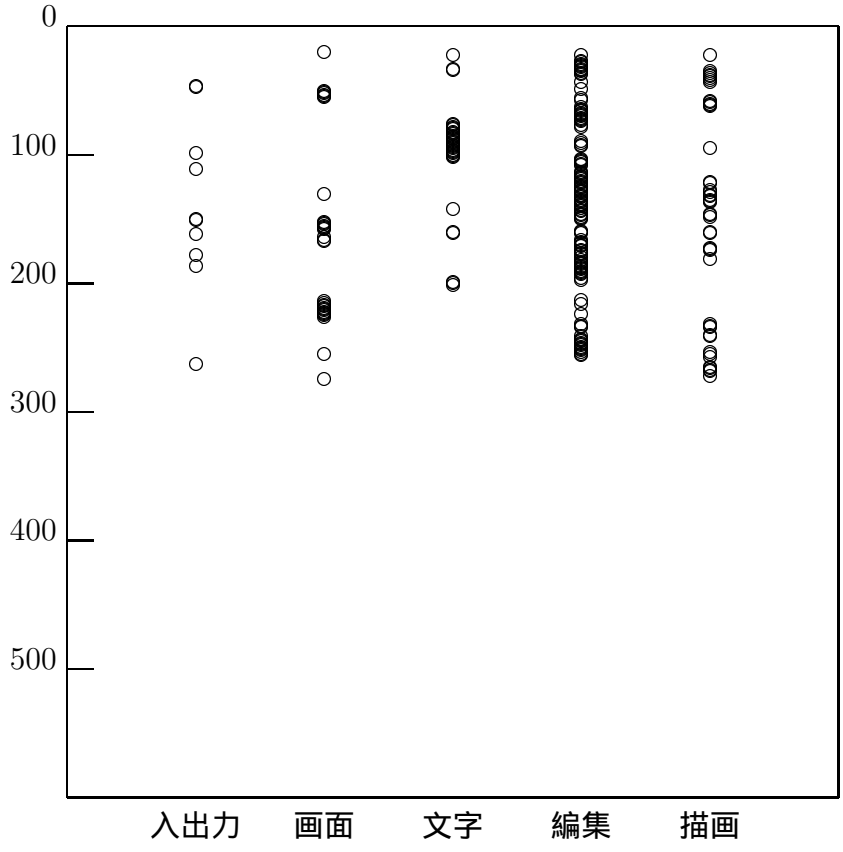


図 3.3: 実験 1 第 1 回の行為の推移 (289 行為)

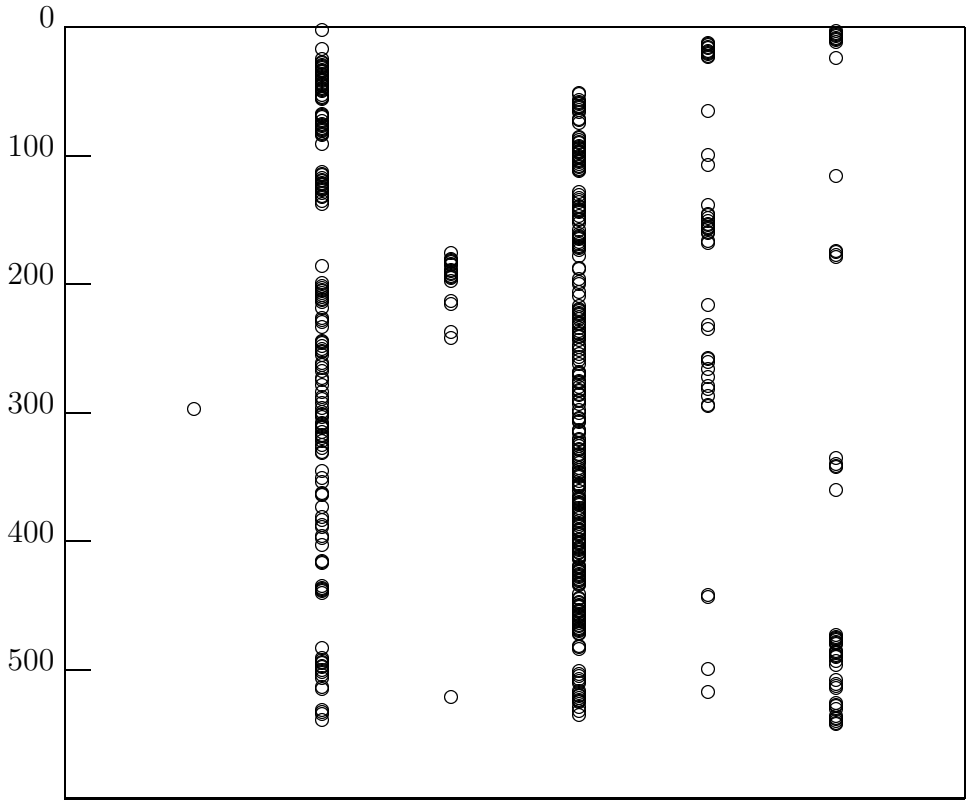


図 3.4: 実験 1 第 2 回の作業者別の行為の推移 (548 行為)

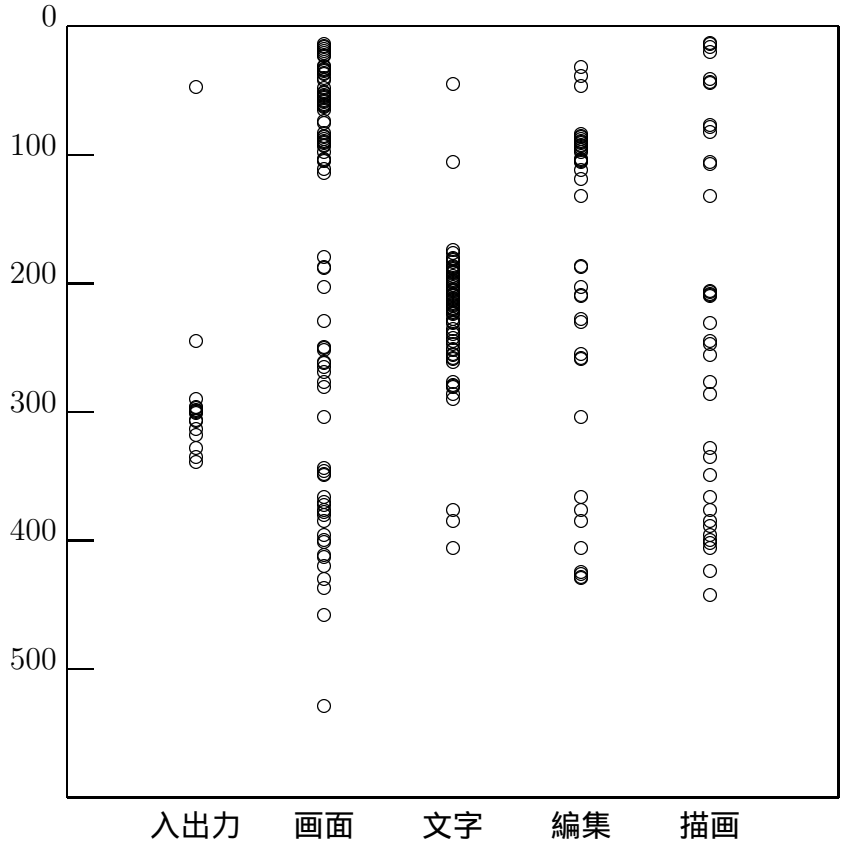


図 3.5: 実験 1 第 2 回の行為の推移 (548 行為)

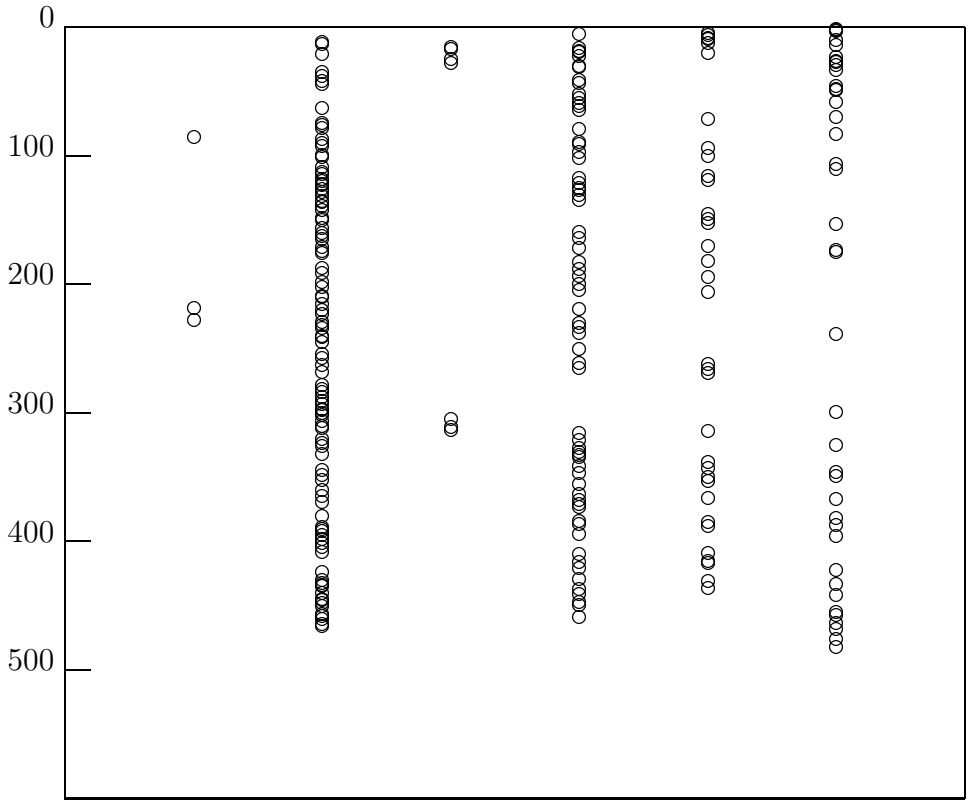


図 3.6: 実験 1 第 3 回の作業者別の行為の推移 (490 行為)

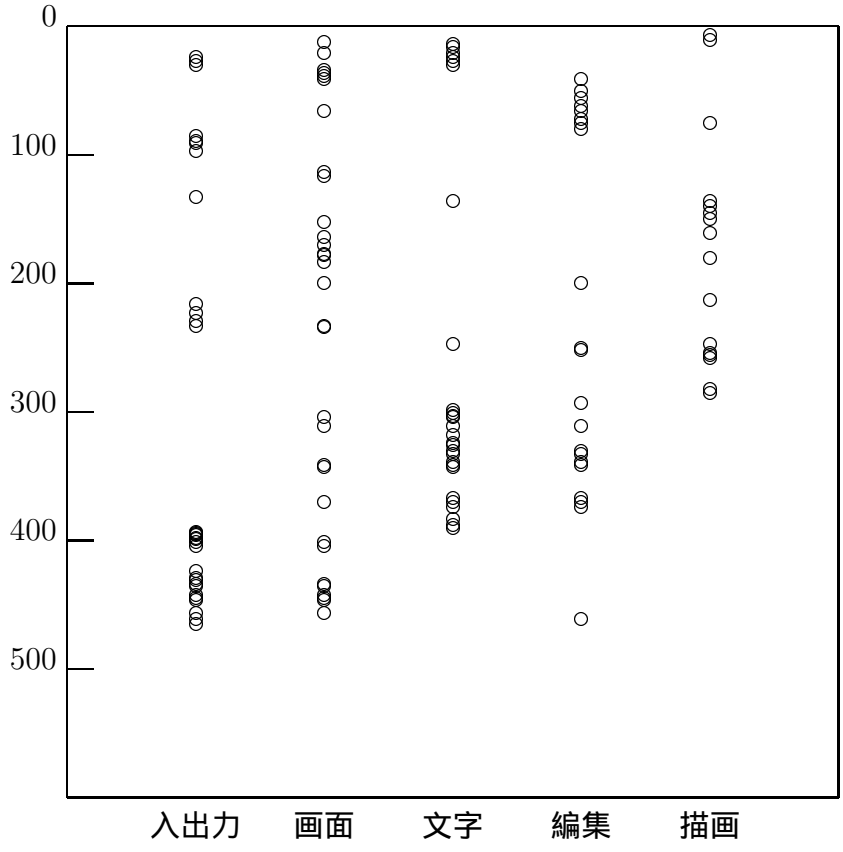


図 3.7: 実験 1 第 3 回の行為の推移 (490 行為)

の作成を文書で指示．

作業過程

会議は合計 4 回行なうこととなり，それぞれの日時と時間は以下の通りである．

第 1 回 7 月 10 日 2 時間 45 分 基本調査
第 2 回 7 月 11 日 2 時間 27 分 要求仕様化
第 3 回 7 月 17 日 2 時間 54 分 設計仕様化
第 4 回 7 月 18 日 3 時間 30 分 設計仕様化

作業後の出力

1. 要求調査書: 自然言語 + 図 2 ページ
2. 要求仕様書: 自然言語 + 図 2 ページ
3. 設計仕様書: 自然言語 + 図 １２ページ

実際の仕様書の例を図 3.8，図 3.9，図 3.10に示す．

実験 2 の問題は実験 1 の問題のような，一般的に良く知られてるソフトウェアではなく，ある特殊な用途に使うソフトウェアである．

よって，説明のため顧客に対して事前にツール提出させた仕様書を 図 3.11 に示す．このソフトウェアを使っての研究は [15] で発表されている．

3.4.2 作業者の役割による分類

作業者	内容	役割
作業者 1	要求を出す立場．	顧客
作業者 2	ソフトウェアを利用する立場	利用者
作業者 3	ソフトウェアを製作する立場	技術者
作業者 4	作業過程を調整する立場	調整者
作業者 5	仕様を記録する立場．	書記

表 3.4: 立場と作業者

表 3.4 は作業者とそれぞれの作業者に割り当てた立場を対比したものである．4 回の実験から，行為を作業者を基準として分類したデータと，立場を基準としたデータを作成した．

表 3.5 は実験データの対応を示している．

立場に対しては時系列的な特徴が見られないので，次章において，内容的基準を基にした考察を行なう．

システムについて考えたとき、

仕様(要求の羅列…自然言語)を書くために、

考え方の履歴を記録するツールを作る。

→ 順番(グラフや木で表わす)。

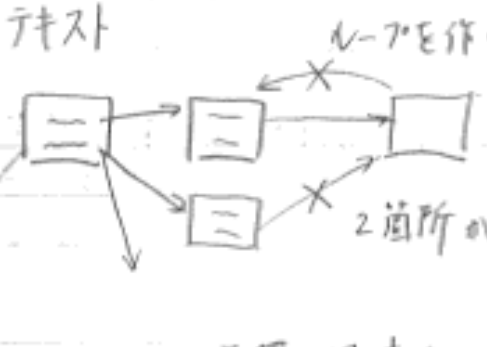
・ビデオで間に合うか: 今までにやっておけ。人間の思考パターンをしぼってみたい。

・手引書で済むか: ハイパーテキスト的なものが欲しい。現在のものは使いにくい。紙でも済むが、一種の横着。

(・実験の目的…どのように考えるのが適当かということではなく、ハイパーテキストの妥当性を考える。)

現行のシステム

・ハイパー・テキスト



1枚1枚のカードには、表題、関連するオプション、関連するアクション、内容が書いてある。

図 3.8: 要求調査書の例

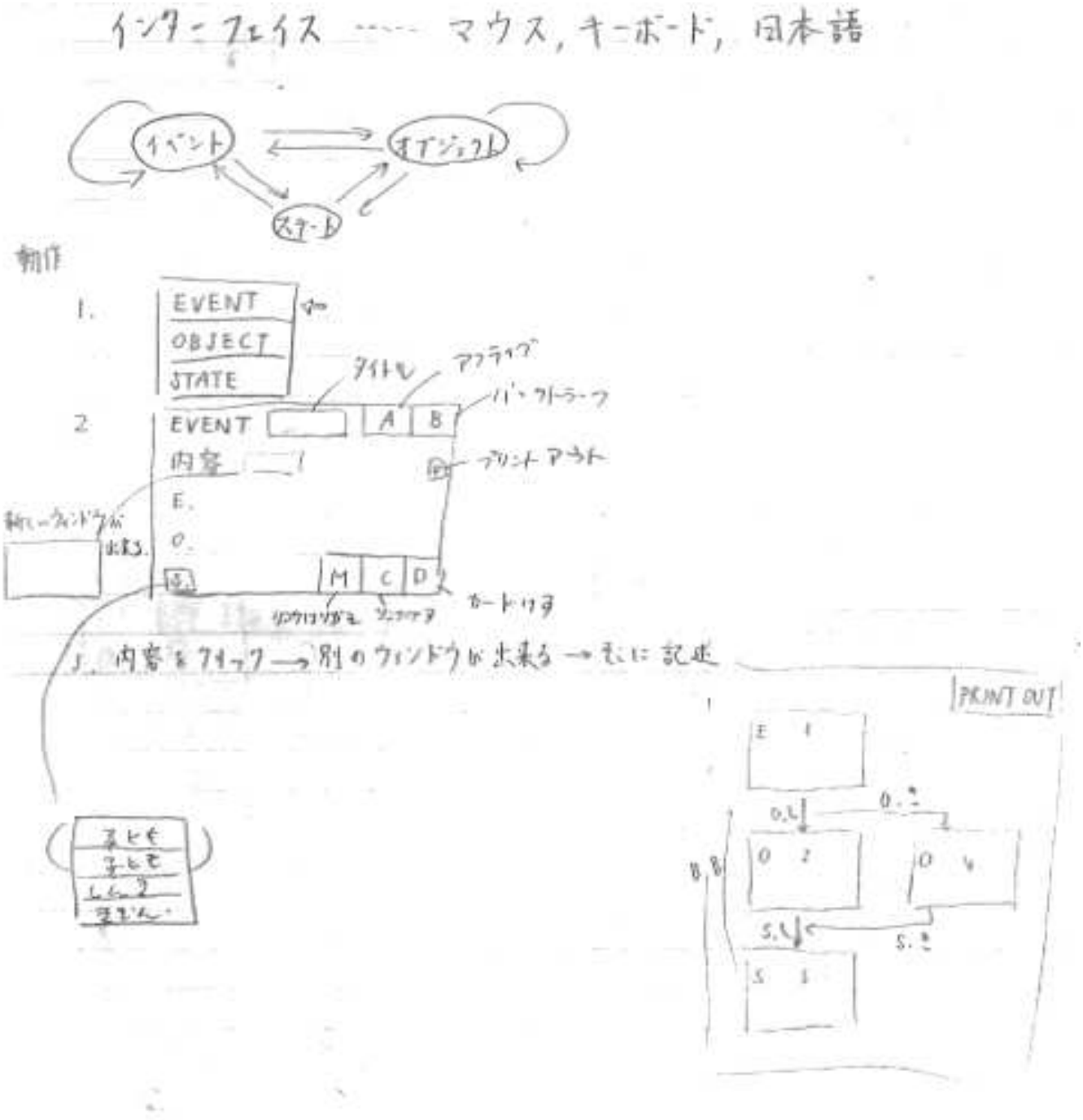

インターフェイス …… マウス, キーボード, 日本語
イベント
オブジェクト
ステート
制作
1.
EVENT
OBJECT
STATE
2
EVENT
内容
A
B
M
C
D
PRINT OUT

図 3.9: 要求仕様書の例

図 3.10: 設計仕様書の例

- ハイパーテキストの構造を持つ
- システムのプロダクトの各々が 1 つのノードに相当する
- ノードはプロダクトのタイトルを記録する場所を持つ
- ノードはプロダクトの内容を記録する場所を持つ
- ノードは次のノードを生成するためのボタン c1, c2, c3 を持ち、ノード間のリンクの情報は保持される
  - c1：関連するイベント
  - c2：関連するオブジェクト
  - c3：関連するステート
- ノードは親ノードを呼び出すボタンを持つ

図 3.11: 作業前のソフトウェアの仕様

	作業者	立場
実験 1	図 3.12	図 3.13
実験 2	図 3.14	図 3.15
実験 3	図 3.16	図 3.17
実験 4	図 3.18	図 3.17

表 3.5: 分類によるデータの対応

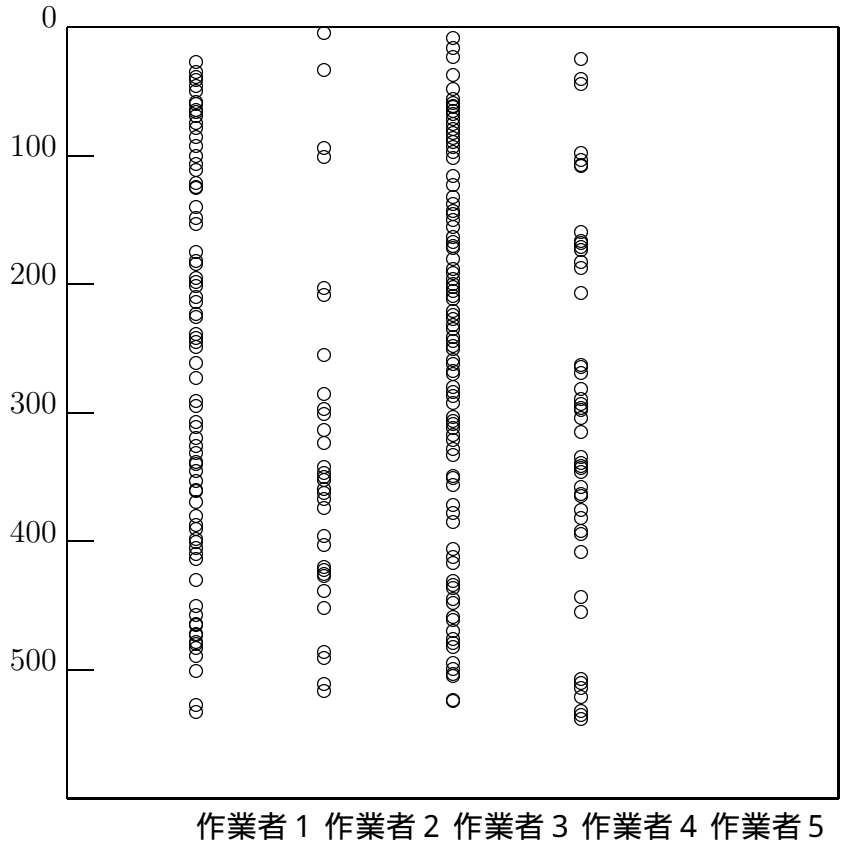


図 3.12: 実験 2 第 1 回の作業者別の行為の推移 (539 行為)

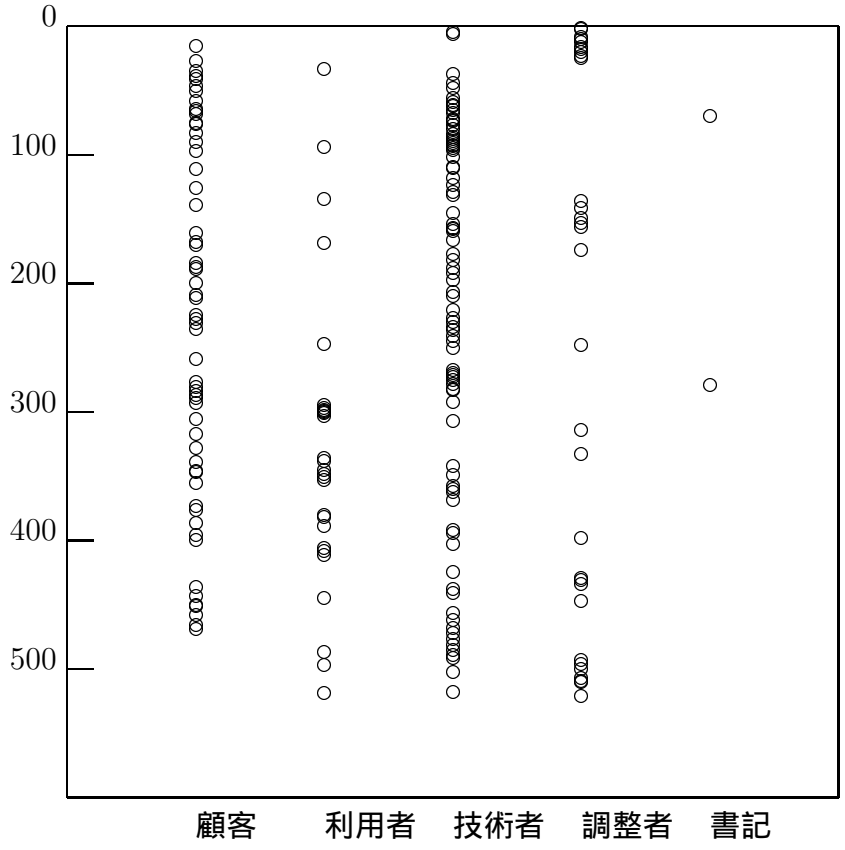


図 3.13: 実験 2 第 1 回の立場別の行為の推移 (539 行為)

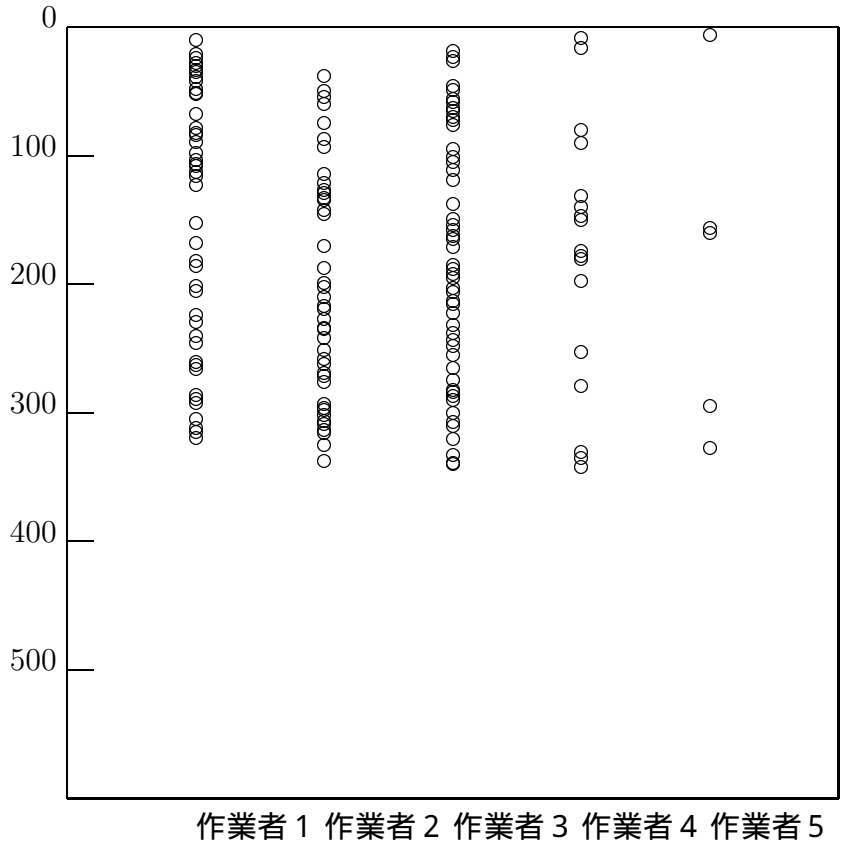


図 3.14: 実験 2 第 2 回の作業者別の行為の推移 (344 行為)

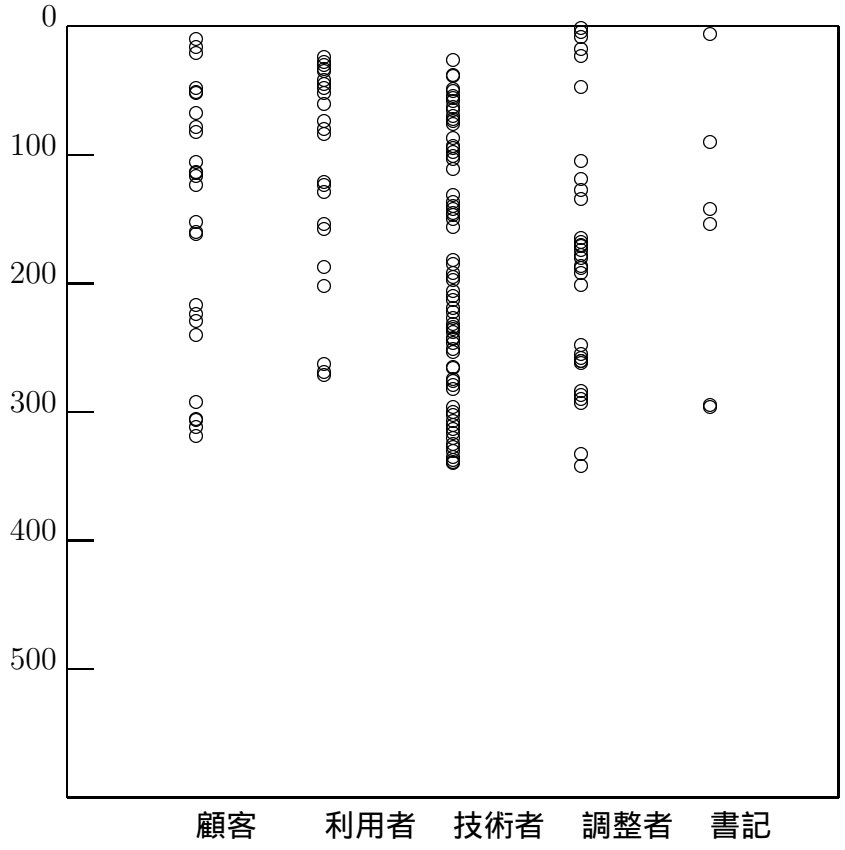


図 3.15: 実験 2 第 2 回の立場別の行為の推移 (344 行為)

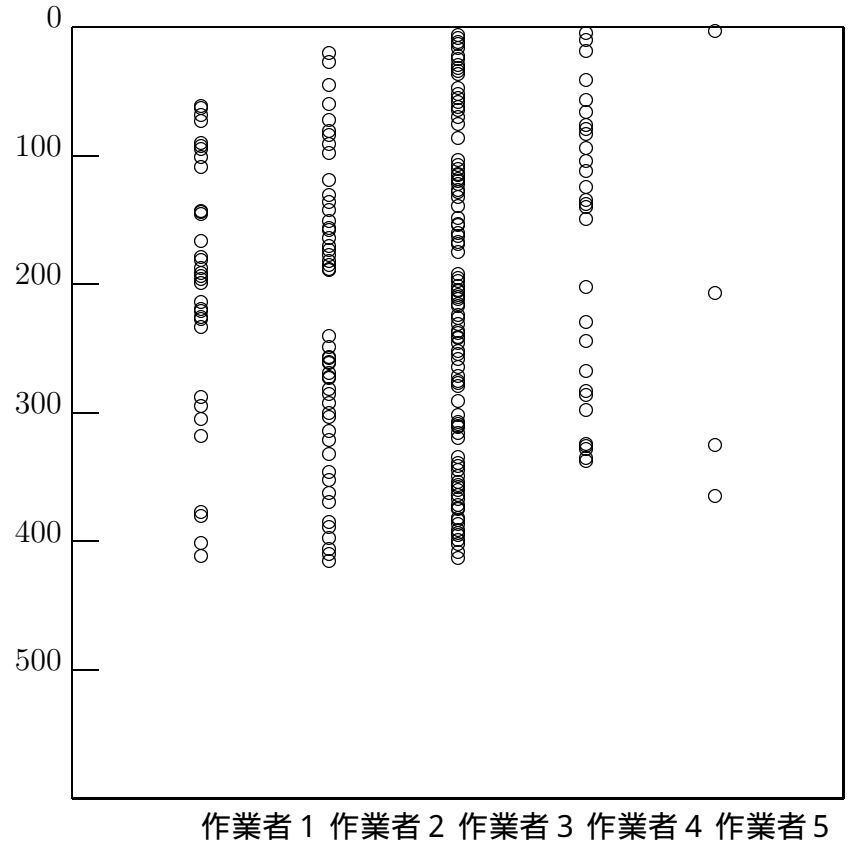


図 3.16: 実験 2 第 3 回の作業者別の行為の推移 (417 行為)

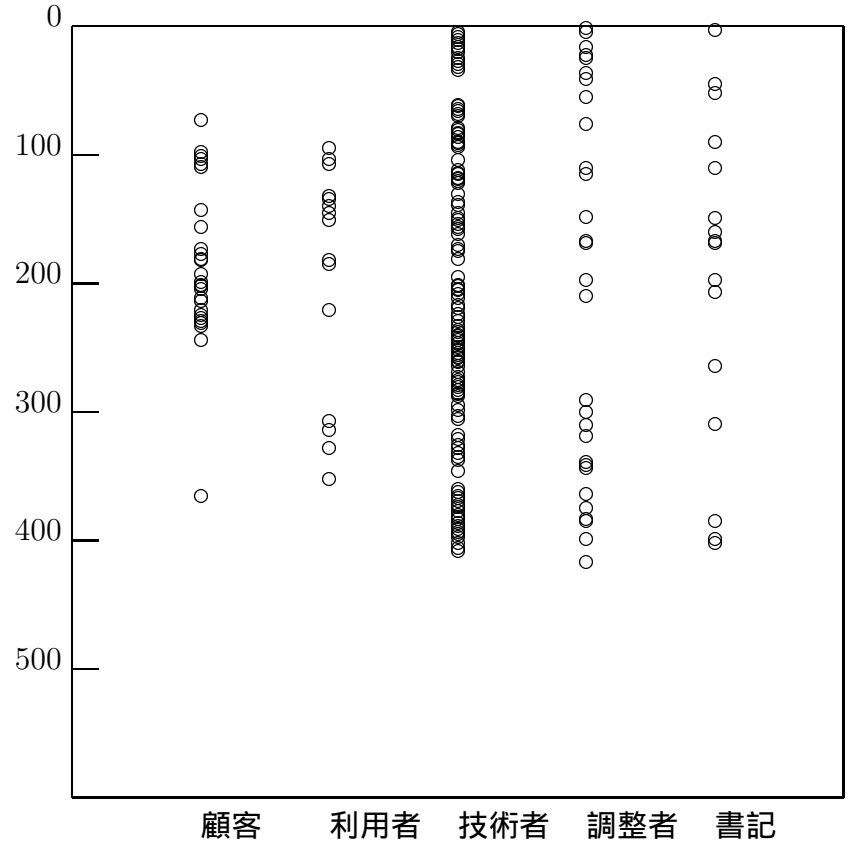


図 3.17: 実験 2 第 3 回の立場別の行為の推移 (417 行為)

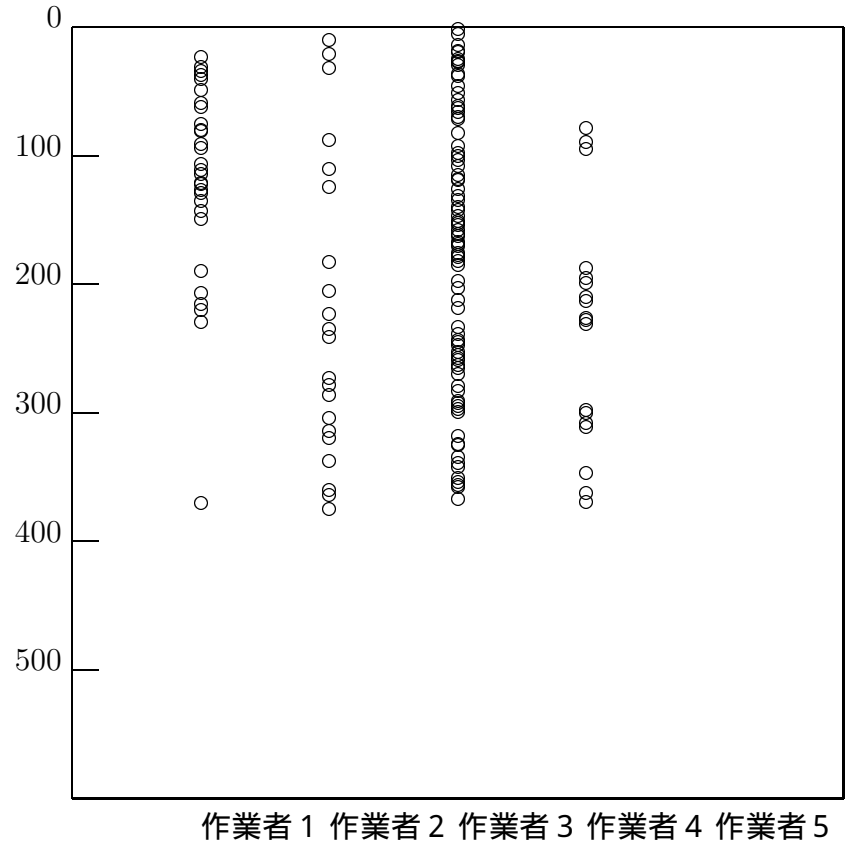


図 3.18: 実験 2 第 4 回の作業者別の行為の推移 (377 行為)

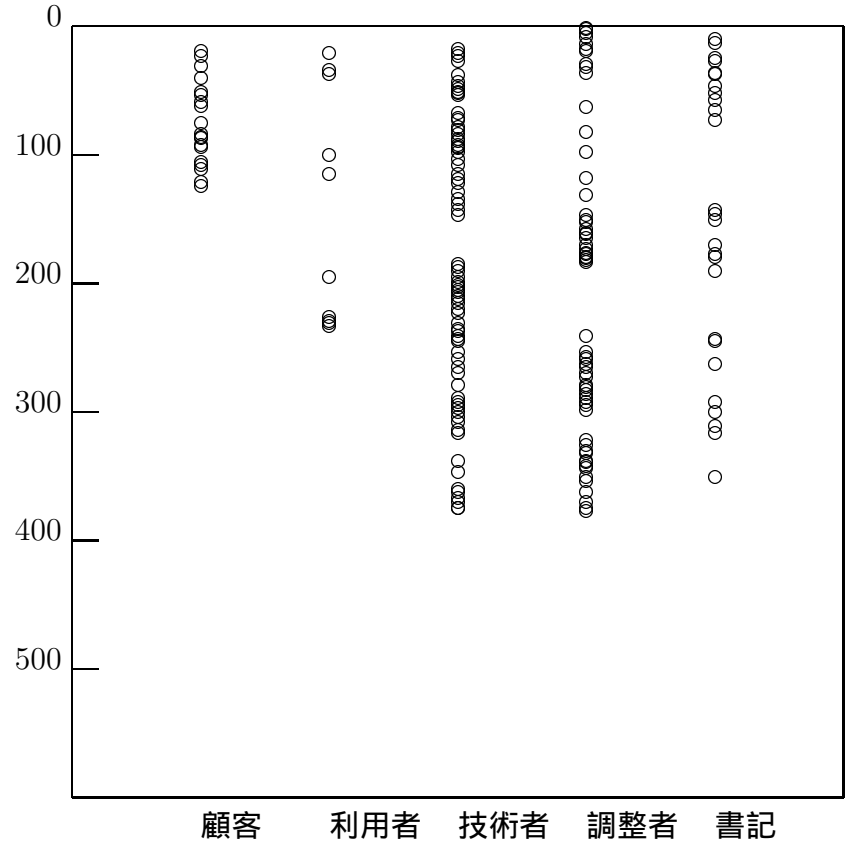


図 3.19: 実験 2 第 4 回の立場別の行為の推移 (377 行為)

第 4 章

実験データの分析と協調作業のモデル

本章では，それぞれの実験に対するデータの分析を行ない，それに基づいて分析法を提案し，分析法の基準を用いて形式化した協調作業のモデルを提案する．

実験 1 は，

1. ソフトウェアの作業分割を強調する役割を作業者に与えている．
2. 対象とするソフトウェアのイメージが作業前に作業者全員に認識されている．

実験 2 は，

1. 顧客や調整者などの作業に関わる立場を強調する役割を作業者に与えている．
2. 対象とするソフトウェアは (顧客と利用者以外の) 作業者にとって事前にイメージが認識されていない．

などの特徴を付けている．

4.1 実験 1

表 3.2 の対応でデータの分析する．実験 1 では，画面，描画などの機能列挙の議論の大半を占めているが，実験 2 ，3 では役割分担された個々のモジュールに対しての議論と分担間の調整が中心となってくる．

結果 **2** 仕様化過程は仕様の列挙，分類の段階と，分類毎の詳細化と分類間の調整の段階に分けられる．

表 3.2 では，作業分割によって分類した行為は時系列の軸において適当な塊になって分布しており，仕様化作業を作業分担に対応した大きな塊としてとらえることができる．そのような塊を 話題 と呼ぶことにする．

結果 **3** (話題) 特定の作業分担に言及している行為の列を 話題 とする．

実験 1 は仕様の列挙分類の段階である．この段階においての協調作業においては以下のような結論が得られる．

結果 4 (立場) 仕様列挙段階では，調整者，書記，製作者，利用者，顧客などの立場に立った作業者の行為の相互作用で協調作業が行なわれる．

結果 4の立場の詳細を 表 4.1 に示す．

立場	役目
調整者	議論の開始．
	顧客に要求を提案するように誘導する．
	利用者と製作者のギャップの相互翻訳．
書記	決定事項の確認．
	項目，機能への命名．
	先送り事項の取り出し．
製作者	顧客の提案を補足する．
	機能を提案する．
	幾つかの選択肢を提案して他の作業者に選ばせる．
	顧客の提案を技術的な観点より分類する．
	実現方法の提案．
	製作上の都合を提案．
利用者	利用者界面に対する要求，選択．
	利用上の都合を提案．
顧客	欲しい機能を提案．
	製作者の提案を評価．
	顧客の提案中の不明な点に対する質問をする．

表 4.1: 立場の内容

典型的に見られた実際の作業例を 6 つのパターンに分けて 図 4.1 から 図 4.6 で例示する．

第 2 ，3 回はモジュールの分担が決まった段階であり，それぞれのモジュールの立場にたった協調作業がみられる．その例を 図 4.7 に示す．

4.2 実験 2

結果 5 実験 1 のデータである 表 3.2 で対比されている図の分布と比べ，表 3.5 は作業者に与えた役割による時系列的な塊が見られない．

実験 2 で作業者に与えた顧客，利用者，技術者，調整者，書記は 図 4.2 のように，ソフトウェアのウォーターフォール型ライフサイクルの作業段階に，ある程度対応させて考えることができる． 結果 5 と 表 4.2 より，以下の結論を得る．

結果 6 ソフトウェアに対する仕様は要求や製作や運用などがあらゆる段階で考慮されている．

つまり，最初に要求仕様を完全に決定してから，製作の仕様に取り掛かるような，段階をはっきり分けた仕様化過程をとっていないことが結論できる．

実験 1 と同様に 図 4.1 の分類に従って整理したものを 図 4.8 に示す．

この例は，実験 1 の局所的分析と同様な作業例が見られる．

例 1
顧客　2 本の直線を 1 本に最適化する機能が欲しいです．
製作者　2 つの四角形を 1 つにするのも同じ機能ですね．
顧客　そうですね．
製作者　逆に 1 つのものを 2 つに分ける機能は必要ですか？
顧客　それも必要です．

例 2
顧客　画面全体の拡大ができたらいいです．
利用者　画面の見え方は，今見えている図形が相対的に小さくなるか，それとも，今見えている図形が全体的に大きくなるかのどちらかがありますね．
製作者　グリッド固定，すなわち前者がいいですね．
顧客　それでは細かい図を書く時にはどうしますか？
製作者　それはべつの機能として必要ですね．

図 4.1: 顧客提案型

例
利用者　画面上の点と直線が交わっていることを示す機能があれば便利です．
製作者　それは図形の端点呼びだしで表せます．
利用者　木のように複数の接点がある場合はどうしますか？
製作者　画面上の同一点は同じ点であることを示す必要があります．
書記　その機能は接着と命名しましょう．

図 4.2: 利用者提案型

例 1
製作者 図形の選び方はどうしますか？
利用者 マウスのクリックでえらべるのがいいです．
利用者 枠をつくってすっぽり入る図形をえらぶのもいいです．
利用者 枠をつくってそれに交わっているものを選ぶのもいいです．
製作者 ソフトウェアを立ち挙げる時に設定すればいいのではないですか？
利用者 ３種類全て使えないと不便です．
製作者 各機能毎に選び方を設定するのはどうですか？
製作者 もっと，簡単に一番最後に使った選び方を継承させる方法はどうですか？
利用者 そうしましょう．

例 2
製作者 グリッドをつけますか？
顧客 グリッドとはなんですか？
製作者 （説明）
顧客 それならつけましょう．

図 4.3: 製作者提案型

作業者の役割	ライフサイクルの段階
顧客	要求
書記	設計
技術者	製作
利用者	運用

表 4.2: 役割とライフサイクル段階の対応

4.3 分析法と協調作業モデル

本章では，実験１，２の実験からえられた知見を基に，仕様化過程を分析するための指針 — 分析法 — を紹介する．

次に，分析法を基にしてまとめた協調作業モデルを提案する．

このモデルでは仕様化作業を３階層で表現する．第１階層は話題の状態遷移であり，第２階層は役割を付けられた作業者の関係，第３階層は役割毎の作業の状態遷移で構成されている．

4.3.1 分析法

我々が観測できるのはどの作業者がどのような行為を行なったかだけである．そこで，複数の作業者によるソフトウェアの仕様化過程を行為の列としてとらえることにする．

行為は，

作業者 × 立場 × 話題影響 × 話題

例 1
設計者 draw 系と paint 系とありますがどちらがいいですか？
調整者 顧客はどのような絵が書きたいのですか？
顧客 （答え）
設計者 それなら draw 系がいいでしょう．
顧客 それでけっこうです．

例 2
製作者 メニューの出し方は，いつも出ているのと，マウスを押したら出てくるのとどちらがいいですか？
顧客 それぞれのイメージを示して下さい．
製作者 （イメージを書いて，それぞれ説明）
顧客 両方使えた方が良いです．

例 3
製作者 文字を入力する時に，画面上に領域をとってそこに，カーソルをだすか，それとも，画面の下にコマンドラインを出してそこに文字を入力した後に，画面上にのせるかのどちらがいいですか？
利用者 図に差し込む程度なので後者でいいのではないですか？
製作者 それでは文字をしたで入力して，入力が終ったら，font size と文字数から大きさを計算して，マウスで画面上の入力位置を決めればいいですね．
調整者 いまのは誰の分担に入るのですか？
製作者 文字入力は文字の人がやって，位置決めは画面か編集のひとが行なえばいいとおもいます．

図 4.4: 製作者選択型

例
調整者 使いたい機能を挙げていって下さい．
顧客 直線，円，長方形，多角形，塗りつぶし，矢印
製作者 矢印は直線に対する修飾です．
書記 （矢印とその他を分類して書く）
顧客 拡大縮小
製作者 拡大縮小は図形とはまた違う範疇なので，別にしましょう．
書記 （別のところに書く）
顧客 回転，コピー，移動，グルーピング
利用者 グルーピングしてから回転などをしますよね．
調整者 それではグルーピングは 1 つ上の段階の機能としましょう．
書記 （書き分ける）

図 4.5: 調整者誘導型

例
書記　　文字にかんして詳細が決まっていません．
製作者　文字の拡大縮小について考えましょう．
利用者　文字の拡大縮小は図形と同様に扱えた方が便利です．
製作者　文字を拡大する時，字を大きくするのですか，それとも，字間を大きくするのですか？
利用者 $_1$　枠が大きくなったら，入る文字数が多くなるのが良いでしょう．
製作者　入力のとき，枠を先に決めてしまうのですか，それとも，入力の始点だけをきめるのですか？
利用者 $_2$　直線のように入力して，拡大すれば文字自体が大きくなるのがいいと私は思いました ．
利用者 $_1$　他の論文などから引用する場合は枠を決めた方がいいでしょう．
調整者　それでは両方とも使えるようにしましょう．

図 4.6: 書記取り出し型

例
編集　画面上で座標をもらって，それを新しい座標に変換して画面に返すの画面の仕事だと思います．
　　座標を選ぶ仕事は画面の人が統一して行なった方がいいです．
画面　それでは編集の人の仕事がすくな過ぎませんか？
編集　描画や，文字入力や，編集の利用者界面は統一した方が保守しやすいと思います．
画面　わかりました．

図 4.7: モジュールの立場による作業例

例 1
顧客　考え方の履歴をとるツールがほしいです．
調整者　具体的な仕様例を示して下さい．
顧客　（示す）
製作者　言語は形式言語を使うのですか？
利用者　自然言語です．
製作者　ビデオで記録するのはだめなのですか？
顧客　あるモデルにあてはめた作業をしてもらい，その履歴を自動的にとりたい．
製作者　わかりました．

例 2
調整者　現行のシステムの足りない点を挙げて下さい．
顧客　作成したカードの関係を木構造的に図示する機能がほしい．
製作者　現行のシステムはなぜできないのですか？
顧客　（理由を述べる）
製作者　わかりました．

例 3
調整者　顧客から要求をだしてください．
顧客　（要求を羅列する）
調整者　イメージが分かりにくいので，実行例を示して下さい．
顧客　そのシステムは被験者の立場と解析者の立場の 2 つの利用者の立場がありますが ..
調整者　その両方の立場を行なって下さい．

図 4.8: 実験 2 の局所的分析

のように表現できる．

作業者は実際のその作業を行なった人物であり，立場は定義 5，話題は定義 4に従う．話題影響は立場と話題との関係の性質を表す．例えば，ある行為は，ある作業者が行なって，その立場は利用者の立場で，話題は入出力で，話題影響は提案である，などである．

以上より，実験的手法を分析法は以下のように提案する．

1. 実験記録より，作業者の行為を取り出す．
2. その行為をどの作業者が行なったかを記述．
3. その行為がソフトウェアのどの部分に言及しているかを記述．
4. その行為が話題に対してどのような影響を与えているかを記述．
5. その行為がどのような立場に立っているかを記述．

このように記録に対して整理を行なうことで分析を行なう．

4.3.2 協調作業モデル

分析法の指針を基に，図 4.9 に示すような協調作業のモデルを構築した．

モデルは 3 階層に分かれる．

3 つの階層は順に，global layer, personal layer，protocol layer に分かれる．

global layer は話題の状態遷移であり，対象とするソフトウェアに対する注目点の移動に対応する．

personal layer は作業者を表す階層であり，それぞれの作業者は話題に対してなんらかの影響を与えることで，global layer との関係を持っている．作業者同士は直接に通信をするのではなく，話題を通して通信を行なう．

protocol layer は作業者の立場ごとの状態遷移を表す階層であり，作業者が特定の立場に立つことによって，その作業者はその立場における protocol に従うことになる．現在考えられる立場は，

顧客	···	最終的にソフトウェアを受けとることを前提とした問題領域に依存する立場．
利用者	···	ソフトウェアを実際に利用することを前提とした利用領域に依存する立場．
製作者	···	実際にプログラムを作ることを前提とした技術領域に依存する立場．
書記	···	仕様化過程の記録をとり，実際に仕様書を書くことを前提とした領域に依存する立場．
調整者	···	作業進行を調整し，速やかなる決定を促すことを前提とした領域に依存する立場．

などがある．

protocol layer における立場毎の状態遷移と，personal layer における作業者の対応は，それぞれの話題によって異なる．

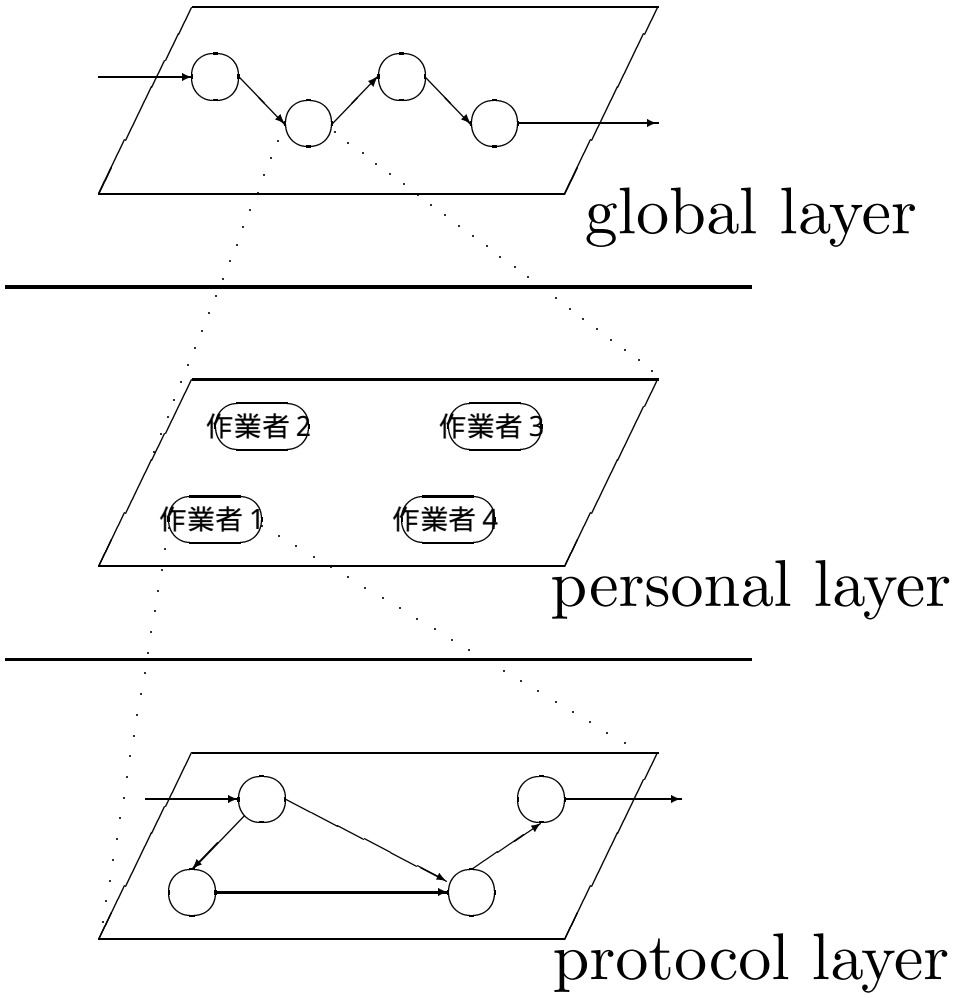


図 4.9: 協調作業モデルの概要図

第 5 章

議論

5.1　モデルの事例としての実験

5.1.1　実験 1

global layer の話題の遷移は特定の問題に依存するので，protocol layer における役割毎の状態遷移を記述してみる．

4.1節で紹介した立場を分類基準とした 6 つの話題に関するパターン毎のプロトコルを以下に示す．

顧客提案型

図 4.1 の作業例を protocol layer の状態遷移として表現する．

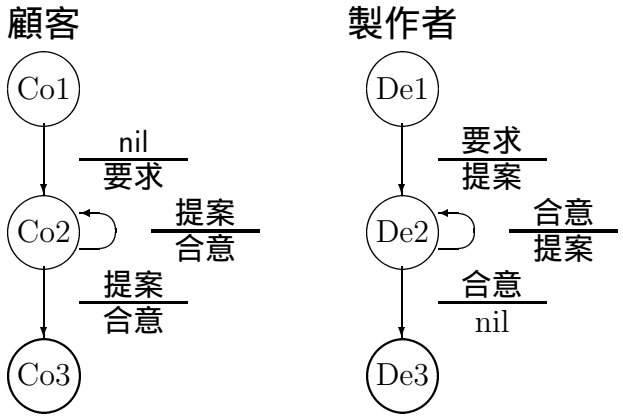


図 5.1: 顧客提案型の protocol layer の例 1

図 5.1 に顧客提案型の protocol layer を図示する．

protocol layer は役割毎の状態遷移を記述する．この例では 顧客 と 製作者 が役割として存在する．

(Co1) から (Co3) は顧客の状態，(De1) から (De3) が製作者の状態である．顧客は初期状態 (Co1) で何らかの話題に対して 要求 をだす．**nil** は直前の話題とは違う話題に対して言及する

行為をその作業者が行なう場合の作業者にたいする入力である．その 要求 を製作者が (De1) で受けて，その話題に対して 提案 を出力する．その 提案 を顧客が (Co2) で受けて話題に対し 合意 を出して (Co3) に遷移し終了するか，もしくは (Co2) に遷移し提案を待つ．そして製作者は話題に対する 合意 を受けて，(De3) に遷移し終了するか，(De2) に遷移し合意を待つ．

以上の状態遷移を以下のように式表現する．

$$
\begin{aligned}
\text{顧客}: \quad & (Co1 - nil/\text{要求} - Co2) \\
& (Co2 - \text{提案}/\text{合意} - Co2) \\
& (Co2 - \text{提案}/\text{合意} - Co3). \\
\text{製作者}: \quad & (De1 - \text{要求}/\text{提案} - De2) \\
& (De2 - \text{合意}/\text{提案} - De2) \\
& (De2 - \text{合意}/nil - De3).
\end{aligned}
$$

以下のパターンでは式表現で protocol を記述する．

もう一つの顧客型のパターンを記述する．

$$
\begin{aligned}
\text{顧客}: \quad & (Co1 - nil/\text{要求} - Co2) \\
& (Co2 - \text{提案}/\text{質問} - Co3). \\
\text{利用者}: \quad & (Us1 - \text{要求}/\text{提案} - Us2). \\
\text{製作者}: \quad & (De1 - \text{提案}/\text{提案} - De2) \\
& (De2 - \text{質問}/\text{解答} - De3).
\end{aligned}
$$

利用者提案型

図 4.2 の作業例を protocol layer の状態遷移として表現する．

利用者は，ある話題の 提案 に対して適当な 反証 を行なう行ない，次の話題に対する行為を待つ．

$$
\begin{aligned}
\text{利用者}: \quad & (Us1 - nil/\text{要求} - Us2) \\
& (Us2 - \text{提案}/\text{反証} - Us2) \\
& \cdots \quad . \\
\text{製作者}: \quad & (De1 - \text{要求}/\text{提案} - De2) \\
& (De2 - \text{反証}/\text{新提案} - De3) \\
& \cdots \quad . \\
\text{書記}: \quad & (Se1 - \text{新提案}/\text{命名} - Se2) \\
& \cdots \quad .
\end{aligned}
$$

製作者提案型

図 4.3 の作業例を protocol layer の状態遷移として表現する．

$$
\text{製作者}: \quad (De1 - nil/\text{提案} - De2)
$$

$$
\begin{array}{rc}
 & (De2 - \text{説明/質問} - De2) \\
 & (De2 - \text{説明/提案} - De3) \\
 & (De3 - \text{合意}/nil - De4) \quad . \\
\text{利用者} & (Us1 - \text{提案/解答} - Us2) \\
 & (Us2 - \text{解答} \mid \text{質問/説明} - Us2) \\
 & (Us2 - \text{提案/合意} - Us3) \quad .
\end{array}
$$

$(Us2 - \text{解答} \mid \text{質問/説明} - Us2)$ の $\mid$ は話題に対して解答もしくは質問がされた場合，話題にたいして説明をするということを表現するのに用いる．

製作者選択型

図 4.4 の作業例を protocol layer の状態遷移として表現する．

$$
\begin{array}{rc}
\text{製作者}: & (De1 - nil/\text{選択肢} - De2) \\
 & (De2 - \text{選択/確認} - De3) \\
 & (De3 - \text{質問/解答} - De4) \\
 & \cdots \quad . \\
\text{利用者}: & (Us1 - \text{選択肢/選択} - De2) \\
 & \cdots \quad . \\
\text{調整者}: & (Co1 - \text{確認/質問} - Co2) \\
 & \cdots \quad .
\end{array}
$$

調整者誘導型

図 4.5 の作業例を protocol layer の状態遷移として表現する．

$$
\begin{array}{rc}
\text{調整者}: & (Co1 - nil/\text{誘導} - Co1) \\
 & (Co1 - \text{誘導/提案} - Co2) \\
 & (Co2 - \text{意見/提案} - Co2) \\
 & (Co2 - \text{意見/合意} - Co3) \quad . \\
\text{書記}: & (Se1 - \text{合意/記述} - Se1) \quad . \\
\text{設計者}: & (De1 - \text{提案} \mid \text{説明} - De1) \quad . \\
\text{顧客}: & (Cu1 - \text{誘導/提案} - Cu2) \\
 & (Cu2 - \text{意見/提案} - Cu2) \\
 & (Cu2 - \text{意見/合意} - Cu3) \quad . \\
\text{利用者}: & (Us1 - \text{提案/説明} - Us1) \quad .
\end{array}
$$

書記取り出し型

図 4.6 の作業例を protocol layer の状態遷移として表現する．

$$
\text{書記}: \quad (Se1 - nil/\text{取り出し} - Se2)
$$

$$(Se2 - \text{合意/記述} - Se3) \quad .$$

設計者 : $(De1 - \text{取り出し/誘導} - De2)$

$$(De2 - \text{提案} \mid \text{解答/質問} - De2)$$

$$(De2 - \text{解答/合意} - De3) \quad .$$

利用者 : $(Us1 - \text{誘導/提案} - Us2)$

$$(Us2 - \text{質問/解答} - Us2) \quad .$$

5.1.2 実験 2

図 4.8 の作業例を protocol layer の状態遷移として表現する．実験 2 では，特に実験 1 のように作業例に対し作業例に対しパターン名をつけていない．

顧客 : $(Cu1 - nil/\text{要求} - Cu2)$

$$(Cu2 - \text{説明指令/説明} - Cu2)$$

$$(Cu2 - \text{提案/反論} - Cu2)$$

$$(Cu2 - \text{質問/解答} - Cu2) \quad .$$

調整者 : $(Co1 - \text{要求/説明指令} - Co1) \quad .$

製作者 : $(De1 - \text{説明/質問} - De2)$

$$(De2 - \text{解答/提案} - De2)$$

$$(De2 - \text{反論/納得} - De3) \quad .$$

顧客 : $(Cu1 - \text{誘導/要求} - Cu2)$

$$(Cu2 - \text{質問/解答} - Cu2) \quad .$$

調整者 : $(Co1 - nil/\text{誘導} - Co1) \quad .$

製作者 : $(Co1 - \text{要求/質問} - Co2)$

$$(Co2 - \text{解答/合意} - Co3) \quad .$$

調整者 : $(Co1 - nil/\text{誘導} - Co2)$

$$(Co2 - \text{要求/説明指令} - Co2) \quad .$$

顧客 : $(Cu1 - \text{誘導/要求} - Cu2)$

$$(Cu2 - nil/\text{要求} - Cu2)$$

$$(Cu2 - \text{説明指令/説明} - Cu2) \quad .$$

5.2 分析法の評価

- 話題の開始と終了について:

ソフトウェアの注目点が変化したことで，話題の変化を検出することができるが，実際には複数のソフトウェアの部分を同時に議論する場合が存在するので，話題の特定が難しい．

- 作業者と役割について:

  実験の結果から作業者の役割は，作業者の行為を束縛するので，どの作業者が行なったかによってどの役割の行為かを特定するのは妥当である．

  また，作業者の間ではなく，役割の間に静的な関係があるようなので，その関係を特定する必要がある．

- 話題の重複の検査:

  今まで言及された話題とその議論を整理することで，話題の重複で意味のあるものとないものが特定できるはずである．

  また，話題間の静的な関係を特定する必要がある．

5.3 協調作業モデルの評価と改良

分析法の評価を基に，作業者関係と話題間関係と話題状態遷移を加えた新しいモデルを提案する．その概要図を 図 5.2 に示す．これは，図 4.9 で示したモデルに product relation と role relation を加えることにより拡張を行なったモデルである．

product relation は作業に現れる話題間の関係を表したものである．global layer の話題の集合から product relation の個体の集合は多対一対応もしくは一対一対応となっている．なぜなら，話題は global layer にそって生成もしくは，言及されていくからである．

role relation は役割の集合と役割間の関係を表している．この関係は，例えば，役割間の対立関係や仲間関係や主従関係に対応している．それぞれの役割に対して，protocol layer によって役割に応じた作業の状態遷移が定義される．

global layer のある 1 つの話題について，role relation の役割の集合の部分集合と personal layer の作業者の集合の部分集合は一対一対応を持っている．すなわち，特定の作業者はたかだか 1 つの役割を持ち，特定の役割はたかだか 1 人の作業者に対し割り当てられていることを示す．

5.4 協調作業の支援について

協調作業の支援をする環境を構築するためには，以下の点が重要である．

- 作業者の役割を明確にした作業指針の明確化．
- 役割間の関係を明確化．
- 決定，もしくは未決定の話題の整理．

我々の協調作業モデルはこれらの点をモデルに包含しており，我々のモデルを基にして支援環境を構築すればこれらの問題点の支援を期待できる．

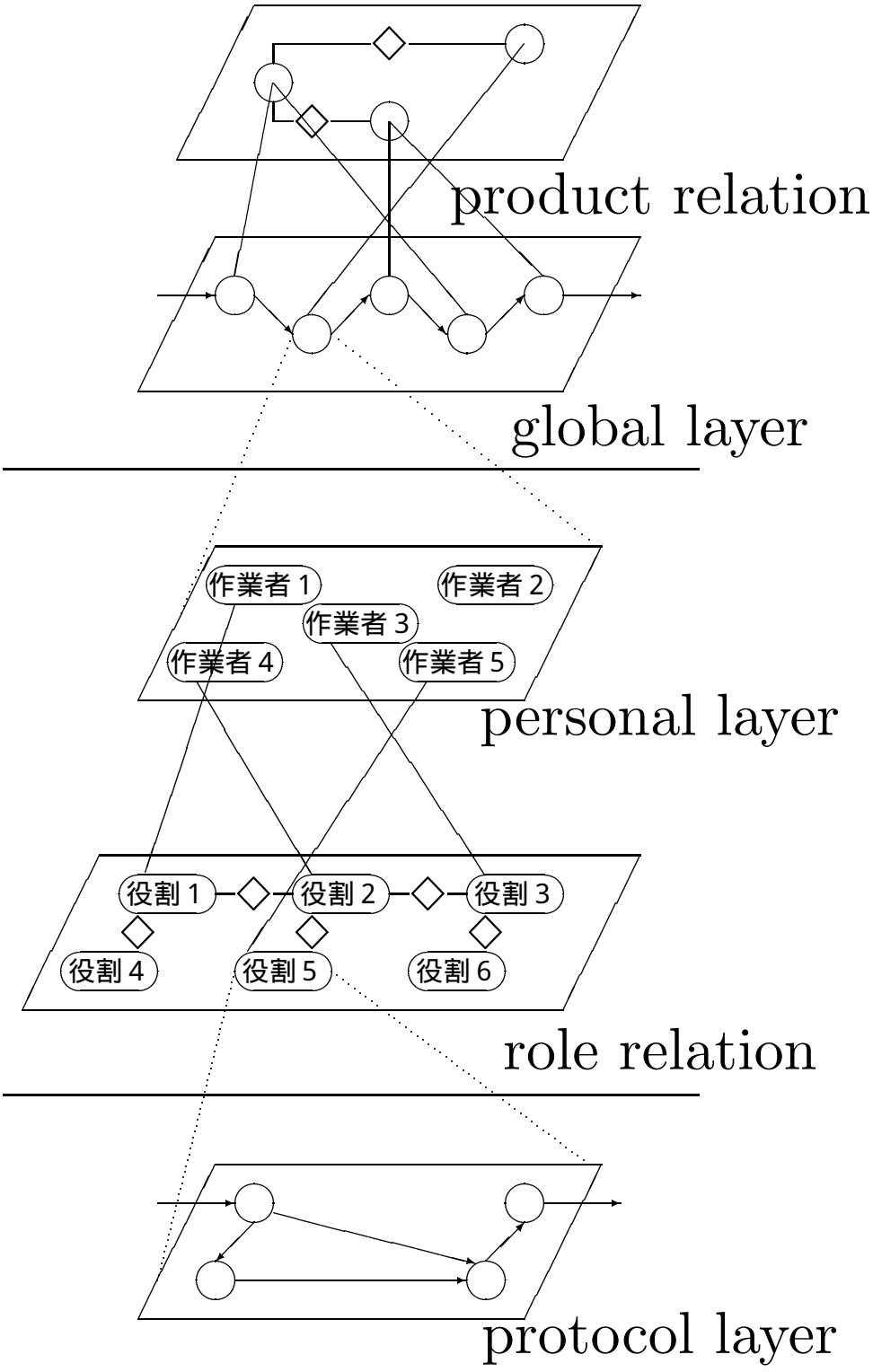


図 5.2: 新しい協調作業モデルの概要図

実験作業中には，対象とするソフトウェアのモジュール以外の話題，例えば，次の作業日をいつにするかなどの作業工程に関する話題などが存在した．

我々のモデルにおいては，話題をソフトウェアのモジュールに対応付けて考えているが，モデルに対して変更を加えずに，ソフトウェアのモジュール以外の話題を取り込むことが可能である．

第 6 章

おわりに

仕様化過程における協調作業を明らかにするための，基礎実験を行ない，分析のための方法を提案した．その方法を基に，協調作業モデルを構築し，実験をモデルの事例として整理した．その整理結果を基に，協調作業モデルの拡張をおこなった．

議論で提示した協調作業モデルの事例は今回の限られた実験の記述にすぎず，一般的な協調作業がこの事例に従うわけではない．

今後，実験を数多く行ない，その分析を行なって作業事例を集めることで協調作業モデルの意味付けを行なう必要がある．

問題点および今後の研究課題として，以下のような項目が考えられる．

1. 実験方法について

   今回の実験では特定の仕様化技法，仕様記述言語を用いないで実験を行なった．今後，構造化分析法，ジャクソン開発法といった仕様化/設計の方法論に従って協調作業を進める場合についても，同様な実験，分析を行い，本研究の手法を評価する．

2. 対象とするソフトウェアについて

   今回の実験では，対話型のソフトウェアに対して実験を行なった．今後，標準問題 [3] などを取り上げる．

3. 分析方法について

   分析方法は提案したが，実際の分析作業を行なうことは大変な作業量である．今後，実験を数多く行ない事例を蓄積していくには，分析方法の自動化を考えなくてはならない．

4. 話題間関係，役割間関係

   議論で述べた新しい協調作業モデルに示した product relation, role relation に対応する側面を具体的に分析し，意味付けする必要がある．

5. 個人の仕様化作業との比較

   仕様化過程において個人で行う方が良い作業とグループで行った方が良い作業が存在する．

   例えば，ソフトウェアに対する一貫したイメージを保つのは個人での作業の方が有利であるが，多面的な視野から仕様を評価検討するには，グループ作業の方が有利である．

   [15], [14] では個人の仕様化過程についての分析を行っており，これらの研究成果との比較検討をする必要がある．

6. 協調作業支援ツールについて

   複数人で仕様化作業を行なう際の協調作業を支援するツール，例えばどの項目について同意が得られたか，どの項目が未解決となっているか等を整理，管理し，提示することによって，仕様作成をガイドするツールをワークステーション上に実現する．

   また，協調作業の利点を利用できるような，個人作業の支援ツールも考えられる．

謝辞

本研究を進めるにあたり，始終御指導戴いた佐伯元司助教授に感謝いたします．また，数々の助言をいただいた本間学氏，快く実験に参加してくれた知識システム講座の皆様に感謝いたします．

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] Bill Curtis. Implication from empirical studies of the software design process. In *Proceedings of an International Conferrence organized by the IPSJ to communicate the 30th Anniversary*, 1990.

[3] R. A. Kemmerer, S. White, A. Mili, and N. Davis. Problem set for fourth international workshop on software specification and design. In *4th International Workshop on Software Specification and Design*, pages 9–10, Apr 1987.

[4] David A. Marca. Specifying coodinators: guidelines for goupware developers. In *5th International Workshop on Software Specification and Design*, pages 235–237, May 1989.

[5] Mark Stefik, Gregg Foster, Daiel G. Bobraw, Kenneth kahn, Stan Lanning, and Lucy Suchman. BEYOND THE CHALKBOAD: computer supported for collaboration and problem solving in meetings. *Communications of the ACM*, 30(1):32–47, Jan 1987.

[6] Terry Winograd. A language perspective on the design of coopearative work. In *CSCW'86 Proceedings*, Dec 1986.

[7] Terry Winograd. A language/action perspective on the design of cooperative work. *Journal Of Human-Computer Interaction*, 3, 1987.

[8] Terry Winograd. Where the action is. *BYTE*, 13(13), Dec 1988.

[9] Terry Winograd and Fernand Flores. *Understanding Computers and Cognition*. Ablex Publishing Corporation, Norwood, N.J., 1986.

[10] 岸本　三江，西田　正吾，後藤　柳一郎．ソフトウェア意思伝達支援ツールＣＯＭＩＣＳ（１）（２）．，情報処理学会第３７回全国大会論文集，857–860 ペ－ ジ，1988 年後期．

[11] 岸本　三江．ソフトウェア生産プロセスにおけるインターラクションの分析．，情報処理学会第３５回全国大会論文集，1141–1142 ペ－ ジ，1987 年後期．

[12] 海谷，佐伯．ソフトウェアの仕様化過程における協調作業の分析．，情報処理学会ソフトウェア工学研究会，77–4 ペ－ ジ，2　1991．

[13] 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，31(1):124–135，Jan　1990．

[14] 池，海谷，佐伯，本間．ソフトウェア仕様記述過程分析のための基礎実験．，情報処理学会ソフトウェア工学研究会，68–5 ペ− ジ，9　1989．

[15] 渡辺智弘，佐伯元司．思考過程モデルに基づいたツールを用いた仕様化作業の分析．，電子情報通信学会全国大会，6:43 ペ− ジ，Oct　1990．

[16] 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５，149–162 ペ− ジ，Sep　1990．