

通信サービスソフトウェア  
自動作成方式の研究

通信サービス



# 通信サービスソフトウェア 自動作成方式の研究

新津善弘

# 目次

## 第1章 序論

1.1 序言	1
1.2 研究の背景	2
1.3 本論文の構成と概要	6

## 第2章 通信サービスソフトウェア自動作成方式の提案

2.1 まえがき	9
2.2 通信ソフトウェア開発の問題点と解決法	10
2.2.1 通信ソフトウェア開発の問題点と解決のアプローチ	10
2.2.2 新ソフトウェア構造通信システムの導入	10
2.2.3 新通信ソフトウェア構造における自動作成対象とその特徴	11
2.3 通信サービスソフトウェア自動作成方式の提案	15
2.3.1 非エキスパートによる通信サービスソフトウェア作成の必要性	15
2.3.2 通信サービスソフトウェア自動作成方式の対象工程	16
2.3.3 通信サービスソフトウェア自動作成方式	18
2.4 通信サービスソフトウェア自動作成方式の構成技術	21
2.4.1 通信サービスの仕様設計/ソフトウェア生成技術	21
2.4.2 通信サービス仕様の検証技術	22
2.4.3 通信サービス仕様の試験・評価技術	22
2.5 むすび	28

## 第3章 通信サービスの仕様設計/ソフトウェア生成技術

3.1 まえがき	29
3.2 通信サービスの仕様設計/ソフトウェア生成技術の概要	29
3.3 仕様設計/ソフトウェア生成システム構築の設計方針	32
3.4 通信サービスの仕様記述法	33
3.4.1 非エキスパートのための仕様記述法	33
3.4.2 メッセージ記述言語 MSDL	34



3.4.3 網内処理機能の仕様記述法	35
3.4.4 拡張形情報シーケンス記述法適用困難なサービスの対処法	36
3.5 仕様入力/編集法	49
3.5.1 インテリジェントエディタ(IED)の構成	49
3.5.2 データベースを用いたサービス仕様入力法	49
3.5.3 画面制御による仕様編集支援	49
3.6 仕様/プログラム自動変換法	53
3.6.1 変換知識データベースの構成法	53
3.6.2 自動仕様変換法	53
3.6.3 準正常手順の自動付加法	54
3.6.4 冗長仕様の最適化手法	55
3.6.5 サービス記述要素追加手法	55
3.7 試作による評価	68
3.7.1 試作システムの構成	68
3.7.2 仕様設計/ソフトウェア生成試作システムの評価	68
3.8 むすび	73

#### 第4章 通信サービス仕様の検証技術

4.1 まえがき	74
4.2 通信サービス仕様の検証・試験の体系	76
4.2.1 非エキスパート設計通信サービス仕様の検証・試験における要求課題	76
4.2.2 非エキスパートのソフトウェア開発における検証・試験技術の体系	77
4.3 単体検証の分類と検証項目の明確化	80
4.4 変換レベルに対応させた階層形仕様検証法	82
4.4.1 概要	82
4.4.2 サービス手順の構文検証法	82
4.4.3 サービス手順の論理検証法	83
4.5 段階的検証方式	87
4.5.1 マクロ化シーケンス検証	87
4.5.2 遷移手順検証	90
4.6 検証結果の上位仕様逆変換表示法	100
4.6.1 下位レベル記述の通信サービス仕様の逆変換	100

4.6.2 検証結果の上位仕様逆変換表示法	100
4.7 複合検証における検証項目	102
4.8 通信サービス仕様間競合検証方式	103
4.8.1 サービス競合検証方式	103
4.8.2 検証対象限定手法	103
4.9 検証方式の試作と評価	106
4.9.1 段階的検証方式試作システム	106
4.9.2 試作結果と考察	106
4.10 むすび	114

#### 第5章 通信サービス仕様の試験・評価技術

5.1 まえがき	115
5.2 通信サービス仕様の試験・評価技術の概要	116
5.3 通信サービス仕様の試験法	118
5.3.1 概要	118
5.3.2 通信サービス仕様試験環境の構成法	119
5.3.3 サービスプロトタイプングシステム構成法	120
5.4 通信サービス仕様試験システムの試作と評価	136
5.4.1 試作システムの構成	136
5.4.2 試作結果	136
5.4.3 サービスプロトタイプングシステムを用いた試験法の効果	137
5.5 通信サービス仕様におけるマン・マシンインタフェース評価法	144
5.5.1 通信サービス仕様におけるマン・マシンインタフェース設計の問題点	144
5.5.2 サービスプロトタイプングシステムを用いたマン・マシンインタフェースの評価法	145
5.6 サービスプロトタイプングシステムを用いた仕様評価法によるモニタ評価結果と考察	153
5.6.1 高度化電話会議サービス仕様のモニタ評価	153
5.6.2 モニタ評価結果	153
5.6.3 考察	154
5.7 マン・マシンインタフェースの設計ガイドライン	160



5.7.1 一般的指針	160
5.7.2 操作手順の設計ガイドライン	160
5.7.3 音声ガイダンス表現の設計ガイドライン	161
5.8 むすび	162

## 第6章 インテリジェントネットワーク用通信サービスソフトウェア作成環境への適用

6.1 まえがき	163
6.2 INアーキテクチャ	163
6.2.1 階層形アーキテクチャ	163
6.2.2 サービス制御メカニズム	164
6.3 仕様設計/ソフトウェア生成技術のINへの適用	167
6.3.1 INサービスシナリオ作成環境に対する要求条件	167
6.3.2 仕様設計/ソフトウェア生成技術適用法の概要	167
6.3.3 詳細条件仕様入力法	168
6.3.4 仕様変換アルゴリズム	169
6.4 サービス仕様検証、試験・評価技術のINサービスシナリオ作成環境への適用	175
6.4.1 通信サービス仕様の検証技術適用法	175
6.4.2 通信サービス仕様の試験・評価技術適用法	175
6.5 INサービスシナリオ自動作成環境	175
6.6 むすび	178

## 第7章 結言

謝辞	183
文献	184

# 第1章 序論

## 1.1 序言

通信網はSPC化されたにもかかわらず、ソフトウェアの持つ融通性、柔軟性が活かせるような環境となっていない。この原因は、主に通信ソフトウェアが一旦作成されて通信網に埋め込まれると、長期間変更されることがなかったり、一部の機能変更を除いては大きく変えることがないことを前提として作成されていることに起因している。このため、通信ソフトウェアの走行環境が専用の環境となっていること、作り込む通信ソフトウェアが専用のものとして特化されており、新たな機能追加・変更が容易でなく、通信ソフトウェア作成の自動化が困難な環境を作りだしている。

従来の通信網関連のソフトウェア技術は、制御対象とするシステムのソフトウェア構成が階層的に整理されてないため、そのしわ寄せが通信ソフトウェアの作成環境に及んでいた。このため、一般に通信サービスの仕様はシステムのアーキテクチャを隠蔽し、簡単であるほど望ましいが、システムを制御するプログラムとのギャップは極めて大きくなり、サービス作成環境だけでそのギャップを自動的に埋めることは実用的なシステムの構築を難しくさせていた。このため、作成環境における仕様入力をできるだけ上位のレベルで行なえて、制御対象のシステムがより高い記述レベルの制御プログラムをそのまま解釈して動作する、通信システムの構築が望まれる。

最近、通信システムの制御対象のハードウェアを論理化して見せ、各サービスに共通な機能を汎用部品として切り出して、サービスに対応した部分のみのソフトウェア（通信サービスソフトウェア）を作成するだけで、自由にサービスを開発できるサービスプラットフォームを構成する方式に対する研究開発が盛んになってきている。インテリジェントネットワークがその代表的な例であり、伝達網の交換機やPBXの構成もこれにはほぼ類似の構成が検討されている。このような環境の下では、サービス対応の通信サービスソフトウェアに限定すれば、自動作成が可能な状況が生まれてきた。

本論文では、上記のようなサービスプラットフォームを持つ通信システムを前提に、この上で動作する通信サービスソフトウェアを対象とし、非エキスパートが通信サービスを仕様設計し、通信サービスソフトウェアを自動作成するための実現技術及びそのシステム構成技術を明らかにすることにより、通信サービスソフトウェア自動作成方式を確立することを目的とする。

本目的の検討の前に、まず本研究の背景として、通信サービス及び通信ソフトウェアを取り巻く状況を次節で見ることにする。

## 1.2 研究の背景

通信サービスと通信ソフトウェアについて、まず要求条件と課題を見ていくことにする。通信サービス開発への要求条件の要素としては主に、サービス提供条件、サービス開発者、サービス動作環境の3つが挙げられる。この要求条件とこれらより導かれる通信ソフトウェアの課題を図 1.1 に示す。

一方、これらの課題に対して現状を見ると、通信システムのソフトウェア構造、OS、仕様記述法、検証技術、評価・試験技術等、いくつかの点が課題解決を困難にさせている。図 1.2 には、通信ソフトウェアの課題とこれらの課題解決を困難にさせている要因の関係を示している。

また、従来技術をそのまま適用することにおける問題点は、仕様記述言語、仕様入力法、仕様検証法、仕様評価法について表 1.1 に示す。

これらの背景を基に、2章以降、通信サービスソフトウェア自動作成方式について論じる。

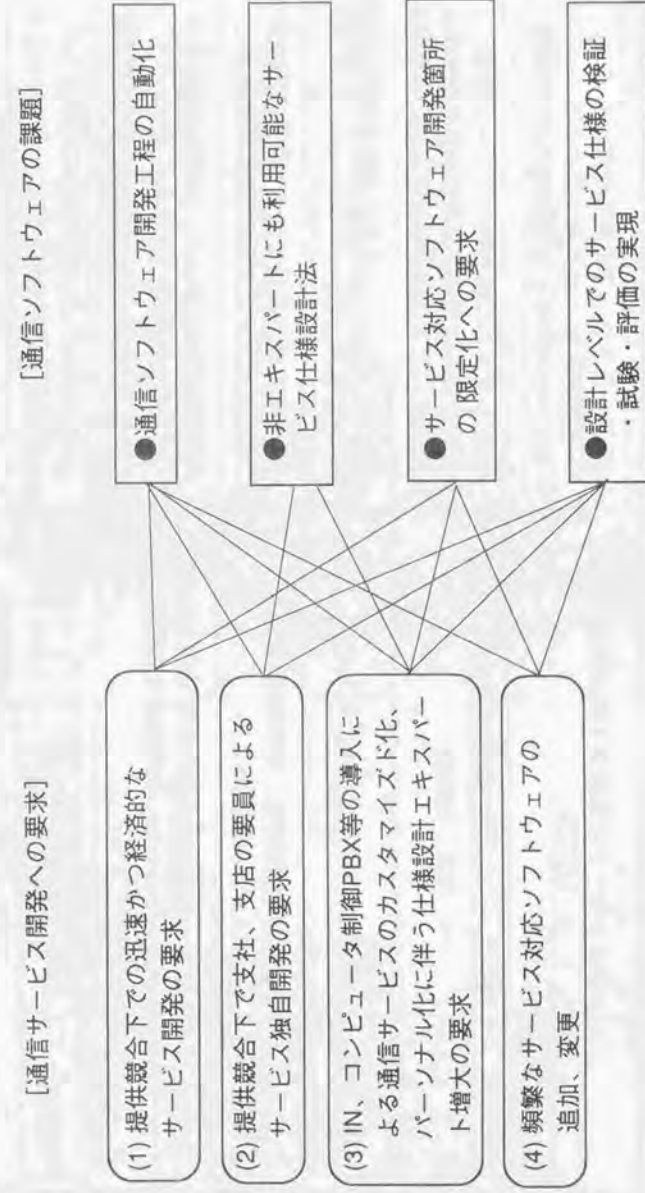


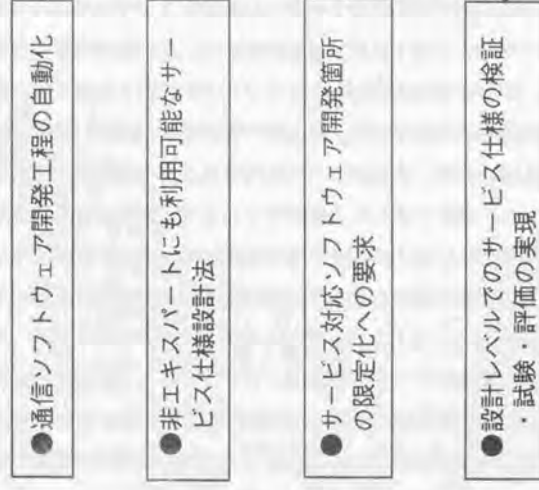
図 1.1 通信サービス開発への要求と通信ソフトウェアの課題



表 1.1 従来技術適用の問題点

技術項目	従来技術	適用の問題点
仕様記述言語	●メッセージの送受信を時系列上に表現 → LOTOS	<ul style="list-style-type: none"> <li>一通信ソフトウェアの設計仕様として一般的なSDLへの変換に難がある。</li> <li>非エキスパートには理解性の点で不向き</li> </ul>
	●メッセージの送受信を状態遷移で表現 → SDL, Estelle	<ul style="list-style-type: none"> <li>一通信システムの仮想的なリソース構成、制御に関する知識が要求され、スキルの低いサービス開発者の仕様入力には不向き</li> </ul>
仕様入力法	●図形エディタによる仕様入力	<ul style="list-style-type: none"> <li>操作性が低く、入力時に必要な知識を暗記する必要がある。</li> </ul>
仕様検証法	●到達可能解析法、時相論理を用いたプロトコル検証法	<ul style="list-style-type: none"> <li>時間的な順序関係に加えて信号の意味まで考慮するサービス手順検証には不十分</li> </ul>
	●サービス仕様の動作を計算機上でシミュレーションし、動作結果を画面表示	<ul style="list-style-type: none"> <li>電話機等の音声メディアを用いる通信サービスの仕様評価には不十分、特にマン・マシンインタフェース仕様の評価が不可</li> </ul>
仕様評価法	●キーボード入力/ディスプレイ出力の端末を用いた通信サービスのマン・マシンインタフェース仕様設計法	<ul style="list-style-type: none"> <li>PBボタン入力/音声出力に制限された電話系通信サービスのマン・マシンインタフェース仕様の設計には適用不可</li> </ul>

[課題]



[困難にさせている要因]

図 1.2 通信ソフトウェアの課題解決を困難にしている要因



### 1.3 本論文の構成と概要

本論文では、通信システムの制御対象のハードウェアを論理化して見せ、各サービスに共通な機能を汎用部品として切り出して構成した、サービスプラットフォーム上で動作する通信サービスソフトウェアを対象とし、設計の非エキスパートであるサービス開発者が使用できる通信サービスソフトウェアの自動作成技術を明らかにする。

サービスプラットフォーム上で、サービス対応のアプリケーションソフトウェアである通信サービスソフトウェアのみの開発により、新規サービスの提供が可能なサービス実行環境を前提とし、非エキスパートによる通信サービス仕様の設計・制御ソフトウェアの自動生成を実現する通信サービスソフトウェア自動作成方式の提案を第2章で行ない、以下第2章で示した各構成技術の主要技術課題について、構成技術対応に第3章～第5章の各章で研究内容を述べる。また、本方式の研究結果の具体的適用例として、現在実用化中のインテリジェントネットワークにおける通信サービスソフトウェア作成環境への適用を第6章に示す。

次に、各章の内容について順次、概要を説明する。

第2章では、従来の通信ソフトウェア開発の問題点を挙げ、通信ソフトウェア開発自動化への課題を明らかにする。また、通信ソフトウェア開発を効率化し自動作成の前提となる、通信システムにおけるサービスプラットフォームの構成とその上で走行し個々のサービスを実現する通信サービスソフトウェアについて示す。この前提の基に、サービス開発の自動化を推進し、非エキスパートによるソフトウェア開発を実現する、通信サービスソフトウェア自動作成方式を提案する。本方式の実現技術として、仕様設計/ソフトウェア生成技術、仕様検証技術、仕様試験・評価技術の3つを明らかにし、それぞれの主要技術課題を抽出する。本章で提案する通信サービスソフトウェア自動作成方式では、非エキスパートがサービス仕様を図形式で簡単に設計でき、サービスプラットフォームを持つ通信システム上で動作可能な通信サービスソフトウェアを自動作成できる画期的な方式である。

本方式は、現在世界的に次世代のサービス提供ネットワークとして注目され、国際標準の場であるCCITTでも検討が盛んとなっている、インテリジェントネットワークで提供するサービス開発には必須の技術である。各国ともサービス作成環境の重要性につい

ては共通認識があり、いくつかの研究が報告されているが、我国のインテリジェントネットワークでは積極的にカスタマの要求を取り入れたサービス提供を行ない非エキスパートによるサービス開発が求められる点で、他国におけるサービス作成環境に比べ、より厳しい要求条件となっている。本論文で確立した通信サービス自動作成技術は、通信ソフトウェアの仕様記述法の標準化と共に、現在開発中のインテリジェントネットワークの構成技術の標準化技術確立のためにも貢献できると考えられる。

第3章では、通信サービスの制御手順を非エキスパートなサービス開発者が作成できるような簡単なユーザインタフェースのみを見せ、目的の通信システム上で走行する通信サービスソフトウェアを生成するための、仕様設計/ソフトウェア生成技術について述べる。ここでの着想の独創性は、メッセージシーケンス記述言語と網内処理機能記述で構成される拡張形情報シーケンス記述法により、非エキスパートによる図形式による簡易な仕様記述を可能にすること、エキスパートの持つ知識で非エキスパートのサービス仕様設計を支援すること、自動変換の過程で不足仕様の付加と冗長仕様の最適化をある程度まで実現することの3つであると言える。

試作システムにより、本仕様設計/ソフトウェア生成技術について評価し、有効性を明らかにする。

第4章では、通信サービス仕様の検証・試験技術の体系的に整理し、このうち、仕様検証を実現する単体検証、複合検証技術を明らかにしている。単体検証では、サービスの仕様入力、変換ステップに合わせて、階層的に検証を行なう階層的仕様検証法を提案し、そのうちの従来検討されていない、サービス手順の仕様検証方式について検討している。ここでの着想のポイントは、シーケンス図で記述されるサービス仕様の大きな流れをマクロに捉えて検証する、マクロ化シーケンス検証と、マクロな検証対象の各要素を構成する個々の細かい記述要素間の正当性を検証する遷移手順検証の2段階で検証することにより、検証システムの実現性が高まること、検証法の理解性を向上すること、及び誤りの重要度の高いものを早く検出することにより仕様修正の効率化が図られることが挙げられる。

複合検証では、個別に開発した通信サービス仕様複数動作時に発生する、通信サービス仕様間競合を事前に明らかにして防止するための、サービス競合検証方式を提案す



る。検証方式は、検証対象によりその特性に合わせて効率的に行なうことが望ましいため、検証要因対応の方式を明らかにする。また、カスタマイズドサービス提供においては、検証対象となるサービス仕様が膨大になると考えられるため、検証対象の限定手法についても述べる。

上記検証方式のうち、段階的検証方式については検証システムを試作し、検証能力を中心に評価する。

第5章では、自動作成された通信サービスソフトウェアに対して、実システム上で動作させる前に、要求仕様どおりの動作が実現されているかどうかを簡単にサービス動作を疑似体験して試験できる環境として開発した、ラピッドサービスプロトタイプシステム構成法について述べる。これを通信サービスソフトウェアの開発で使用することにより仕様レベルの誤りの検出、修正ができ、実システム上での試験を軽減できること、また仕様へのフィードバックを早期にできることにより、開発工数が削減されることを明らかにする。また、本システムを用いて作成したサービスのマン・マシンインタフェース仕様を評価する手法について述べる。これを現在実用化開発中の「高度化電話会議サービス」と「音声応答代行サービス」について適用し、お客様モニタによる評価実験を行い、マン・マシンインタフェース仕様の評価、改善を実現した。この結果より、本方式導入によるマン・マシンインタフェース仕様の改善効果を確認すると共に、電話系サービスのマン・マシンインタフェースの仕様設計におけるガイドラインを導出する。

第6章では、本論文で述べた通信サービスソフトウェア自動作成方式が、現在開発中の公衆網のインテリジェントネットワークにおけるサービス開発に果たす役割について具体的な技術を通じて示している。現在、上記技術を導入したIN用通信サービスソフトウェア自動作成システムを試作中であり、この上で各技術の有効性を確認していく予定である。また、本方式はINだけではなく、伝達網における階層化ソフトウェア構成を持つ交換機のサービスソフトウェアや、コンピュータで制御されるPBXにおけるサービスソフトウェアの自動作成技術としても適用可能である。

## 第2章 通信サービスソフトウェア自動作成方式の提案

### 2.1 まえがき

従来、通信ソフトウェアの開発工程の自動化については、製造、試験工程が中心に研究が進み、実用化レベルのツールが開発されている。しかし、設計工程のレベルでは自動化の検討は、研究としては盛んになりつつあるが、実用化レベルに到達しているものはほとんどない。これは、従来のアプローチが一般的、汎用的すぎること、また対象としての通信システムのソフトウェア構成の問題点をそのまま開発環境の中に取り込んでしまっていたこと等に起因している。また、通信ソフトウェア自動作成環境のユーザは、主に設計のエキスパートとしており、あくまでもエキスパートによる設計の補助、支援をするものという位置づけが一般的であった。このため、本論文で対象とする自動化には、従来技術がそのまま適用することはできない状況にある。

通信システムの制御対象のハードウェアを論理化して見せ、各サービスに共通な機能を汎用部品として切り出して、サービスに対応した部分のみのソフトウェア（通信サービスソフトウェア）を作成するだけで、自由にサービスを開発できるサービスプラットフォームを構成する方式に対する研究開発が盛んになってきている。インテリジェントネットワークがその代表的な例であり、伝達網の交換機やPBXの構成もこれにはほぼ類似の構成が検討されている。このような環境の下では、サービス対応の通信サービスソフトウェアに限定すれば、自動作成が可能状況が生まれてきた。

本章では、従来の通信ソフトウェア開発の問題点を挙げ、通信ソフトウェア開発自動化への課題を明らかにする。また、通信ソフトウェア開発を効率化し自動作成の前提となる、通信システムにおけるサービスプラットフォームの構成とその上で走行し個々のサービスを実現する通信サービスソフトウェアについて示す。この前提の基に、サービス開発の自動化を推進し、非エキスパートによるソフトウェア開発を実現する、通信サービスソフトウェア自動作成方式を提案する。本方式の実現技術として、仕様設計/ソフトウェア生成技術、仕様検証技術、仕様試験・評価技術の3つを明らかにし、それぞれの主要技術課題を抽出する。



## 2.2 通信ソフトウェア開発の問題点と解決法

### 2.2.1 通信ソフトウェア開発の問題点と解決のアプローチ

通信ソフトウェア開発における問題点は、第1章で示したように、大きく以下に示す7つの問題点がある。

- ・仕様記述モジュール構成と現実の通信システム構成とのギャップが大きい。
- ・リソース制御や実行制御等のOS的な機能とサービス制御のアプリケーションとしての機能が混在したソフトウェア構造である。
- ・高効率な多重処理を実現する通信用OSの不整備
- ・簡易で理解性の高い通信サービス仕様記述法が未確立
- ・効率的な仕様/プログラム自動変換法が未確立
- ・プロトコル仕様より上位のレベルの検証技術が未確立
- ・サービス仕様記述のみでサービス動作が実現できる環境がない

このうち、前者の3つは、通信システムのソフトウェア構造に起因する問題であり、2.2.2で詳しく述べる。新ソフトウェア構造を持つ通信システム導入により解決する。また、後者の4つは通信ソフトウェア作成の自動化に直接関連する問題であり、本論文で解決を図る。

### 2.2.2 新ソフトウェア構造通信システムの導入

次世代の新通信システムは、高効率な通信用OSであるCTRON<sup>®</sup>の適用を前提とし、この上に階層化ソフトウェア構造を持つ。通信用OS上に、リソース制御階層、その上にサービス制御階層を持つ。各階層は、以下の機能を持つ。

リソース制御階層：上位のサービス制御階層に仮想的リソース制御論理インタフェースを提供し、このインタフェースによる制御メッセージを受け、実リソースを駆動する機能と仮想リソースを管理する機能を持つ。

サービス制御階層：仮想的リソース制御論理インタフェースによる制御メッセージで構成される、サービス対応の制御ソフトウェアの実体とそれを解釈・実行する機能を提供する。

通信用OSとリソース制御階層及びサービス制御階層の制御ソフトウェアの解釈・実行機能を合わせたものは、サービス対応制御ソフトウェア共通の言い換えれば、サービス共通の動作基盤であり、以後本論文ではサービスプラットフォームと呼ぶ。インテリジェントネットワーク<sup>®</sup>では図2.1(a)に示すように、サービスプラットフォームの機能はNSP(ベルコアのSCPIに相当)とSAP(ベルコアのSSP、IPに相当)の一部に、伝達網の次世代の交換機であるオブジェクト指向形の交換機<sup>®</sup>では、図2.1(b)に示すように交換機内のサービス制御階層の解釈・実行機能とリソース制御階層がこれに当たる。このようなサービスプラットフォームを持つソフトウェア構造の通信システムを前提とすることで、サービス対応の開発は対応する制御ソフトウェアに限定されることになる。

### 2.2.3 新通信ソフトウェア構造における自動作成対象とその特徴

#### (1) 通信サービスソフトウェア

上述したサービスプラットフォームの中に配備される通信ソフトウェアは、一旦作り込めば実リソース(ハードウェア)の構成に変更がない限り、一般的には固定である。これに対して、サービスプラットフォーム上で走行する制御ソフトウェアは、サービス対応に作成され、新規サービス要求や機能追加等に合わせて追加・変更が頻繁に行なわれる(図2.2)。このため、この部分に対しては、自動化による開発工数の削減効果が高いと言える。このサービス対応の制御ソフトウェアを以下、特に「通信サービスソフトウェア」と呼ぶことにする。

#### (2) 通信サービスソフトウェアの特徴

通信サービスソフトウェアの特徴としては、主に以下の3点が挙げられる。

- (i) 実リソース(ハードウェア)構成、実リソース制御インタフェースの知識が不要
- (ii) 実行制御(多重処理)を考慮したソフトウェア記述が不要
- (iii) 通信サービスの新規要求、機能追加・変更は、本ソフトウェアのみの作成で実現可能

(i)は、通信サービスソフトウェアで意識するリソース制御のインタフェースは、仮想的な論理インタフェース構成としたためであり、これにより(i)で示す知識を持たない非エキスパートによる通信サービスソフトウェア作成が可能となる。



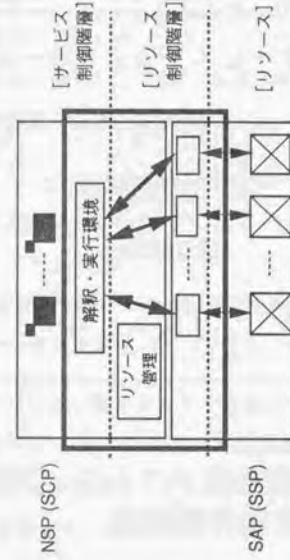
(ii) は、多重処理機能はサービスプラットフォームが提供しており、通信サービスソフトウェアはそれぞれ単一のプロセス構成で実現されるためである。これにより、汎用のプログラミング言語が適用でき、プログラム開発に汎用のツールで適用できることになる。

(iii) は、サービスプラットフォームの構成によるものであり、これにより簡単にカスタマのサービス要求を実現できるようになる。

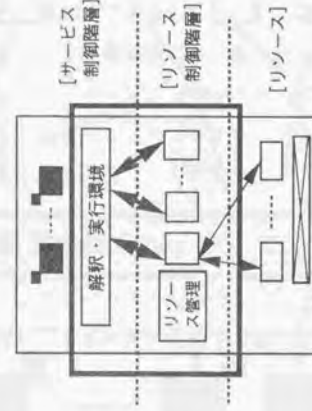
● 階層化ソフトウェア構造

- サービス対応制御ソフトウェアのためのプラットフォーム構成
- 仮想リソース制御論理インタフェース
- 高効率通信用OS、CTRONの適用

(a) インテリジェントネットワーク (IN)



(b) オブジェクト指向形交換機



■ : サービス対応制御ソフトウェア    □ : サービス対応制御ソフトウェア共通のプラットフォーム  
 ↔ : 仮想リソース制御論理インタフェース

図 2.1 新ソフトウェア構造通信システムの導入

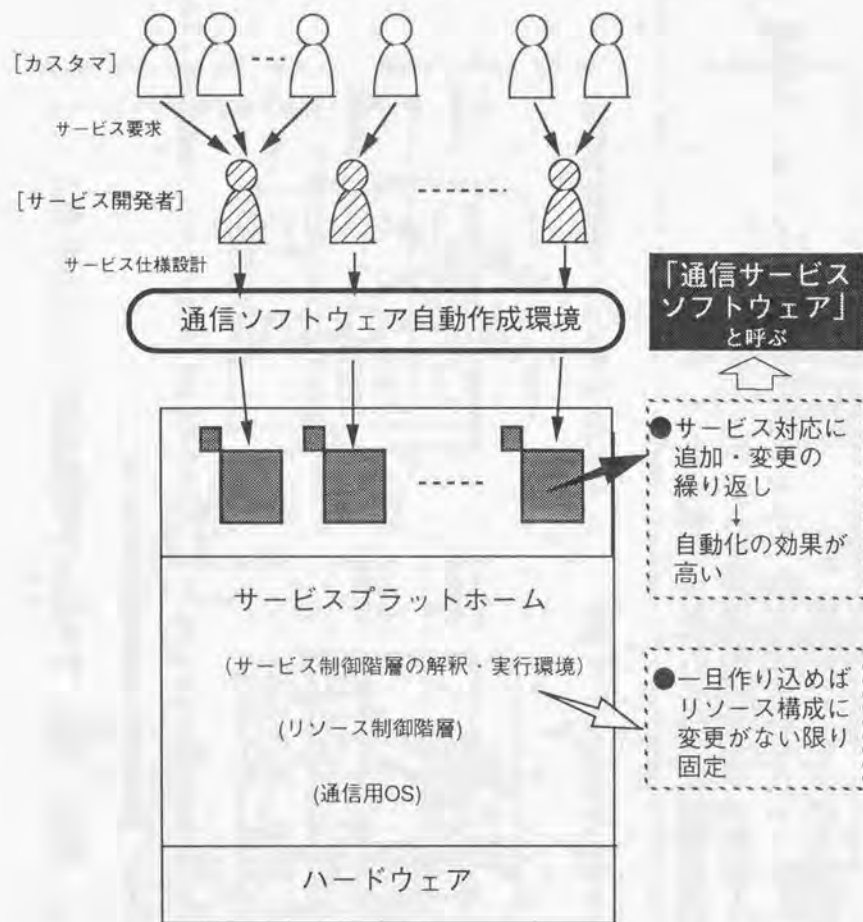


図 2.2 新通信ソフトウェア構造における自動作成対象

## 2.3 通信サービスソフトウェア自動作成方式の提案

通信ソフトウェア開発の自動化については、製造、試験工程が中心に研究が進み、実用化レベルのツールが開発されている。しかし、設計工程のレベルでは自動化の検討は、研究としては盛んになりつつあるが、実用化レベルに到達しているものはほとんどない。これは、従来のアプローチが一般的、汎用的すぎることで、また対象としての通信システムのソフトウェア構成の問題点をそのまま開発環境の中に取り込んでしまっていたこと等に起因している。前節で述べたように、サービスプラットフォームを持つ新ソフトウェア構造を持つ通信システムにおける、通信サービスソフトウェアの自動作成を対象としており、これにより設計工程の自動化を実用化レベルで実現できる可能性が生まれてきた。一方、いままでの通信ソフトウェア作成の自動化はあくまでもエキスパートの補助的支援ツールとして位置づけられていたが、非エキスパートに対する通信ソフトウェア作成の要求が高まっている現在、非エキスパートのための作成支援までをめざす必要がある。そこで、本論文では非エキスパートをその主な利用対象とした、通信サービスソフトウェアの自動作成方式を提案する。

### 2.3.1 非エキスパートによる通信サービスソフトウェア作成の必要性

通信サービスソフトウェアの開発は、前節に示したサービスプラットフォームの導入により、サービス対応の追加・変更要求に合わせて、頻繁に迅速に対応できる環境となり、開発の種類や規模自体がますます拡大すると考えられる。このような環境では、特に以下の点が強く要望されるようになる。

- (i) 多様化するカスタマ要求への即応
- (ii) 地域独自のサービス開発
- (iii) INにおいては、キャリアが提供するサービス機能要素を用いて、カスタマが自ら定義するカスタマイズドサービスを実現

(i)からは、支社や支店等のカスタマに近い所でカスタマ要求にきま細かく対応するため、現場に近い技術者による通信サービスソフトウェアの作成が必要となる。また、(ii)からは、地域の支社、支店の技術者による独自サービスの開発が要求される。さらに、(iii)からは、各支社、支店の技術者がカスタマに提供するサービス機能要素を作成することが必要となる。



一方、通信サービスソフトウェアの仕様設計やプログラミングのエキスパート（本論文では、このスキルの技術者をシステム開発者と呼ぶ）は、大部分がサービスプラットフォームの機能追加や変更の要員として割り当てられ、アプリケーションである通信サービスソフトウェアに割り当て可能な要員は、極めて少数であると予測される。このため、上述の要求条件は、通信サービスソフトウェアの仕様設計やプログラミングの非エキスパート（本論文では、このスキルの技術者をサービス開発者と呼ぶ）に対して求められることになる。この要求条件と非エキスパートのスキルとのギャップを埋めるためには、通信サービスソフトウェアの自動化は必須の技術と言える。

## 2.3.2 通信サービスソフトウェア自動作成方式の対象工程

### (1) 通信サービスソフトウェアの開発工程

通信サービスソフトウェアの開発工程を一般的なライフサイクルモデルで表現すると図 2.3 のようになる。図中の各工程の概要を以下に示す。

〔サービス要求定義工程〕：サービスの動作や属性に対する要求をユーザから見た仕様として、文章やアイコン等で定義する工程であり、プロダクトとして要求仕様書が得られる。

〔概要機能設計工程〕：要求仕様を実現するために、ユーザ・網間で必要となる情報（意味に着目した論理的信号）の送受をシーケンス図として記述する工程であり、概要機能仕様書が得られる。

〔詳細機能設計工程〕：概要機能仕様を実現するために、ユーザ・網間で必要となる信号の送受及び通信ノード間や通信ノード内の信号送受をシーケンス図やSDL図として記述する工程であり、詳細機能仕様書が得られる。

〔製造工程〕：詳細機能仕様を基に、プログラミング言語で記述しデバッグする工程であり、プログラム（通信サービスソフトウェア）が得られる。

〔試験工程〕：作成された通信サービスソフトウェアの正常動作をテストし、誤りを修正する工程である。

### (2) サービス開発非エキスパートのスキル

通信サービス開発の非エキスパートと一口にいってもそのスキルにおいては幅があり、これが自動化の範囲と密接に絡んでいるため、最初に本論文で対象とする非エキスパートのスキルレベルを明確にしておく必要がある。

本論文では、前述したように、通信サービスソフトウェアの開発を、支社、支店における技術者に期待している。このため、交換や通信サービスについての全くの素人は対象とは考えない。また、通信サービスの詳細な仕様設計やプログラミングのエキスパートも対象ではない。そこで、本論文では以下の点を、非エキスパートに期待するものとする。

〔通信サービスソフトウェア開発の非エキスパート〕

#### ① 交換や通信サービスの一般的な知識

- ・通信網の構成要素と機能、動作の概要
- ・通信サービスの手順
- ・信号方式の概要

#### ② ユーザ・網間の論理的な信号（情報）のシーケンス図で書かれた通信サービスの概要機能仕様に対する理解

- ・サービス仕様の意味
- ・サービス仕様の正しさ

#### ③ サービス仕様で陽に見える網内の処理手順

スクリーニング手順、タイマ、認証、課金等

②は、信号方式対応のプリミティブ（Q.931では、SETUPやCONNECT等）によるシーケンス図（信号シーケンス図）に対する知識は不要であることを示している。このため、さらに深い網内の信号やサービスプラットフォーム内のプロセス間の信号を表現した、SDL図についての理解は不要である。また、これらの仕様から得られる、プログラミング言語記述による通信サービスソフトウェアについての知識も当然不要となる。

③は、ユーザ・網間情報シーケンス図では記述できない部分であり、これがないと要求仕様が陽に表現されないため、通信サービスソフトウェアの自動作成の点からも必要な要素である。

本論文での対象非エキスパートは、上記レベルであるため、前述の概要機能設計工程が直接対応するレベルと言える。このため、非エキスパートであるサービス開発者の仕

様入力のレベルは、この概要機能設計工程と一段上のサービス要求定義工程となる。サービス要求定義工程でも仕様入力ができるが、サービス仕様の定義としては具体的な手順や網内の処理手順があいまいであり、カスタマ要求にきめ細かい対処ができないことから、本論文ではサービス開発者のスキルレベルである概要機能設計レベルで直接入力するものとする。一方、仕様を入力した後は、通信サービスソフトウェアまでを自動作成するが、変換途中での誤り検出や作成された通信サービスソフトウェアの試験の結果については、作成者である非エキスパートのサービス開発者のレベルまで引き上げて提示できることが望まれる。

### (3) 開発工程の対象範囲

上述の非エキスパートであるサービス開発者のスキルを前提とし、非エキスパートによる通信サービスソフトウェアの自動作成の対象とすべき範囲を考察する。ここで考慮すべき点は、以下の4つである。

- (i) 通信サービス仕様の入力は概要機能仕様のレベルで行なう。
- (ii) 非エキスパートに意識させるのは、入力レベルのみの仕様であり、それ以降の変換された仕様にさわることはない。
- (iii) 試験工程は、従来の通信ソフトウェア開発における試験工程の自動化技術が活用できる。
- (iv) 試験工程は、実機上で通信サービスソフトウェアを走行させての試験であるため、サービスプラットフォームの知識（エキスパートの知識）が要求される。

以上を考慮すると、非エキスパートのための通信サービスソフトウェアの自動作成では、図 2.3に網かけした範囲が対象とする工程であり、この自動化技術が本論文の検討対象となる。

### 2.3.3 通信サービスソフトウェア自動作成方式

今までに述べた点をまとめると、本論文で提案する通信サービスソフトウェア自動作成方式は次のように定義できる。

#### 通信サービスソフトウェア自動作成方式：

通信サービスの詳細設計やプログラミングの非エキスパートのサービス開発者を

対象とし、計算機支援により、ユーザ・網間の論理的な信号（情報）のシーケンス図で通信サービスの概要機能仕様を入力すると、詳細機能仕様に自動変換すると共に仕様の自動検証を実施し、仕様レベルの試験や評価法を提供することにより品質の高い仕様にリファインした後、プログラム（通信サービスソフトウェア）を自動生成することにより、良質な通信サービスソフトウェアを迅速かつ経済的に自動作成する方式。



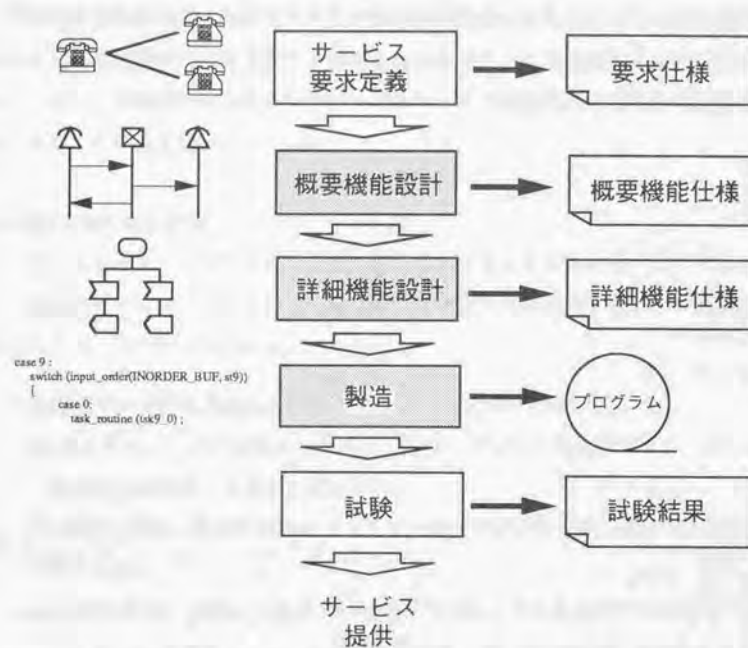


図 2.3 通信サービスソフトウェア自動作成方式の対象工程（自動化範囲）

□ : 本論文で対象とする範囲

## 2.4 通信サービスソフトウェア自動作成方式の構成技術

前節で示したように、品質の高い通信サービスソフトウェアを迅速かつ経済的に自動作成する方式を実現するためには、計算機支援による簡易な仕様設計を支援し、通信サービスソフトウェアを生成する機能、設計された仕様を自動検証する機能、及び生成された通信サービスソフトウェアの動作正常性や要求仕様との整合性を試験し、マン・マシンインタフェース仕様を動的に評価する必要がある。これより、図 2.4 に示すように、通信サービスソフトウェア方式を構成し、本論文の研究対象となる技術を大きく次の 3 つと捉え、以下に具体的技術課題を述べる。

- (a) 通信サービスの仕様設計/ソフトウェア生成技術
- (b) 通信サービス仕様の検証技術
- (c) 通信サービス仕様の試験・評価技術

### 2.4.1 通信サービスの仕様設計/ソフトウェア生成技術

サービスプラットフォーム上で走行させる通信サービスソフトウェアを自動作成するためには、サービス開発者が通信サービスの仕様設計を行なう必要がある。本論文で対象とするサービス開発者は、サービスの動作やユーザ・網インタフェースについては知識はあるが、網内の仮想リソース制御処理、言い換えるとサービスプラットフォームに関する知識はあまり持たない、いわゆる通信サービスの非エキスパートを主な対象としている。このため、非エキスパートでも簡単に通信サービス仕様の設計ができる環境が必須となる。また、この設計仕様からターゲットマシン上のサービスプラットフォームで走行する通信サービスソフトウェアを自動生成する必要がある。これらの点を考慮すると、ここでの主要技術課題は、以下の 3 つが挙げられる。

- 仕様記述法
- 仕様入力・編集法
- 仕様/プログラム自動変換法

これらの課題と本技術の構成イメージを図 2.5 に示す。

## 2.4.2 通信サービス仕様の検証技術

通信サービス仕様の検証は、設計したサービス仕様为正しく動作するだけでなく、ネットワークに既に存在するサービスに悪影響を与えないこと、通信リソースを無効保留しないこと等のために必要な技術である。これらの点を保証するために、通信サービス仕様設計時に仕様の構文や論理レベルの正当性を調べ、誤り部分を検出する技術が必要となる。

仕様の検証を実現する上でのポイントは、検証すべき対象をレベル別に分類し、それぞれのレベルで最も効率的に行われる方式を開発することである。一方、従来より通信サービスの仕様検証については多く検討されており、プロトコル手順の検証方式やリソース制御のための状態遷移手順の検証方式については、その成果を活用するため、本論文での検討対象とはしない。この下位レベルの検証結果はそのままでは非エキスパートが理解できないため、上位の仕様表現中に逆変換して見せる必要がある。また、上記方式は、通信サービス仕様単体の検証であり、非エキスパートが作成した通信サービス仕様間で発生する矛盾（競合）を検証する方式も必要となる。以上の点を考慮すると、ここでの主要技術課題は、次の4つが挙げられる。

- 階層的仕様検証法
- サービス手順検証法
- 上位仕様逆変換表示法
- サービス仕様間競合検証法

これらの課題と本技術の構成イメージを図 2.6 に示す。

## 2.4.3 通信サービス仕様の試験・評価技術

前述のサービス検証技術は、論理的な誤りを静的に検出するものであるが、静的な仕様検証だけで全ての誤りが検出されるという保証はなく、動的な面で発生する誤りは、実際に記述した仕様を動作させて調べる必要がある。また、サービスの要求仕様との整合性という、より深いレベルの検証は自動仕様検証では困難であり、実際にサービス動作を人間が確認することで初めて、サービスの要求仕様との整合性が保証できると言える。これらは、従来の通信ソフトウェア開発における試験工程のうち、通信サービス仕様に関するものの試験工程であると言えるが、実際のターゲットマシン用のソフトウェ

アの製造の前に試験を行なう点が新しいと言える。これは、情報処理の分野で最近盛んになっている、ソフトウェアのラビッドプロトタイピング技術<sup>[11]</sup>と言えるものであり、通信サービスソフトウェアに対するラビッドプロトタイピング方式を、本論文では新たに開発する。

上記の試験は、動作の正常性や要求仕様との整合性といった、仕様の正/誤を判断することが主な目的であるが、サービスの品質は正しいだけでは不十分であり、使い易さやサービスとしての有効性や利便性についても事前に調べる必要がある。これは、試験技術に対して、通信サービス仕様の評価技術と呼ぶことにする。評価の対象のうち、サービスとしての有効性や利便性は、あくまでも主観評価や政策的、戦略的なもので評価が決まることから、ここでは技術的な評価可能な使い易さを示す、通信サービスのマン・マシンインタフェース仕様を対象とする。通信サービスのマン・マシンインタフェース仕様の評価は、サービス開発者だけでなく、実際にサービスを使用するユーザ自身の評価ができることが最も望ましく、上述のラビッドプロトタイピングシステムを適用したモニタ評価法の実現を検討する。また、この評価の結果は、対象とした通信サービス仕様効率に効率的にフィードバックさせる必要がある。

以上の点を考慮すると、ここでの主要技術課題は、図 2.7 に示すように、次の5つが挙げられる。

- サービス仕様の試験環境構成法
- マン・マシンインタフェース仕様のモニタ評価法
- マン・マシンインタフェースの評価尺度・分析手法
- モニタ評価結果のフィードバック法
- マン・マシンインタフェース仕様設計のガイドライン構成法



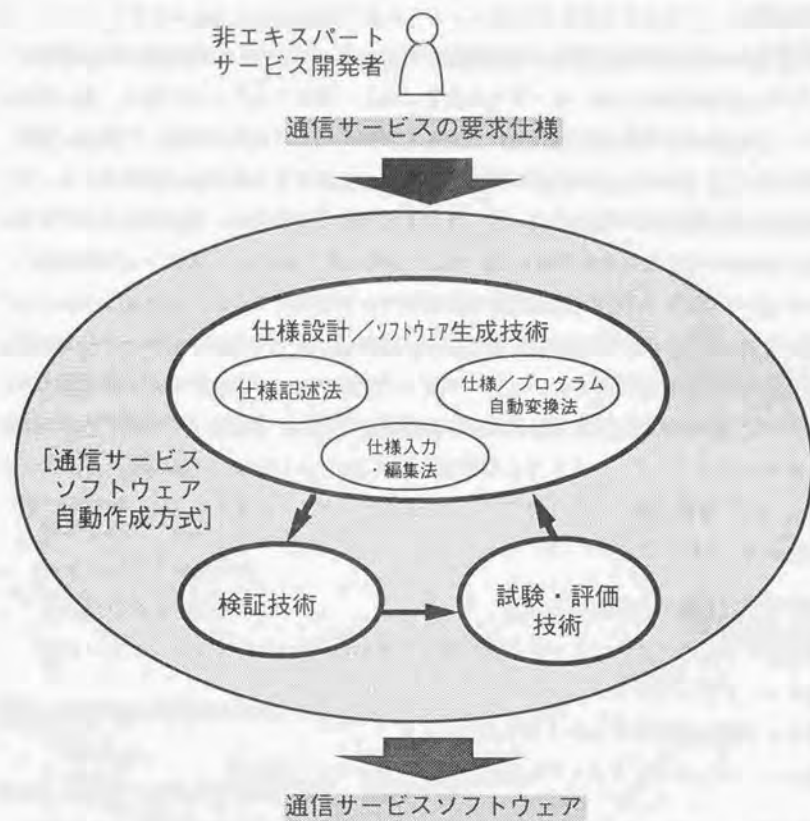


図 2.4 通信サービスソフトウェア自動作成方式の構成技術

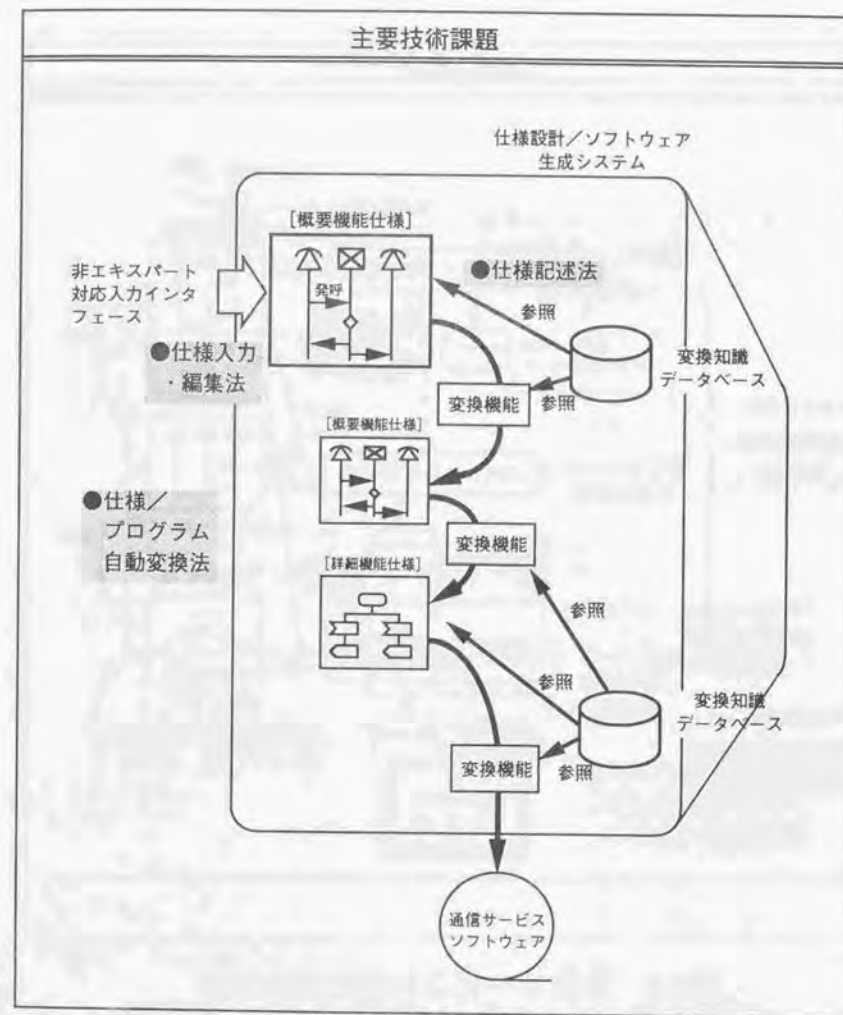


図2.5 仕様設計/ソフトウェア生成技術

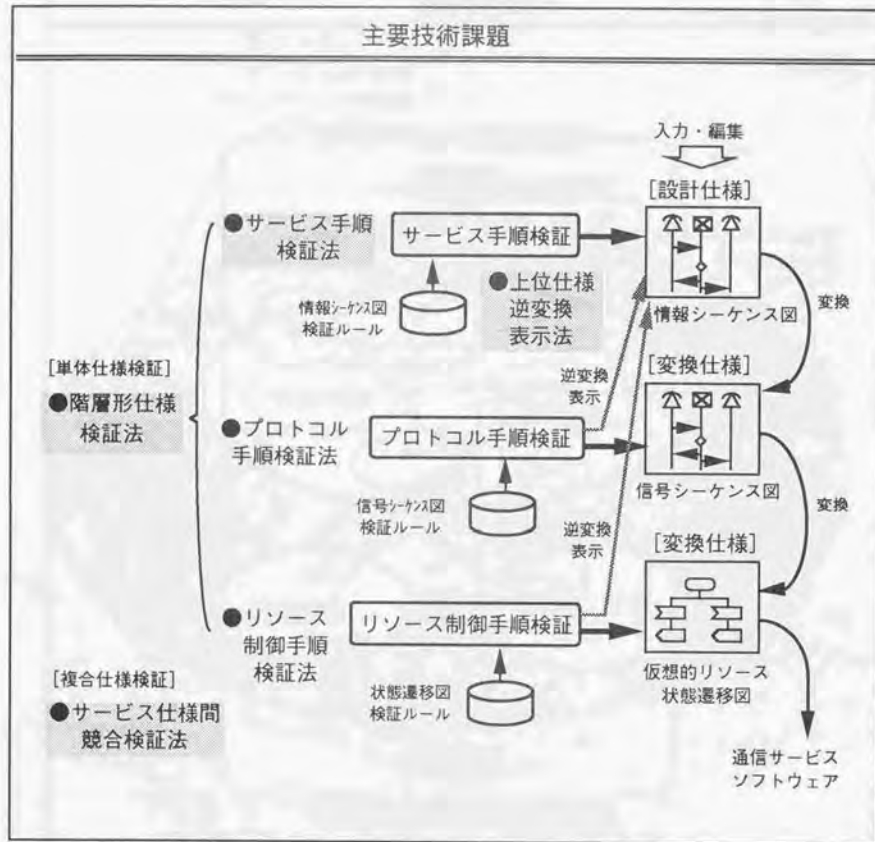


図2.6 通信サービス仕様の検証技術

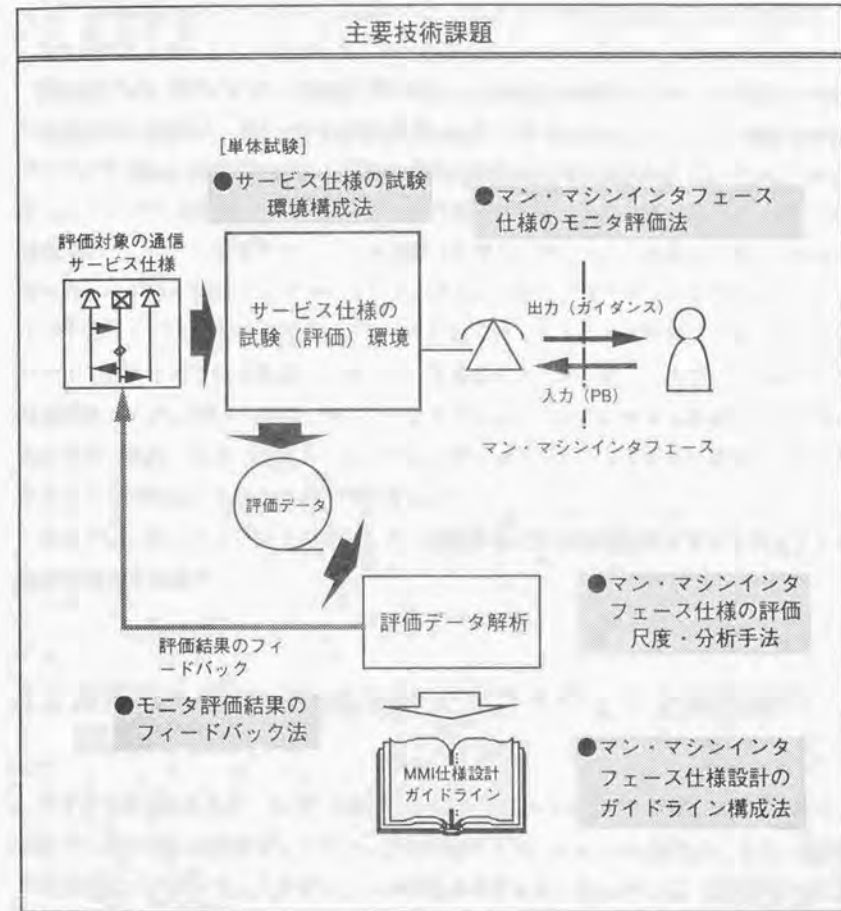


図 2.7 通信サービス仕様の試験・評価技術



## 2.5 むすび

多様化するユーザ要求に迅速に対応させたサービス提供を実現するため、通信ソフトウェア開発の現状と問題点を明らかにした。この問題点を解決するため、新通信ソフトウェア構造を前提とした通信システム上で走行する、アプリケーションサービス対応の通信サービスソフトウェア開発を自動化するための、通信サービスソフトウェア自動作成方式を提案した。この方式に対して、主要な技術課題を洗い出し、本研究での検討対象を明らかにした。検討対象に対する詳細な研究内容は、第3章以降に示す。

## 第3章 通信サービスの仕様設計／ソフトウェア生成技術

### 3.1 まえがき

第2章では、階層化ソフトウェア構造で、サービス対応制御ソフトウェアのためのプラットフォームを持つ、通信システムを前提とした、通信サービスソフトウェアの自動作成方式を提案し、通信サービスの開発が簡単かつ迅速に実現される見通しを示した。しかし、ここでの通信サービスソフトウェアの記述は、あくまでもプログラミング言語が前提であり、さらに仮想的なリソース制御の論理インタフェースを陽に意識した形式であった。これは、通信サービスソフトウェアは、設計、プログラミングの非エキスパートであるサービス開発者が作成するという前提を考えると大きな問題となる。非エキスパートであるサービス開発者は、サービスの動作やユーザ・網インタフェースについては知識はあるが、網内の処理、特にサービスプラットフォームに関する知識はあまり持たないのが一般的であると言える。このため、非エキスパートでも簡単に通信サービスソフトウェアが作成できる方式が必須となる。

本章では、非エキスパートを指向した、通信サービスの仕様設計／ソフトウェア生成技術の確立を目指す。

### 3.2 通信サービスの仕様設計／ソフトウェア生成技術の概要

第2章で示したように、本章では通信サービスソフトウェア自動作成方式を構成する、通信サービスの仕様設計／ソフトウェア生成技術を明らかにする(図3.1)。まず、最初に仕様設計／ソフトウェア生成システム構築の設計指針を明らかにし、以下の3つの主要課題について実現技術を明らかにする。

- (1) 仕様記述法
- (2) 仕様入力・編集法
- (3) 仕様／プログラム自動変換法

また、具体的実現技術と共に、これらの技術による通信サービスソフトウェア作成環境の実現性、有効性を把握するために、試作システムを構成し、評価を行なう。

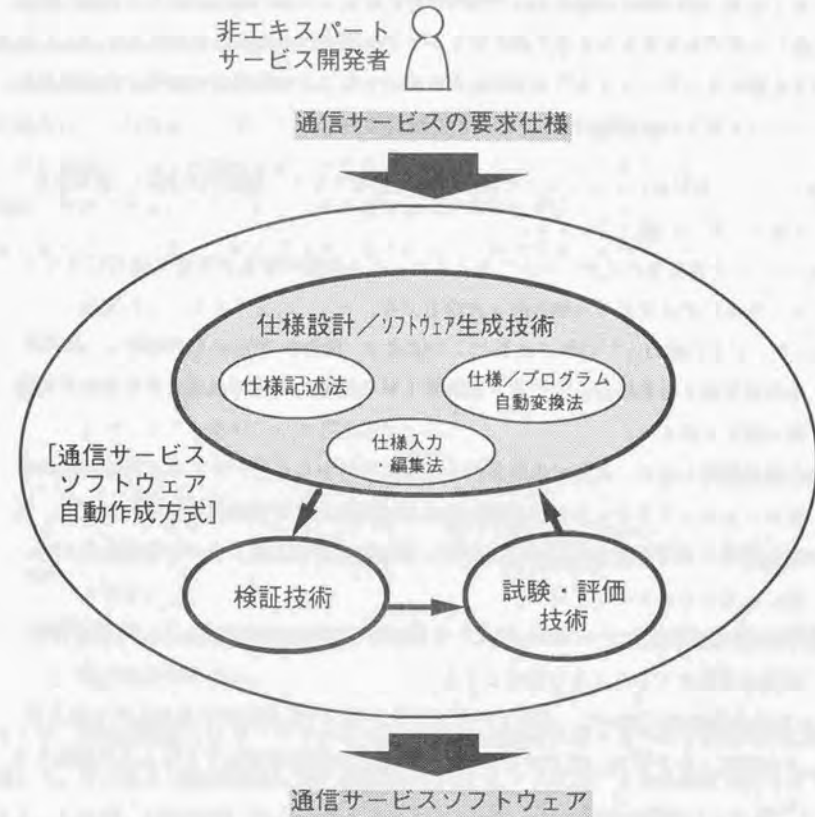


図 3.1 通信サービスの仕様設計/ソフトウェア生成技術



### 3.3 仕様設計/ソフトウェア生成システム構築の設計指針

非エキスパートを指向した通信サービスの仕様設計/ソフトウェア生成システムとは、言い換えると、サービスに対するユーザの要求仕様をユーザの持つ通信サービスに対する知識レベルで入力するだけで、ターゲットとする通信システムのプラットフォーム上で走行する通信サービスソフトウェアを生成可能にすることを意味している。この観点に基づき、本システムの設計指針を以下のように定義する。

- (i) サービス開発者にとって入力の抵抗感が比較的少なく、理解性の高い、図形式によるサービス仕様入力とする。
- (ii) サービス開発者の入力レベルで見えるサービス機能の要素が新規に追加になっても、簡単に記述要素が追加可能な構成とする。
- (iii) サービス仕様の入力の際に必要な知識は、極力システムより提供し、選択的な処理や準正常処理等に関する予備知識学習の軽減や知識不足による仕様の不足箇所の発生を抑える。
- (iv) 変換知識を基に、入力・編集したサービス仕様からプログラミング言語記述の通信サービスソフトウェアへ、変換知識を基に段階的に自動変換される。ただし、変換過程でのより詳細な設計仕様は、サービス開発者のスキルレベルを越えるため、陽には見せないものとする。
- (v) 仕様の変換知識はデータベース化して機械的に利用できる形式とし、知識の追加、変更が簡単にできるような構成とする。
- (vi) 自動仕様変換において、通信サービスソフトウェアの動作時に欠如していると大きな問題となり得る、準正常の手順については自動的に付加できるような構成とする。
- (vii) 自動仕様変換において、仕様の冗長性に強く影響する要因については、最適化が図れる構成とする。

以下、本設計指針に基づき、仕様設計/ソフトウェア生成システム構築の各技術課題における具体的実現技術を明らかにする。

### 3.4 通信サービスの仕様記述法

#### 3.4.1 非エキスパートのための仕様記述法

従来、通信サービス仕様の記述には、CCITTで勧告されている仕様記述言語SDL<sup>[13]</sup> (Specification Description Language) やEstelle<sup>[14]</sup>、OSIで勧告化されているLOTOS<sup>[15]</sup>及び現在、CCITTで勧告化が検討されているシーケンス記述<sup>[16]</sup>がある。これらの特徴を特に利用目的の観点から比較したものを表3.1に示す。サービス仕様表現の特徴は、各ユーザと網間の1対1の関係を全て示すのではなく、全ユーザ及び網間を横通しに見た表現が一般的である。このため、この特徴に適した仕様記述法は、シーケンス記述と言える。さらに、シーケンス記述の良さをまとめると、以下の6つがあげられる。

- ・視覚的にわかりやすく、サービス仕様の理解性が高い。
- ・システム内部の動作(状態)の知識が不要。
- ・サービス仕様のシーケンス記述は、支社、支店レベルの技術者で開発の非エキスパートでも慣れている記述法である。
- ・使用するメッセージ(プリミティブ)の抽象度を上げることで、機能仕様の上位レベルが記述可能。
- ・CCITTのサービス記述のガイドラインに準拠した仕様記述が可能(以下に詳細を示す)。
- ・CCITT SG-Xで、現在メッセージシーケンス図(MSC)による仕様記述法の標準化が検討中である。

そこで、本作成環境ではサービス仕様作成において最低限必要と思われる情報のみに限定して、サービスの概要機能仕様を表現可能なシーケンス記述による仕様記述法を提案する。これは、CCITTのI.130<sup>[17]</sup>で勧告された以下のサービスの記述手順のうちの、ステップ1にあたるもので、情報シーケンス記述と呼ばれるものを基にしている。ただし、この記述法では網内の処理手順でサービス仕様に関与する処理手順は表せないため、この処理手順の記述も含めて表現可能な拡張形情報シーケンス記述法を新たに開発し、非エキスパートのための仕様記述法とする。

(ステップ1) 拡張形情報シーケンス記述: ユーザと網間の信号方式に非依存な論理的な信号送受の手順と網内の処理手順で表現。記述者のスキルは、交



換サービスに関する基礎知識はあるが、信号方式（プロトコル）の知識までは持たないものを対象とする。

仕様記述は、この拡張形の情報シーケンス記述で行なうが、通信サービスソフトウェアへの自動変換の過程では、以下の各ステップに段階的に変換されるものとする。これを図 3.2に示す。

（ステップ2）信号シーケンス記述：ユーザと網間のプロトコルに依存する物理的な信号と網内での処理概要で表現。

（ステップ3）仮想リソース制御状態遷移記述：網内のサービスプラットフォーム上における仮想リソースの制御手順を表現。

### 3.4.2 メッセージシーケンス記述言語 MSDL

上述の拡張形情報シーケンス記述法に基づき、メッセージシーケンスによりサービス仕様を表現する手法として、メッセージシーケンス記述言語 MSDL (Message Sequence Description Language) を開発した。MSDLは、図形式 (MSDL/GR) と言語形式 (MSDL/PR) の両方を持ち、記述者のスキルに応じて選択できるようにした。しかし、一般的にはサービス仕様の記述者は非エキスパートであり、より簡単でイメージが得られやすい、図形式を基本に使用する。

シーケンス図記述に要求される主な記述要素は、以下のように分類できる。

- (i) ユーザ・網間のメッセージ
- (ii) リソース状態による分岐
- (iii) ユーザの振る舞いによる分岐
- (iv) 網内の処理機能
- (v) 通信中状態

これを表現するためのMSDLの記述要素を表 3.2 (a)に、図形式と言語形式を対応させて示す。また、MSDLのシンタックスダイアグラムを図 3.3に示す。

上記の記述要素が一般的に記述されるシーケンス図の要素であるが、ここではより記述能力をあげるため、図 3.4に示すような三者通話等における、同じ手順の繰り返しを

記述するものと制御の終了を明示する記述要素を導入した。繰り返し記述には、制御の移行を示すため、表 3.2 (b)に示す制御の移行元 (next) と制御の移行先 (label) を表現する2つの記述要素 (図形式では  $\curvearrowright$  と  $\curvearrowleft$ ) を設けた。また、制御の終了は、同表に示す、stop 記号 (図形式では  $\bullet$ ) により表現する。

MSDLを用いて、ステップ1の拡張形情報シーケンス図を記述した例を、図形式と言語形式を比較して図 3.5に示す。

### 3.4.3 網内処理機能の仕様記述

情報シーケンス記述を拡張させるためには、上記MSDLの記述要素のうちの「網内の処理機能」について、その中身をサービス仕様に合わせて、大まかな処理手順として記述する必要がある。網内の処理機能としては、主として以下のものがあげられる。

- ・番号翻訳
- ・認証
- ・スクリーニング
- ・タイマ
- ・待ち行列
- ・課金

これらを全て個別の記述要素で構成すると、一部機能を変更・追加するたびに、これとは別に新たに記述要素を設定しなければならないため、できるだけ共通機能をくくりだし、共通機能と固有機能から構成される記述要素を以下のように構成した。

- ①データの演算：四則演算、論理演算等
- ②データの検索：特定のテーブル内の指定要素を抽出
- ③データの比較：特定の参照値との比較 例)  $>$ 、 $<$ 、 $=$
- ④データの代入：特定のデータバッファへの書き込みやテーブル内の指定要素の更新
- ⑤呼の分配：呼を各種設定条件 (スクリーニング条件) により、異なる複数の処理手順に配分 例) 割合、時刻、曜日、ユーザ設定最大呼数
- ⑥ループ：一連の処理機能の繰り返し
- ⑦タイマ：ある指定時間をタイマに設定し、タイマ満了時に通知



⑧時刻の取得：現在の時刻（年月、曜日等も含む）を取得

上記記述要素の図形式表現を表 3.3 に示す。また、上記記述要素の組み合わせで網内の処理手順を記述するためには、MSDL とのデータの引き継ぎやデータテーブルのアクセスをするための以下の記述要素が必要となる。これを、表 3.4 に示す。

- (a) 入力データ
- (b) 既取得データまたは出力データ
- (c) テーブル内アクセスデータ
- (d) 戻り値データ

これらの記述要素を用いた網内処理機能の記述例を、図 3.6 に示す。

### 3.4.4 拡張形情報シーケンス記述法適用困難なサービスの対処法

#### (1) 拡張形情報シーケンス記述法適用困難なサービス

ユーザと網とのメッセージのインタラクションに着目したサービスのほとんどは、他の呼への割り込みが発生せず、ひとつの呼に現われるユーザ数が常に決まっているので本仕様記述法のようなシーケンス図形式で容易に記述できる。しかし、(a)他の呼への割り込みが発生するサービス(例：コールウェーティングサービス(以下CWと表記))と(b)ひとつの呼に現われるユーザ数があらかじめ決めることができないサービス(例：電話会議サービス)は、次の理由によりシーケンス図形式では一般的に記述しにくい。

##### (a)他の呼への割り込みが発生するサービス：

CW を例にとるとユーザAとユーザBが通話中である事象と、ユーザCがユーザAに発信する事象は独立であるのに、ユーザAがユーザCから着信を知り、通話を切り替えることにより2つの事象が統合されるので、図 3.7 (a) のように記述すると、ユーザCの発信が必ずユーザAとユーザBの通話中である(厳密にはユーザBが応答した後)であることになる。

##### (b)ひとつの呼に現われるユーザ数があらかじめ決めることができないサービス：

呼に参加する人数が最大数しか規定されず、全ての場合を図 3.8 (a) のように網羅的に記述するとシーケンス図はユーザ数とユーザの応答順序による組み合わせ毎に異なり、

最大  $n$  名のメンバを呼出して会議を始める場合ではメンバーの応答順序だけで、 $\sum_{k=1}^n k!$  とおり必要となる。

#### (2) 拡張形情報シーケンス記述法の改善による記述対象の拡大

以下の手法を用いることにより、上記2つのサービス記述が可能となる。

##### (a)CW：

ユーザCがユーザAの話中に遭遇した(すなわちCWが起動される)場合、CWのための交換ノードの内部処理"CW開始(Start\_CW)"を実行し終了する。"CW開始(Start\_CW)"が実行されると、それ以降の処理は、ユーザA-ユーザBの呼にある"CW起動(Invoke\_CW)"が引き継ぐと考える(図 3.7 (b))。

##### (b)電話会議サービス：

会議のホスト(主催者)をユーザA、複数のメンバーをユーザX、メンバーのうち特定の扱いを行なう者をユーザBとして表現する。交換ノードからXへの情報送信はメンバー全員に対するものと解釈し、ユーザXから交換ノードの情報受信はメンバーのうちの誰かからと解釈する(図 3.8 (b))。メンバーを追加する場合、ユーザBを呼び出し、ユーザBを参加させるときにユーザBをユーザXのメンバーとする内部処理"BX変換"を実行し、それ以降をユーザXとして扱う。もし会議中にメンバーの誰かを特定するには、ユーザXからユーザBへの移行を行なう内部処理"XB変換"を実行し、以降の情報送受はユーザBに対して行なう。(図 3.8 (c))。

CWと電話会議サービス以外のサービスでも同様な手法を使用できると考えられるため、本サービス仕様記述法を用いることにより、ユーザ・網間のメッセージインタラクションで構成される大部分のサービスが記述できると考えられる。

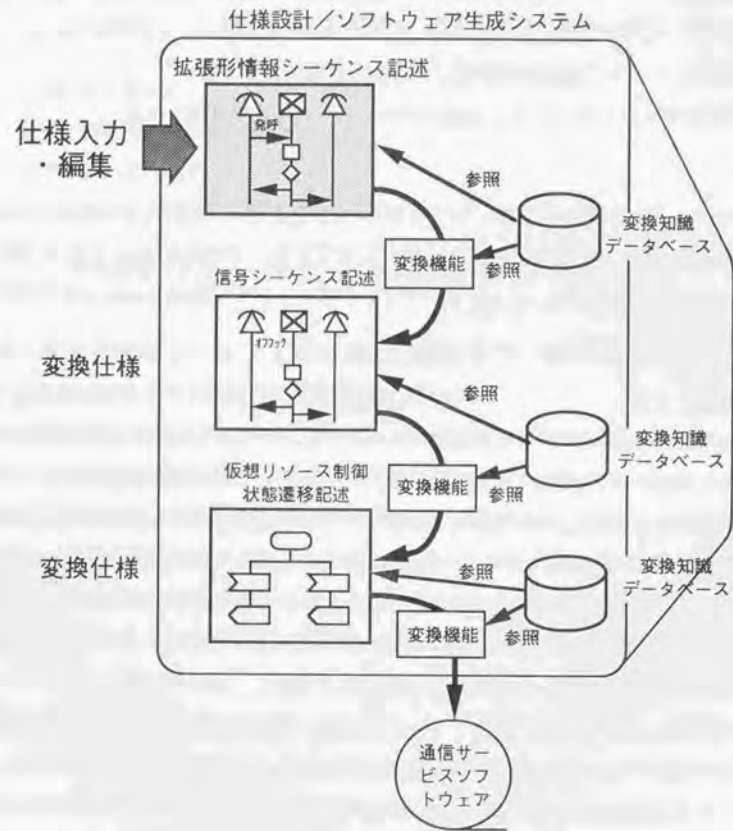


図 3.2 拡張形情報シーケンス記述法

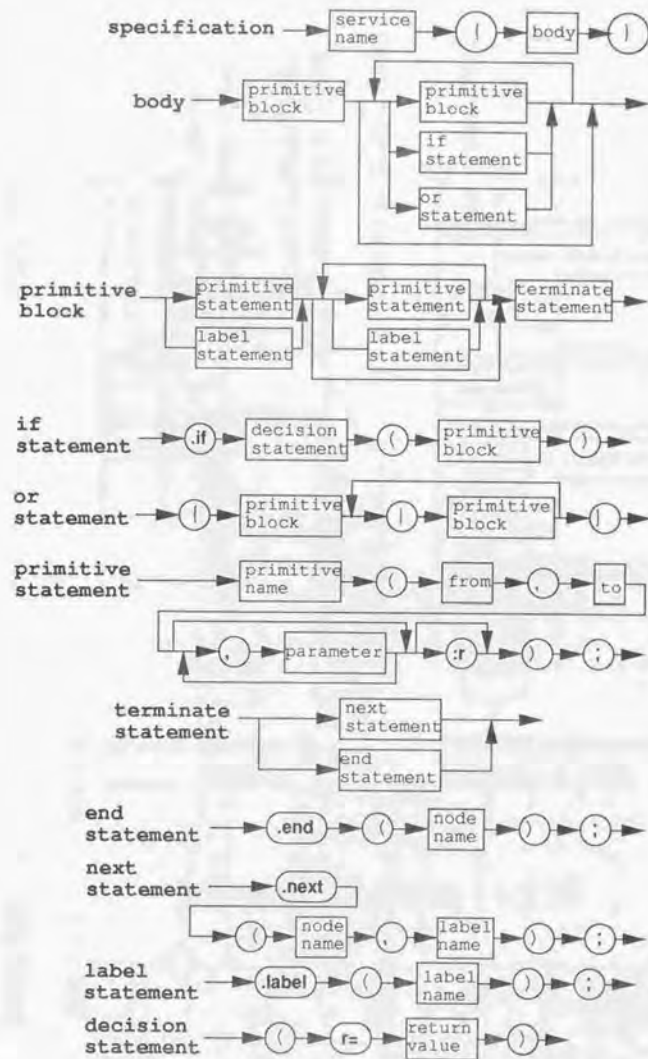
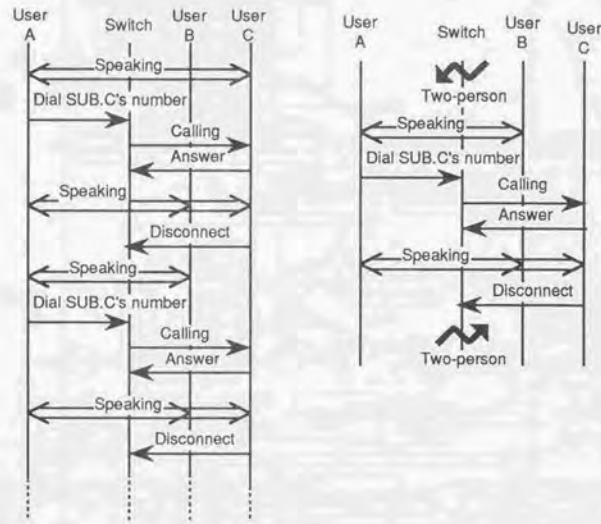


図 3.3 MSDL シンタックスダイアグラム





(a) next-label 記述要素を使用しない仕様記述

(b) next-label を用いた仕様記述

図 3.4 三者通話の仕様記述

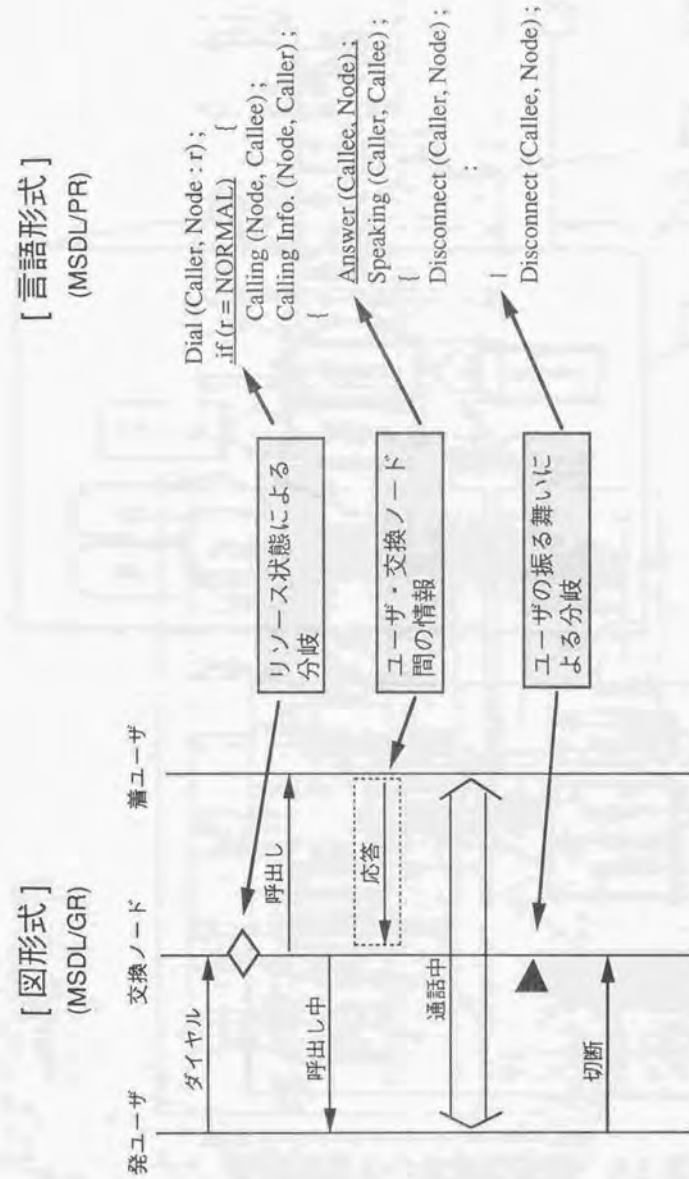


図 3.5 ステップ 1 記述例 (MSDL)

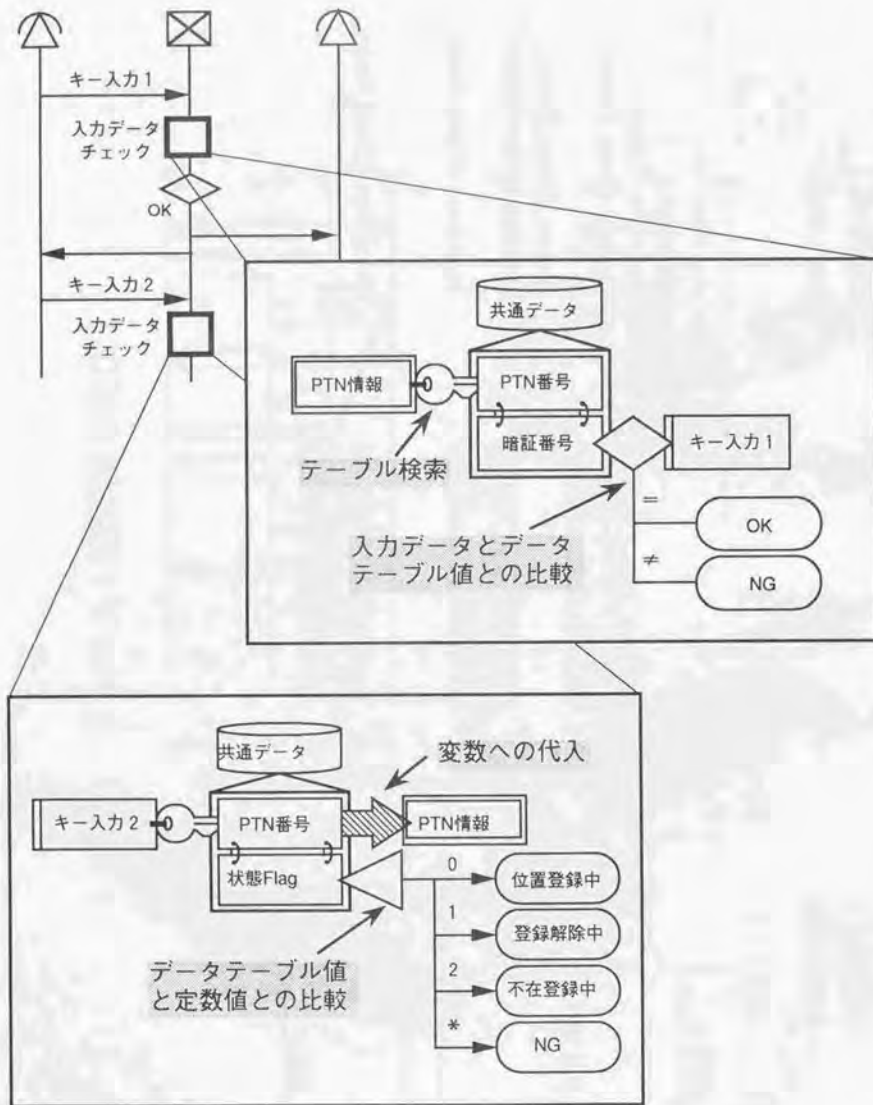


図 3.6 ネット処理機能の仕様記述例

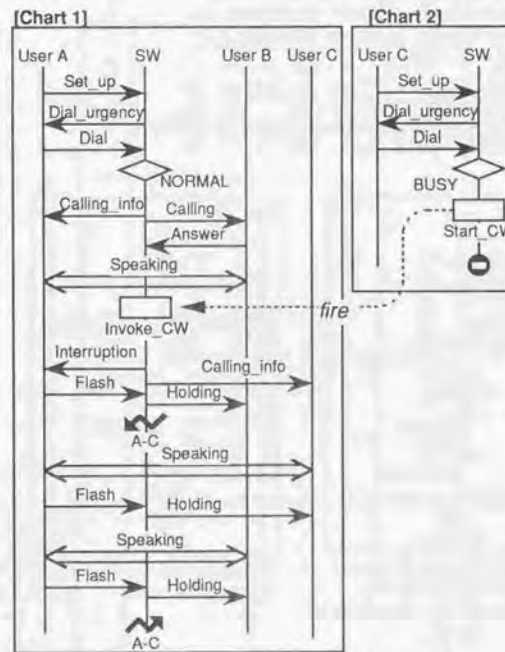
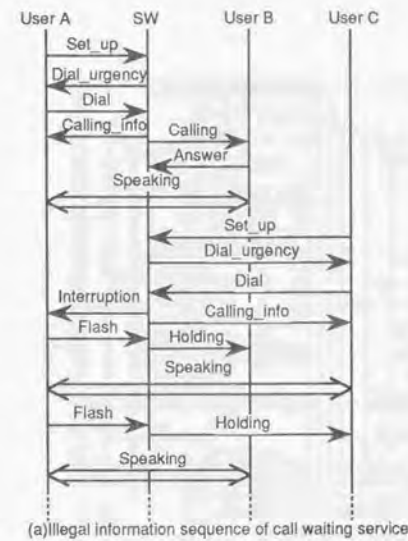
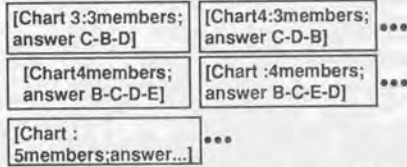
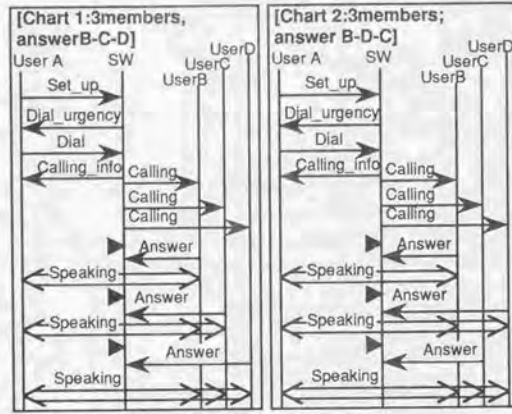
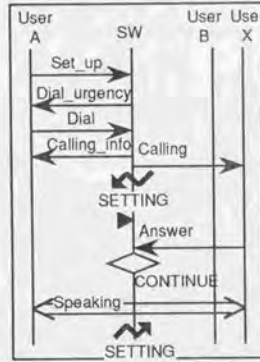


図 3.7 コールウェイティングサービスの仕様記述

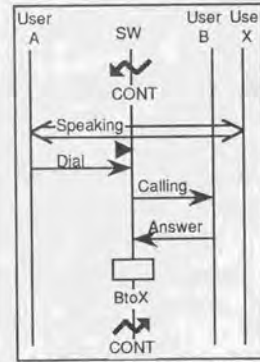




(a) Whole information sequence chars required setting up tele-conference service



note: Branch "CONTINUE" means some members are still calling.  
(b) Condense using "Subscriber X"



(c) Add Special member "User B" to the conference






図 3.8 電話会議サービスの仕様記述

表 3.1 仕様記述法の比較

	LOTOS	SDL、Estelle	シーケンス記述
記述目的	システムの外部から見た入出力信号 (アクション) の時系列で表現	複数のプロセスで構成されるシステムにおいて、各プロセスの状態遷移を表現	システムの全構成要素間を横通しに見て、各構成要素間のメッセージの時系列で表現
目的に適した適用対象	プロトコルの厳密な仕様記述	有限状態機械でモデル化される通信ノードの処理記述	ユーザ・網間のような大局的な動作の流れを記述

表 3.2 MSDL の記述要素

(a) 基本記述要素

意味 (説明)	図形式表現	言語形式表現
(i) ユーザ・網間のメッセージ (AからBへのメッセージ。A,Bの一方がユーザ、他方が交換ノード。nameはメッセージ名)		name (A, B, parameter...);
(ii) 交換ノードのリソース状態による分岐 (Bは交換ノード。nameは処理機能名)		.if(r=cause) ( )
(iii) 交換ノード内の処理機能 (Bは交換ノード。causeは分岐要因)		name (B, NULL, parameter...);
(iv) ユーザの振る舞いによる分岐 (Aがユーザ、Bが交換ノード。nameは分岐要因となるメッセージ名)		{name...[...]}
(v) 通信中状態 (A,Bは。nameは通信中状態を特定する名前)		name (A, B);

(b) 付加記述要素


意味 (説明)	図形式表現	言語形式表現
● 制御の移行元 (Bは交換ノード。nameはラベル名を持つラベルの箇所に制御が移る)		.next (name, B);
● 制御の移行先 (Bは交換ノード。nameはラベル名)		.label (name);
● 制御の終了点 (Bは交換ノード)		.stop (B);

表 3.3 網内処理機能の基本記述要素


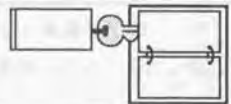


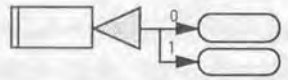
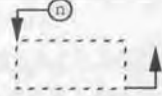
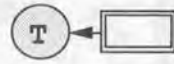
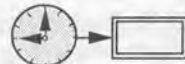



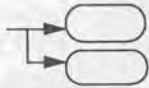
意味 (説明)	図形式表現
①データの演算 四則演算、論理演算等	
②データの検索 特定のテーブル内の指定要素を抽出	
③データの比較 特定の参照値との比較	
④データの代入 特定のデータバッファへの書き込みやテーブル内の指定要素の更新	
⑤呼の分配 呼を各種設定条件により、異なる複数の処理手順に分配	
⑥ループ 一連の処理機能の繰り返し	
⑦タイマ ある指定時間をタイマに設定し、タイマ満了時に通知	
⑧時刻の取得 現在の時刻を取得	



表 3.4 網内処理機能の付加記述要素

意味 (説明)	図形式表現
(a) 入力データ	
(b) 既取得データまたは出力データ	
(c) テーブル内アクセスデータ	
(d) 戻り値	

### 3.5 仕様入力／編集法

#### 3.5.1 インテリジェントエディタ(IED)の構成

非エキスパートがサービス仕様を入力する場合、できるだけ負担の少ない方式が望ましいのは言うまでもないことである。図形式の表現は親和性、理解性は高いが、言語形式のように入力が簡単ではない点が問題である。単なる図形式のエディタでは、相当な労力となるため、ここでの入力に完全に機能を限定して適合させ、ワークステーションの持つ高機能なマン・マシンインタフェースを活用した操作性の高い入力を支援するエディタを提案する。ここでは、本エディタが高機能で、さらに次で示す知識を用いた入力支援を実現することより、インテリジェントエディタ (Intelligent Editor、以下IEDと略す) と呼び、拡張形情報シーケンス図を記述するために拡張形情報シーケンス図 IEDを開発した。以下、拡張形情報シーケンス図 IEDを構成するため、新たに開発した実現技術を示す。

#### 3.5.2 データベースを用いたサービス仕様入力法

IEDで仕様の入力／編集に、システムに予め作り込まれているユーザ・網間のメッセージ、網内の処理機能および通信中状態やかつて使用したものを再び使用するためには、これらに関する知識を登録しておくデータベースが必要である。この情報シーケンス図の上記述要素に関する知識を登録しているデータベースを情報定義データベースと呼ぶ。データベースの構成については次節で詳細に述べるが、サービス仕様入力では、記述要素の名称と取り得る方向の2つの知識が必要となる。

IED上では、図 3.9に示すように、データベースに登録されている記述要素をメニュー形式で常時表示しており、記述要素表示メニューから記述要素を選択し、記述要素名表示メニューから1つの名称を選択の後、シーケンス図編集ウィンドウ上の位置を指定するとその記述要素と名称が書き込まれる。

リソース状態による分岐は、分岐発生箇所および分岐条件の知識を上記データベースに予め埋め込み、仕様入力途中で自動的に分岐記号および分岐条件を挿入する。これを図 3.10の上半分に示す。

#### 3.5.3 画面制御による仕様編集支援

網リソースによる分岐条件を表示後、1つを選択することにより、対応する条件のシ

号を指示することにより、移行先（飛び先）のシーケンス部分を自動的に画面を切り替えて表示する。前者の例を図 3.10 の下半分に示す。

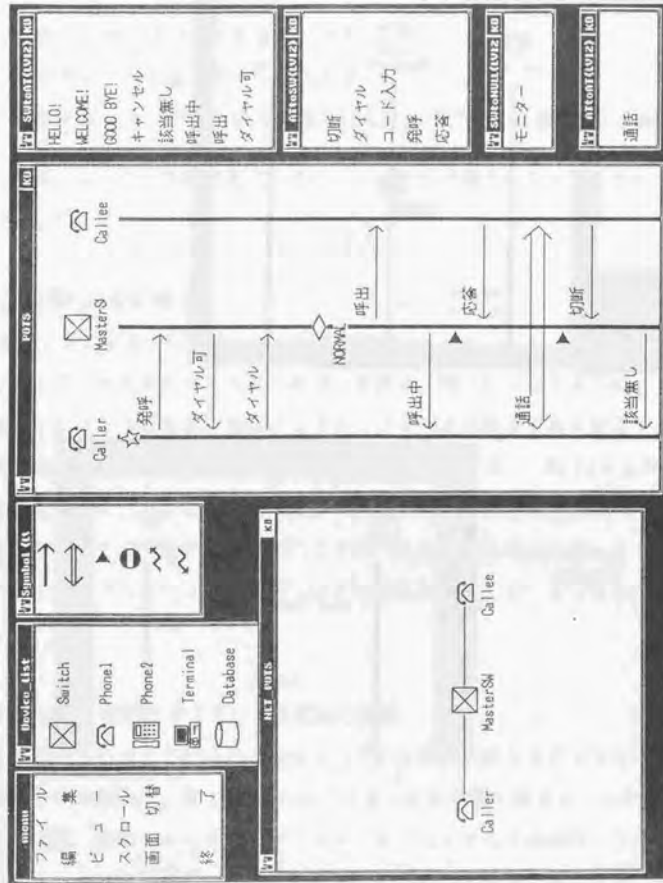


図 3.9 拡張形情報シーケンス図 IED 表示例



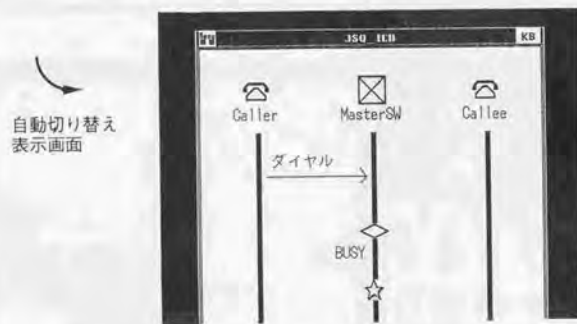
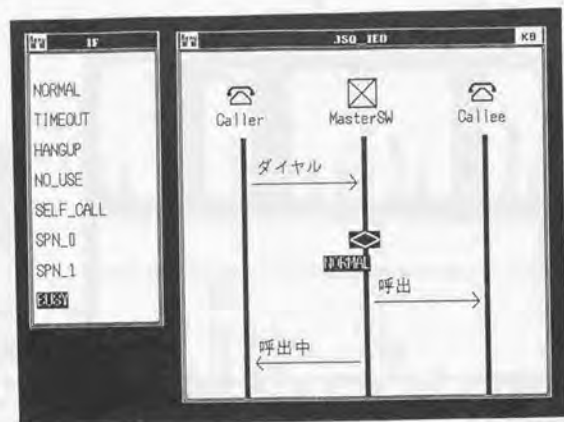


図 3.10 分岐条件の自動表示及び自動切り替え表示

### 3.6 仕様／プログラム自動変換法

#### 3.6.1 変換知識データベースの構成法

情報定義データベース、信号定義データベース共、以下の知識で構成する。

- (a) ユーザ・網間メッセージ、網内の処理機能、通信中状態の名称
- (b) メッセージの取り得る方向
- (c) 次のステップの記述要素との対応
- (d) パラメータの種類とデフォルト値
- (e) 網がメッセージ受信時の判断分岐条件（リソース状態による分岐条件を含む）

上記知識について、情報定義データベース、信号定義データベースそれぞれの例を図 3.11 に示す。

#### 3.6.2 自動仕様変換法

##### (1) ステップ1記述からステップ2記述の変換

ステップ1記述からステップ2記述の変換は、図 3.12 に示すように、基本的には情報定義データベースの内容に基づいてステップ1記述の記述要素に対応するステップ2記述の記述要素で置き換えることにより実現できる。ただし、図 3.13 (a) のように網の判断による分岐が生じる場合、単にステップ1の記述要素を置き換えただけでは、置き換えられたステップ2の記述要素により生じる分岐が無視されてしまう (図 3.13 (b)) ので、信号定義データベースを参照して次の記述要素がくるべき位置を決定する (図 3.13 (c))。

##### (2) ステップ2記述からステップ3記述の変換

一般的にステップ2記述のようなシーケンス図形式からステップ3記述のような状態遷移図形式の変換は、図 3.14 (a) のようにユーザから網へのメッセージを入力、入力の直前を状態、網からユーザへのメッセージを出力とする方法が用いられている。この方法は、シーケンス図形式の仕様と状態遷移図形式の仕様は等価となる。ところが、ステップ2記述はユーザと網間の信号送受であり、ステップ3記述は網の内部処理であるため、このような形式的な変換方法は使用できない。そこで、図 3.14 (b) のようにステップ2記述の記述要素に対応する処理仕様を部品としてデータベースに登録し、変換時に

部品のリンクを行なうことで処理仕様を生成する。

上述のステップ1-2、2-3記述変換により、入力・編集したステップ1記述が変換された例（ステップ2のプロトコルは、Q.931を使用）について、各ステップ間の対応付けを図3.15に示す。

### (3)ステップ3記述から通信サービスソフトウェアへの変換

SDLで記述された処理仕様を通信サービスソフトウェアに変換する必要がある。通信サービスソフトウェアの記述は、仮想的リソースの論理インタフェースによる状態遷移制御を記述すればよいため、以下に示すC言語状態遷移記述法で記述が可能である。これは、SDLの記述に極めて近いため、SDLの記述形式をC言語の形式に変換する方法だけで実現できる。変換例を図3.16に示す。

#### C言語状態遷移記述法

状態遷移制御の処理構造は、C言語のswitch、case文を用いて状態番号で判定・分岐させるswitch文の中にプラットフォームからの受信メッセージの判定・分岐させるswitch文を入れ子にした構造で、簡単に表現できる（状態遷移手順と呼ぶ）。また、受信メッセージのテーブル（分析テーブルと呼ぶ）とプラットフォームへのメッセージ送出テーブル（タスクテーブルと呼ぶ）を配列データ定義として表すことにする。一方、共通部品はC言語記述の関数として作成し、状態遷移手順内で通常の関数呼び出しとして用いる。これを以下、C言語状態遷移記述法と呼ぶ。図3.17の状態遷移図をC言語状態遷移記述法により記述した例として、状態遷移手順を図3.18に、分析及びタスクテーブルを図3.19に示す。

### 3.6.3 準正常手順の自動付加手法

準正常手順のうち、陽にサービス仕様に記述されたもの以外にも必要となる事象は発生する可能性があり、特に下記の2点の原因による準正常手順の抜けが重要となる。

- (i) 網内の処理結果について対応する処理が網羅されていない。
- (ii) 記述された以外のメッセージを網が受信する。

これらの原因で発生する準正常手順の抜けについては、以下の方法で自動的に手順を付加する。

- (i) については、信号定義データベースに登録されている内部処理結果と記述されている処理とを比較し、終了処理を付加する。
- (ii) については、SDL記述における予定外入力の記述子である“INPUT\*”を用いることにより、記述された入力信号以外の入力がある場合には処理を終了できるようにする。

準正常を自動的に付加すると、一般的には要求外動作の仕様が生じられる可能性があり、要求外動作を削除する処理が必要となる。しかし、サービス要求の定式化が難しく、要求外動作仕様の機械的検出が困難であるため、本方式ではこの自動処理は行わず、動的な検証の段階（サービスプロトタイプシステム導入による試験）で、ユーザやサービス開発者により洗い出しを行ない、必要に応じて修正する方法をとることにする。この方法の有効性については、第5章で評価する。

### 3.6.4 冗長仕様の最適化手法

上記自動仕様変換では、最初の仕様入力であるシーケンス図が基本的には木構造になっていること、また変換過程で上述の準正常自動的付加処理が実施され、これがまた多くの枝を発生させる。この結果、ステップ3のSDL図においては、同一状態や同一遷移枝が生じられ、仕様が冗長に膨らむ結果が発生する。このうち、仕様の冗長さに強く影響を与えるのは同一遷移枝であり、これが最終的にプログラミング言語による通信サービスソフトウェアの記述量を増加させることになる。このため、同一状態に向かう同一遷移枝については、図3.20に示すようなマージ処理を実施することにする。これによる効果を見積るため、一般通話サービス、留守番録音サービス及び三者通話サービスの3つでマージ可能な状態遷移枝を調査し、それによるSDLの記述量を測定した。この結果を表3.5に示す。この結果、約20%~30%の記述量の削減が期待されることがわかった。このうち、いずれのサービスでも約20%は終了処理の遷移枝のマージ効果である。

### 3.6.5 サービス記述要素追加手法

サービスを構成する機能の構成要素（以下、これを機能要素と呼ぶ）は、伝達網のハードウェアやサービスプラットフォームの機能が追加されるに伴い、常に増加していくものである。このサービスの機能要素の追加は、そのままサービス記述要素に反映し、そ



の度にサービス記述要素を追加しなければならない。このため、できるだけサービス記述要素の追加が容易な構成が求められる。

記述要素の追加は、以下の項目が必要となる。

(i) 記述要素アイコン/言語表現の定義

(ii) 記述要素対応の処理系機能追加

① 構文解析

② 変換部品

サービス機能要素の追加では、シーケンス図記述要素の場合と網内処理記述要素の場合とでは対応が異なり、以下にそれぞれを示す。

シーケンス図記述要素の場合：シーケンス記述要素としては、これ以上新規に追加するものはないため、この場合の追加は、新規プリミティブの追加に相当する。新規記述要素の追加がないため、(i)や(ii)の構文解析としては変更がなく、新しい名前を持つ変換部品を追加すれば実現する。

網内処理記述要素の場合：この場合は、新規記述要素の追加が考えられるため、上記(i)と(ii)の処理が必要となる。(i)と(ii)の②は、各エディタで対応できるが、(ii)の①の処理が問題となる。このため、特に、網内処理記述要素の構文解析の処理系は、UNIXで提供されるツールである、“lex”（入力字句解析処理）と“yacc”（構文解析処理）を用いて構成した。これにより、新たに記述要素が追加されても比較的簡単に機能追加を行なうことができる。

```
[a]Information name : DIAL
[b]Direction : User -> Switch
[c]Signal primitives : dial_reception
                       : dial_analyze
                       : callee_examination
[d]Parameter         : timing = 00:00:20
                       : count = default
                       : buffer = tel_num
                       : callee = SUB_b
[e]Division causes  : NORMAL,HANGUP,TIMEOUT,NO_USE,
                       SELF_CALL,SPECIAL_NUMBER,BUSY
```

(a) 情報定義データベースの変換知識例

```
[a]Signal name : dial_reception
[b]Direction : User -> Switch
[c]Parameters and default values : timing = 00:00:20
                                   : count = 10
                                   : buffer = tel_num
[d]Division causes : NORMAL,HANGUP,TIMEOUT
[e]SDL parts file  : DIALRECEPTION.DGM
```

(b) 信号定義データベースの変換知識例

図 3.11 変換知識データベースの知識例

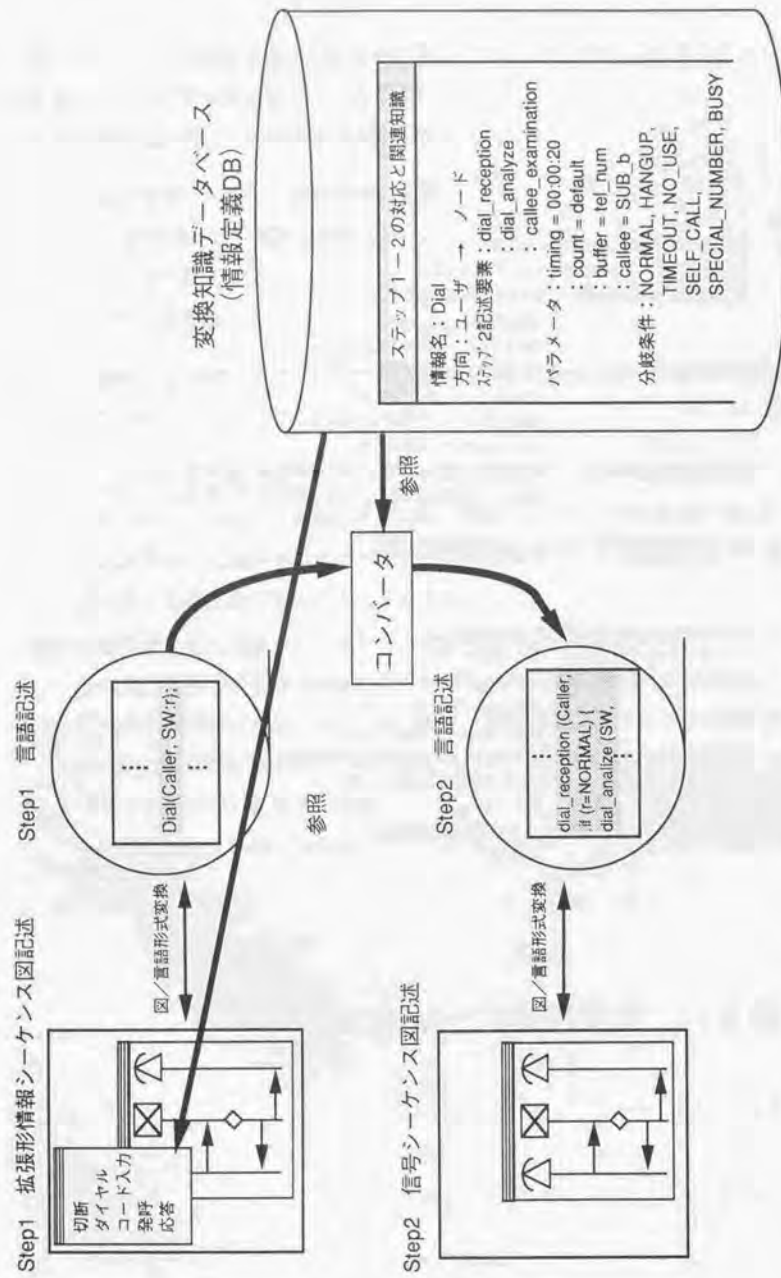


図 3.12 情報定義 DB を用いたステップ1-2記述変換

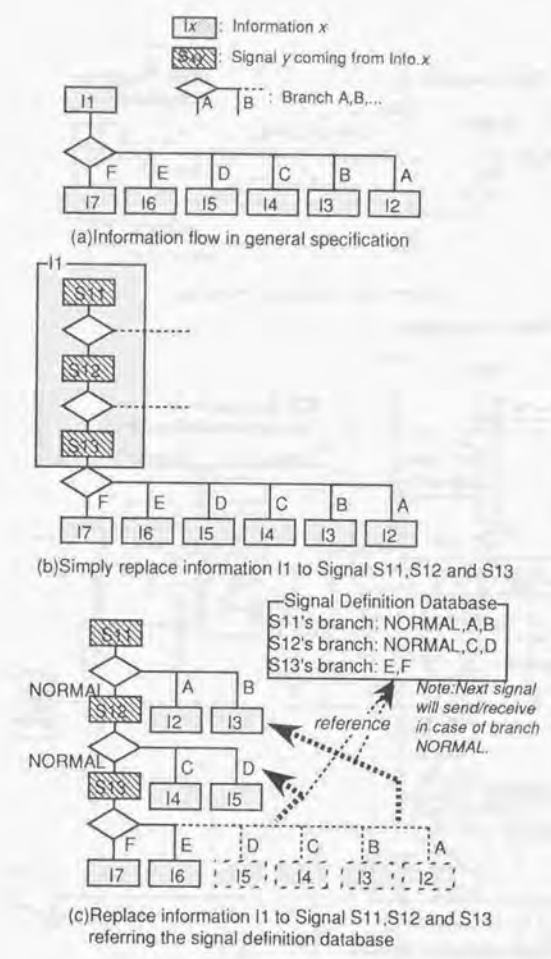


図 3.13 情報定義データベースと信号定義データベースの両方を用いた変換 (ステップ1→ステップ2)



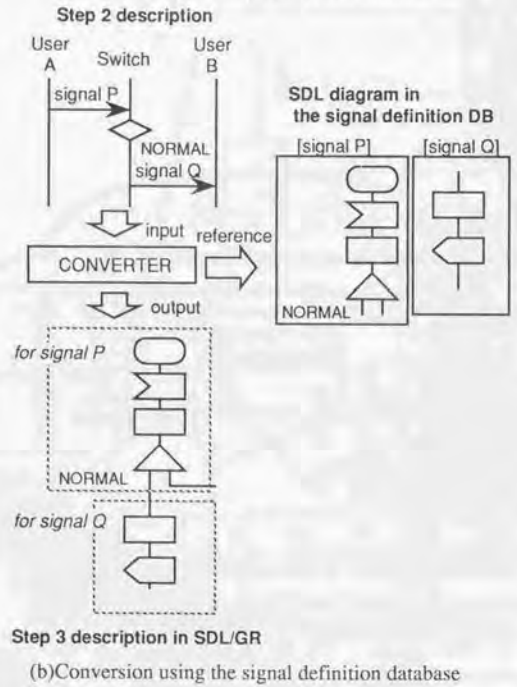
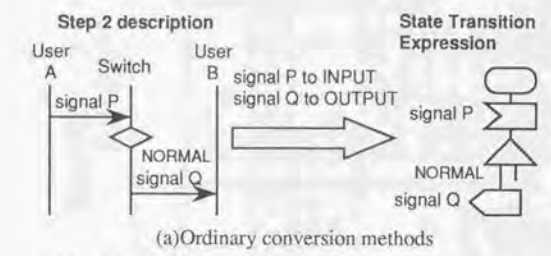
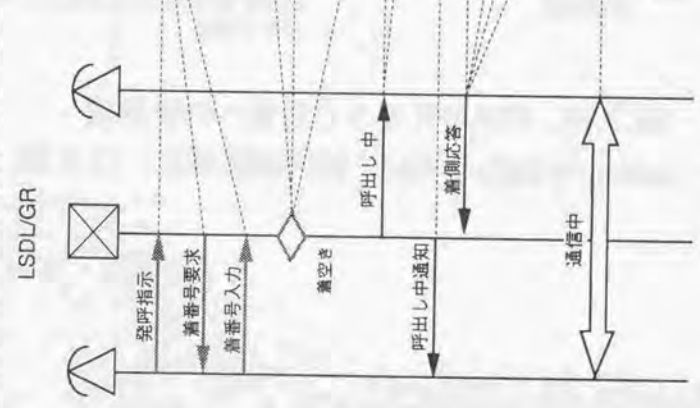


図 3.14 ステップ 2 記述からステップ 3 記述への変換

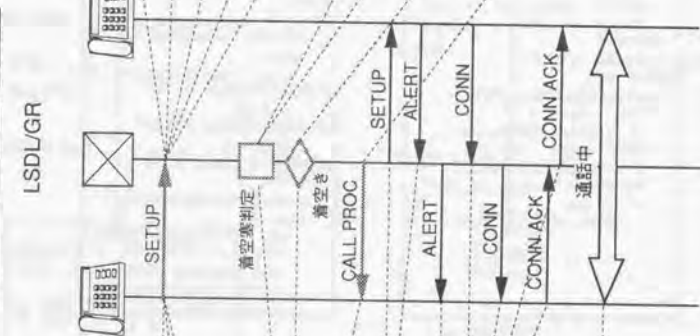
[入力・編集]

Step 1: 拡張情報シーケンス図



[自動変換]

Step 2: 信号シーケンス図



[自動変換]

Step 3: 仮想リソース制御SDL図

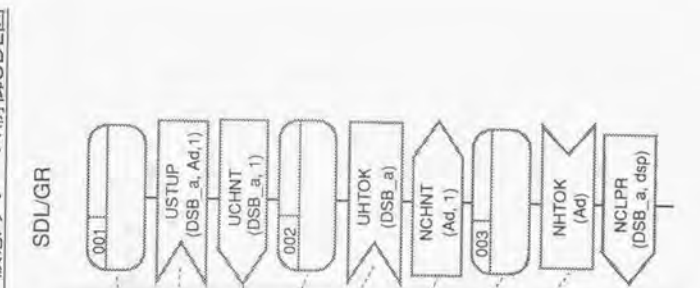


図 3.15 入力・編集した仕様記述と自動変換された記述との対応

```

PROCESS POTS;
STATE 001;
INPUT LSCNR (Caller, OF);
OUTPUT TRHNT (PBR);
NEXTSTATE 002;
INPUT *;
STOP;
STATE 002;
INPUT THTK (PBR);
OUTPUT HWSET (Caller, PBR);
OUTPUT TONON (Caller, DTN);
NEXTSTATE 003;
INPUT *;
STOP;
STATE 003;
INPUT DIREC (Caller);
TASK 'MAX :=
      DIAL_STORE (num);
TASK 'count := 1';
OUTPUT TONOF (Caller);
DECISION 'count_EQ_MAX';
('YES'):
  HWREL (Caller, PBR, FO);

```

(a) SDL/PR による処理  
仕様記述

```

POTS(){
switch (stn){
case 1:
  switch(inorder(st1)){
  case 0:
    task_routine(tsk1_0);
    stn = 2;
    break;
  default:stn = -1;
    break;}
  break;
case 2:
  switch(inorder(st2)){
  case 0:
    task_routine(tsk2_0);
    stn = 3;
    break;
  default:stn = -1;
    break;}
  break;
case 3:
  switch(inorder(st3)){
  case 0:
    MAX =DIAL_STORE (num);
    count = 1;
    task_routine(tsk3_0);
    switch (count==MAX){
    case YES:

```

(b) C 言語状態遷移記述法に  
よる通信サービスソフト  
ウェア記述

図 3.16 SDL/PR から C 言語への変換例

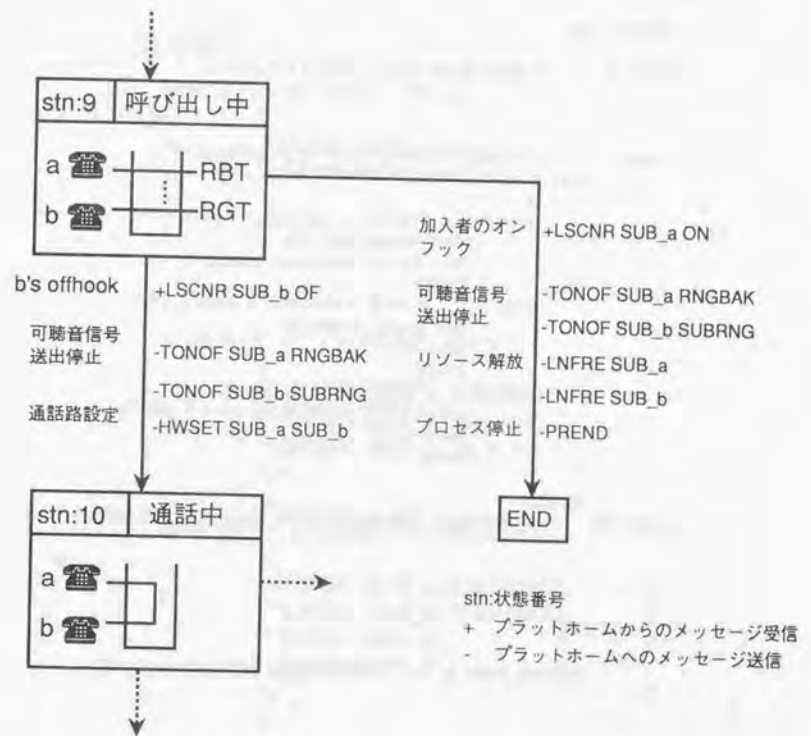


図 3.17 状態遷移図例 (呼出し中から通話中への遷移)



```

scp_std ()
{
    switch (stn)
    {
        case 0: /* Idle; Waiting for SUB_a's offhook */
            .
            .
            .
        case 9: /* Calling; Waiting for SUB_b's offhook */
            switch (input_order (INORDER_BUF, st9))
            {
                case 0: /* SUB_b's offhook: TASK 9_0 */
                    task_routine (tsk9_0);
                    stn = 10; /* Next state number */
                    break;
                case 1: /* SUB_a's onhook: TASK 9_1 */
                    task_routine (tsk9_0);
                    stn = -1;
                    break;
                default: /* ERROR */
                    perror ("### ERROR !!! stn = 9 ###");
                    stn = -1;
                    break;
            }
        case 10: /* Speaking; Waiting for SUB_a's or SUB_b's onhook */
            .
            .
            .
    }
    if (stn < 0)
    {
        process_end (); /* Calling for SCP end routine */
    }
}

```

図 3.18 C 言語状態遷移記述法による通信サービスソフトウェア記述—状態遷移手順

(1) 分析テーブル

```

char st9 [ ] [RORDER_MAX] =
{
    "LSCNR SUB_b OF",
    "LSCNR SUB_a ON",
    "x",
};

```

(2) タスクテーブル

```

char tsk9_0 [ ] [SORDER_MAX] =
{
    "TONOF SUB_a RRGBAK",
    "TONOF SUB_b SUBRNG",
    "HWSET SUB_a SUB_b",
    "x",
};
char tsk9_1 [ ] [SORDER_MAX] =
{
    "TONOF SUB_a RRGBAK",
    "TONOF SUB_b SUBRNG",
    "LNFRE SUB_a",
    "LNFRE SUB_b",
    "x",
};

```

図 3.19 C 言語状態遷移記述法による通信サービスソフトウェア記述—分析及びタスクテーブル

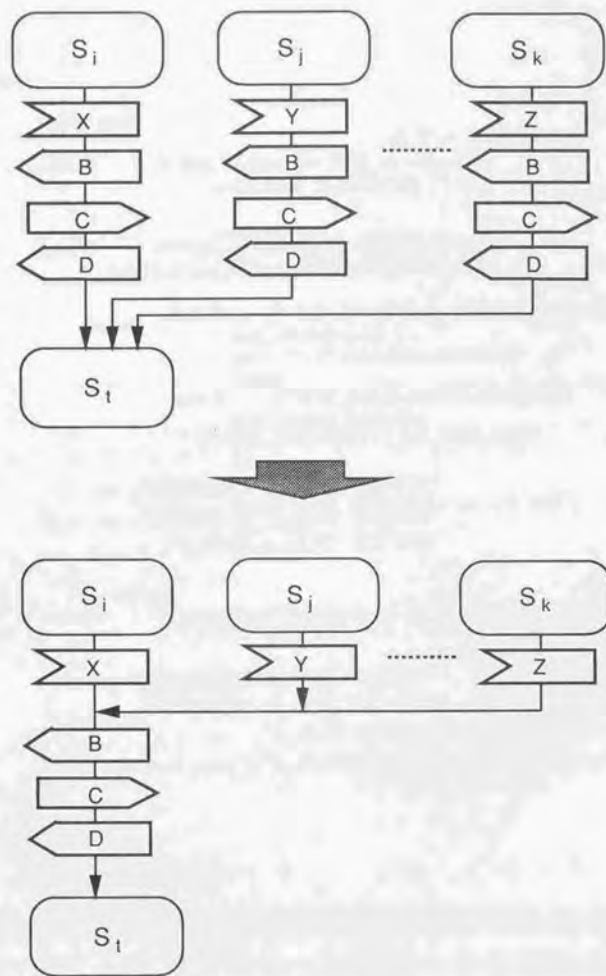


図 3.20 同一遷移枝のマージ処理

表 3.5 冗長仕様の最適化による記述量削減効果

対象サービス	最適化による削減量
一般通話	21 %
留守番録音サービス	24 %
三者通話サービス	32 %



## 3.7 試作による評価

### 3.7.1 試作システムの構成

以上述べたきた実現技術による仕様設計/ソフトウェア生成システムの構成を図 3.21に示す。本作成環境の試作システムをワークステーションSUN3/470上に構築した。OSはUNIXであり、ウィンドウシステムはX windowを用いた。この結果、試作システムは表 3.6に示す規模で実現できた。

### 3.7.2 仕様設計/ソフトウェア生成試作システムの評価

#### (1) サービス仕様の記述結果

本システムにおけるサービス仕様記述の容易性を評価するため、現在実用化のため開発中の新サービスである、改良形電話会議サービス、電話応答代行サービス、CCBS (Call Completion to Busy Subscribers、通称キャブオンサービス)、呼び返しサービス、迷惑電話拒否サービスの5つについて記述を行なった。仕様入力は拡張形情報シーケンス図により入力した(MSDL/GR)。入力の作業量は、手動で入力した記述要素の数で計測した。また、この入力のステップ1の言語形式に変換した記述量(MSDL/PR)、変換されたステップ2の記述量(MSDL/PR)、変換されたステップ3の記述量(SDL/PR)および最終的に生成されたC言語による通信サービスソフトウェアの記述量を行数で計測した。この結果を通信サービスソフトウェアのC言語の記述量で正規化して、図 3.21に示す。比較のために一般通話の記述量も示している。

#### (2) 考察

図 3.21に示した結果より、以下の点が明らかになった。

- (i) ステップ1の図形式では、C言語で直接記述する場合に比べて、いずれのサービスの場合も約10%以下で仕様記述が可能となる。
- (ii) ステップ1の言語形式では、C言語記述に比べて、約20%以下に、ステップ2の言語形式では、約25%以下に変換される。
- (iii) ステップ3のSDL記述は、C言語記述の約50%に変換される。

一方、変換用の知識データベースが全て準備されているという条件の下では、以下の点が明らかとなった。

(iv) いずれのサービスもステップ1の図形式入力では、1週間以内で実現できる。

これより、本作成環境を用いると、先に示したC言語状態遷移記述法に比べて、約90%以上の記述量が削減でき、大幅な記述性向上が実現される。また、仕様作成に要する期間も削減され、開発効率の大幅な向上が期待される。

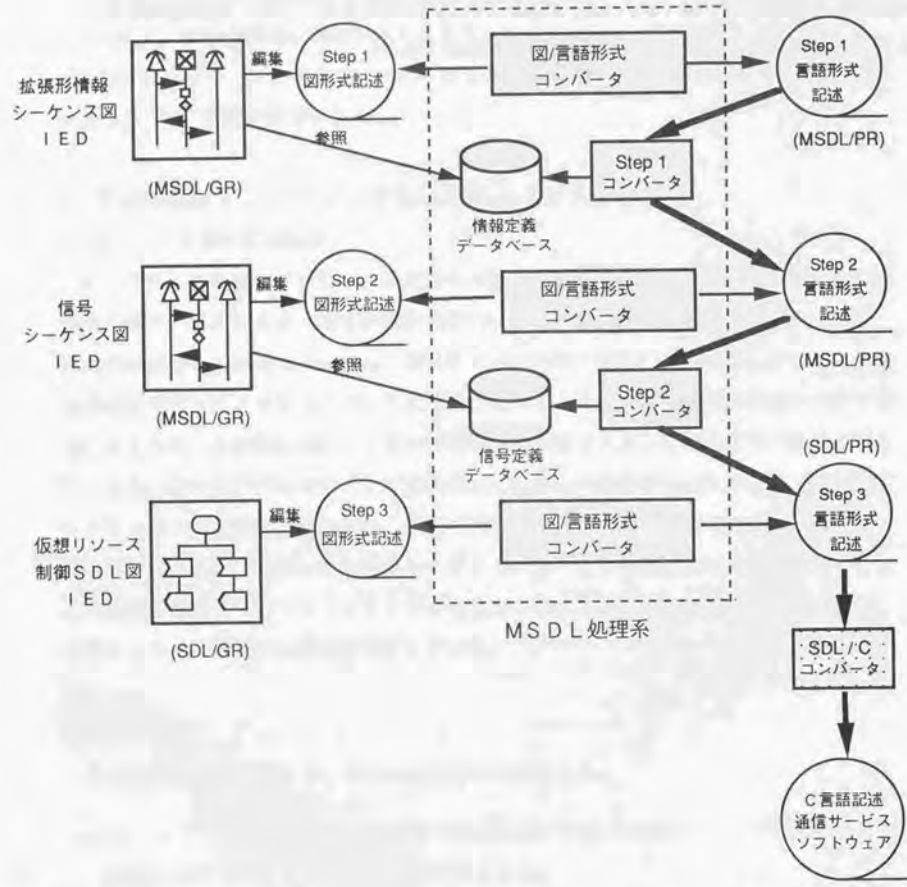


図 3.21 仕様設計/ソフトウェア生成システムの構成

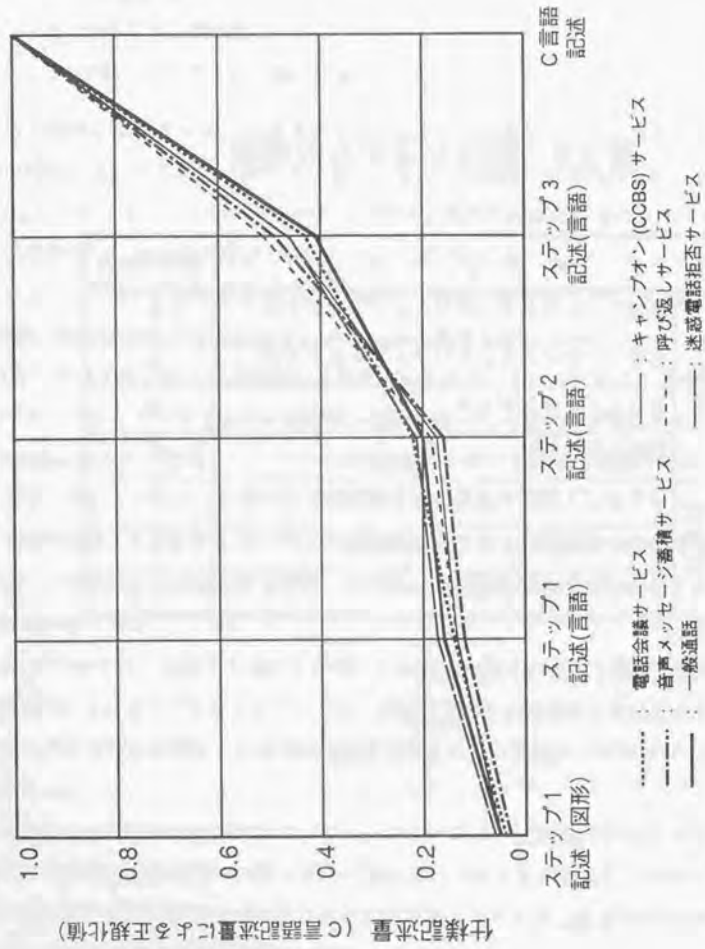


図 3.22 新サービス記述における各ステップの記述量



表 3.6 試作システムの規模

	モジュール	規模(Kline)
編集系	情報シーケンスインテリジェントエディタ	15
	信号シーケンスインテリジェントエディタ	8
	SDL/GRエディタ	6
	画面共通処理部	4
変換系	ステップ1記述→ステップ2記述コンバータ	1
	ステップ2記述→ステップ3記述コンバータ	1
	SDL/PR-C言語コンバータ	0.5

### 3.8 むすび

通信サービスソフトウェア自動作成方式のうち、サービス仕様設計/制御ソフトウェア生成環境に関して、以下の3つの主要技術課題についての検討結果を述べた。

- (1) 仕様記述法
- (2) 仕様入力/編集法
- (3) 仕様/プログラム自動変換法

まず最初に通信サービス仕様設計/ソフトウェア生成システムに対する5つの設計指針を作成した。この設計指針に基づき、上記3つの課題の実現技術を明らかにした。

(1)については、メッセージシーケンス記述と網内処理機能記述の2つで構成される拡張形情報シーケンス記述法を提案した。本記述法におけるメッセージシーケンスを記述するため、新たに図形式と言語形式の両方を持つ、階層形サービス仕様記述言語MSDL (Message Sequence Description Language)を開発した。

(2)については、サービス仕様の入力を非エキスパートにも可能にするため、高機能なマン・マシンインタフェースを持ち、知識を用いた入力支援を実現する、IED (Intelligent Editor)を構成した。サービス仕様記述に関する知識を登録するデータベースとして、ステップ1とステップ2に関連する情報定義データベース、ステップ2とステップ3に関連する信号定義データベースを設定した。ここに登録されている知識により、リソース状態による分岐の発生箇所、条件は、定義環境が自動的にサービス仕様設計者に提供可能となった。

(3)については、上記2つのデータベースに変換知識として、5種類を埋め込み、これに基づいてステップ1からステップ2、ステップ3と自動変換する仕組みを確立した。ステップ3のSDL記述は、C言語状態遷移記述法による通信サービスソフトウェアに変換される。

本システムの有効性を示すため、ワークステーション上に試作システムを作成し評価した。現在実用化を検討中のサービス仕様を本システム上で記述し、ステップ1の図形式ではC言語記述に比べて約10%以下で実現可能となること、及びいずれのサービスも1週間以内で記述可能となることを明らかにした。

## 第4章 通信サービス仕様の検証技術

### 4.1 まえがき

通信サービス仕様設計/ソフトウェア生成システムを構成することにより、通信サービスソフトウェアの効率的作成が実現されるようになった。しかし、ここで作成された通信サービスソフトウェアは、その仕様入力为非エキスパートであるサービス開発者が行っており、仕様自体に多くの誤りを含んでいる可能性が高い。そこで、このシステムによる仕様入力が行なわれた後で、十分な検証を行なう必要がある。また、検証結果は仕様入力をしたサービス開発者に戻し、仕様を修正させることが前提となるため、入力した仕様のレベルで検証し、そのレベルで即、結果をフィードバックさせる方式が望ましい。本章では、非エキスパートを指向した通信サービス仕様設計/ソフトウェア生成システム上で実現する、効率的な仕様検証技術の確立を目指す。

従来、この分野では、検証と試験という用語が用いられてきており、検証は機械的に厳密に行なうもので、試験は人間が動作を判断しながら行なうものといった、おおまかな分類はあったが、厳密には定義されていないので、本論文では以下のように定義して用いることにする。

**検証：**通信サービス仕様について、ある決められたアルゴリズムにより、機械的にその正常性を確認するものを呼ぶ。静的なチェックと言える。

**試験：**通信サービス仕様を動作させて、人間が判断することにより、その正常性を確認するものを呼ぶ。動的なチェックと言える。

試験は、一般には通信サービス仕様だけではなく、この仕様に基づいて作成された通信サービスソフトウェアの正常動作の確認を行なうためにも実施される。しかし、通信サービスソフトウェアの動作正常性の試験においても、仕様上の試験がかなりの部分を占めること、またプログラムロジックのレベルの試験については従来の検討成果が活用できること等より、本論文では通信サービスソフトウェアの試験でも通信サービス仕様の試験として取り扱うことにする。すなわち、本論文ではこの点も含めた形で、通信サービス仕様についてのみの検証・試験技術を明らかにする。

本章では、通信サービス仕様の検証技術を述べ、試験技術については第5章で述べる。

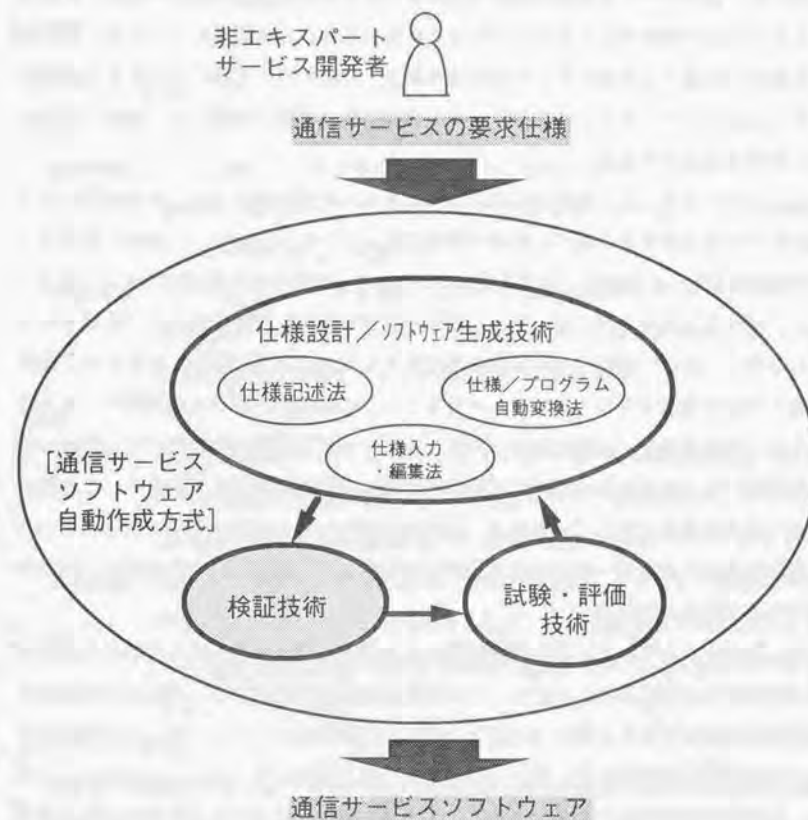


図 4.1 通信サービス仕様の検証技術



## 4.2 通信サービス仕様の検証・試験の体系

### 4.2.1 非エキスパート設計通信サービス仕様の検証・試験における要求課題

従来より、通信サービス仕様の検証や試験については多く検討されているが、これはあくまでも仕様の設計がエキスパートにより行なわれることを前提としている。このため、本論文の対象とする通信サービス仕様の検証・試験については、そのままでは適用できず、非エキスパートによる通信サービス仕様開発の特徴を考慮した、検証・試験の体系を構築する必要がある。

非エキスパートサービス開発者による通信サービス仕様の設計では、その設計における対象をサービス開発者に閉じた範囲で設計できるようにするため、一般的に他のサービス開発者の設計した通信サービス仕様については全く考慮せずに作成することになる。さらに、サービス開発者内においても、自分が過去に作成した通信サービス仕様を全て参照しながら、新しい通信サービス仕様を設計することは非常に困難であるので、設計時は全くそれを考慮せずにできることが望ましい。このため、設計した通信サービス仕様について単体の検証・試験を行なった後、まずサービス開発者内に閉じて、既作成済みの通信サービス仕様との矛盾（以下競合と呼ぶ）の検証・試験を実施する。この場合は、サービス開発者に閉じているため、既作成通信サービス仕様も自身で管理していることや設計済みの通信サービス仕様の数が比較的少ないと思われ、十分な検証・試験が可能であると考えられる。

また、この後、さらにサービス開発者間をまたがり、既通信サービス仕様との競合を検証・試験する必要がある。ただし、この場合は対象が全国となり、作成済みの通信サービス仕様も膨大であると考えられることから、厳密性は欠いても（サービス性は欠くが、ラフな検証・試験により、疑わしい通信サービス仕様を全てリジェクトすることにより、安全側に保証することはできる）、効率的に検証・試験を行なう手法が要求される。

これらの検証・試験を実施して十分な準備を行なってもなお、実際に商用のシステム上で通信サービスソフトウェアを動作させると仕様間で矛盾が発生する可能性も残っており、これに確実に対処するため、サービスプラットフォーム側にも、通信サービスソフトウェアの動作異常時にはそれを検出し、直ちに停止させる機能やオンラインで試験を実施する機能が必要である。

### 4.2.2 非エキスパートのソフトウェア開発における検証・試験技術の体系

上述の点を考慮した、非エキスパートによる開発を前提とした、通信サービスソフトウェアの検証・試験技術の体系を図 4.2 に示す。以下に、各検証・試験技術の具体的な内容を示す。

#### [オフライン環境]

##### 単体

**単体検証：**1つの通信サービス仕様内に閉じた正常性の静的チェックで、サービス開発者が通信サービスソフトウェア作成環境上で自身が作成した通信サービス仕様に対して実施する。

**単体試験：**1つの通信サービス仕様内に閉じた正常性の動的チェックで、サービス開発者がサービスプロトタイピングシステム（第5章で述べる）上で自身が作成した通信サービス仕様に対して実施する。

##### 複合

**複合検証：**作成した通信サービス仕様に対して、同一サービス開発者内または他サービス開発者内の既に作成済みの通信サービス仕様の中で、サービス競合の可能性のある候補に対して、競合の発生を静的にチェックする。

**複合試験：**作成した通信サービス仕様に対して、他の全てのサービス開発者が既に作成済みの通信サービス仕様の中の、仕様間の競合の可能性のある候補に対して、仕様間の競合を試験用の実機システム上で動的にチェックする。

#### [オンライン環境]

**異常動作防止メカニズム：**サービスプラットフォーム内部に埋め込まれる機能であり、通信サービスソフトウェア間での動作矛盾発生を予測した場合、優先度もしくはネゴシエーションにより、どちらか一方の動作を停止する。

**異常動作監視・オンライン試験：**サービスプラットフォームの外部から、通信サービスソフトウェアの動作を監視し、異常時には直ちに停止させる機能とオンラインで指定する通信サービスソフトウェアの動作試験を実施する機能を持つ。

上記技術のうち、複合試験は従来より検討されてきた技術であり、その成果が適用できること、またオンライン環境の2つの技術は、現在プラットフォーム構築技術としてNTTを初めとして各方面で検討されていることより、本論文では単体検証、単体試験、複合検証についての検討を行なう。本章では、単体検証と複合検証の実現技術を明らかにし、単体試験技術については次章で述べる。

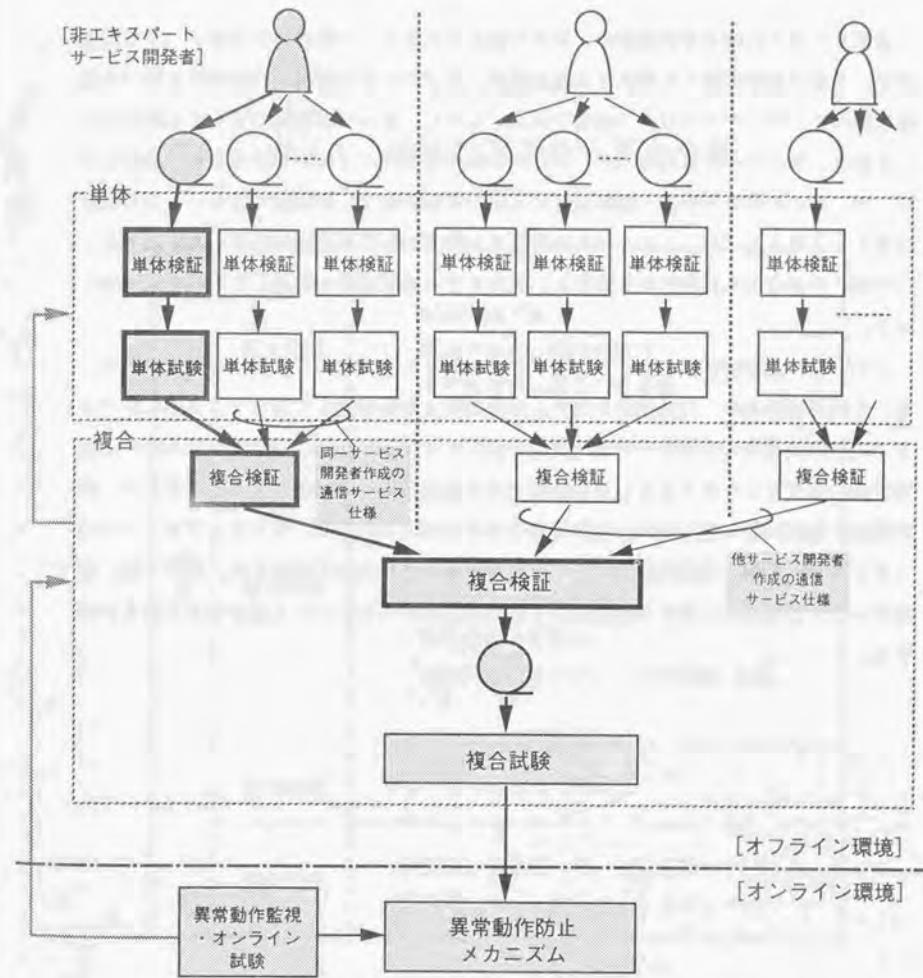


図 4.2 通信サービスソフトウェアの検証・試験技術体系

→ [ ] → : 検証・試験の流れ  
 [ ] : 本論文での対象技術



### 4.3 単体検証の分類と検証項目の明確化

通信サービス仕様の単体検証は、深さの観点で見ると、一番浅い文法的レベルの構文検証、仕様の論理的誤りを検出する論理検証、及びサービス要求との整合性を調べる意味検証の3つのレベル分けが一般的である。しかし、サービス仕様の質までも問われている現在、単にサービス仕様の正しさだけを検証するだけでは不十分である。本論文では、サービスの質を利便性や操作性等の人間の主観評価でしか検証できないものも検証対象として取り上げた。このレベルの検証を本論文で新たに価値検証として定義する。この深さの観点から仕様検証を分類し、対象とする検証項目を記述したものを表4.1に示す。

このうち、意味検証は、サービス要求をどのように定式化して表現するかにおいて、2、3試みはあるが、自動検証システム構築技術までは到達しておらず、実用化レベルまでの技術開発には今後多くの研究成果が必要とされる。また、価値検証は人間の主観的評価に負うところが大きく、自動検証では実現されにくい対象と言える。そこで、意味検証と価値検証に対しては、基本的には人手によることにし、サービスプロトタイプシステムを用いた動的試験として実現を図ることを次章で検討する。以下では、通信サービス仕様単体に対する構文検証と論理検証のレベルについて検証方式を明らかにする。

表 4.1 検証の深さから見た分類

深さ	構文検証	<ul style="list-style-type: none"> <li>・記述の過不足</li> <li>・記述の形式、属性の誤り</li> <li>・その他言語仕様との整合性</li> </ul>
	論理検証	<ul style="list-style-type: none"> <li>・無限ループ、デッドロック</li> <li>・情報、信号順序規則との整合性</li> <li>・準正常、スクリーニング条件の正当性</li> <li>・信号方式との整合性</li> <li>・状態遷移の正常性</li> <li>・交換装置制御プロトコルとの整合性</li> </ul>
	意味検証	<ul style="list-style-type: none"> <li>・サービス要求との整合性</li> </ul>
	価値検証	<ul style="list-style-type: none"> <li>・利便性、有効性—サービス機能</li> <li>・操作性—マン・マシンインタフェース</li> </ul>

## 4.4 変換レベルに対応させた階層形仕様検証法

### 4.4.1 概要

通信サービス仕様の検証は、4.3で示したように、深さの観点から見た4レベルの検証のうち、構文検証、論理検証の2レベルが対象となる。ここでは、2レベルの検証を、仕様変換の各ステップに対応させ、以下の3つのステップに分けて検証を行う。この構成を図4.3に示す。これは、仕様の変換部品単体の誤りは事前に検証しているが、その組み合わせにより発生する誤りについては全てを事前に網羅することは不可能なため、変換の過程で誤りの発生が予測され得ること、各変換ステップが個々の検証技術として独立性が高いこと、及び従来の検証方式との親和性が高いこと等を考慮したためである。

#### [ステップ1] サービス手順検証

- (i) 拡張形情報シーケンス図の構文検証
- (ii) サービス手順規則との正常性検証

#### [ステップ2] プロトコル手順検証

- (i) 信号シーケンス図の構文検証
- (ii) 信号方式との整合性検証

#### [ステップ3] リソース制御手順検証

- (i) 状態遷移の構文検証
- (ii) 仮想的リソース状態遷移手順との整合性検証

上記のステップのうち、ステップ2及びステップ3については、従来プロトコルや状態遷移手順の検証方式として研究され、確立した技術となりつつある、到達可能解析法<sup>[26], [27]</sup>を適用することにし、本論文では従来ほとんど検討がなされていない、ステップ1のサービス手順についての検証法のみを対象とする。以下、4.4.2にサービス手順（情報シーケンス記述）の構文検証法を、4.4.3に論理検証であるサービス手順規則との整合性をチェックする、段階的検証方式を示す。

一方、これらの仕様検証の結果、誤りが検出された場合には、サービス開発者が入力・編集したレベルであるステップ1の通信サービス仕様上に表示し、サービス開発者による修正を促す必要がある。この場合、ステップ1のサービス手順の検証では、拡張形情報シーケンス図で記述された仕様を検証するので、そのまま検証結果より誤り箇所を表示できが、ステップ2、ステップ3については、誤り箇所をステップ1の拡張形情報シ

ーケンス図レベルに逆変換して表示する必要がある。この誤り箇所の逆変換表示に実現方式については、4.6節で述べる。

### 4.4.2 サービス手順の構文検証法

構文検証の対象としては、以下の項目が対象となる。

- (i) 記述の過不足
- (ii) 記述の形式、属性の誤り
- (iii) 言語仕様の構文規則との整合性

表4.2にサービス手順の構文検証として対象とする項目を示す。これらの検証は、サービス仕様の図形式記述の入力/編集を行う、IED (Intelligent Editor)上で実現するため、論理検証の対象である“無限ループ”や情報を示す矢印の方向の検証も便宜的に含ませている。

### 4.4.3 サービス手順の論理検証法

#### (1) サービス手順記述における仕様誤り

サービス手順記述における仕様誤りは以下の二つに分類できる。

##### (エラー#1) 通信中状態とその遷移の誤り

拡張形情報シーケンス図で通信中状態とその他のプリミティブとを分けると、図4.4のようなシーケンス図として考えることができる。このシーケンス図をマクロ化情報シーケンス図と呼ぶ。マクロ化情報シーケンス図では、二つの通信中状態間にあるプリミティブ群は一つの遷移手順とみなす。この遷移手順内には、ある状態から別の状態へ遷移させるためのプリミティブがあり、これを遷移トリガと呼ぶ。

通信中状態と遷移トリガの順序誤りが、エラー#1の原因となる。言い換えると、このエラーはサービス仕様のマクロなフローに関するエラーであると言える。このエラーを検出するためには、通信中状態と遷移トリガのみをチェックすればよく、遷移に関与しない他のプリミティブについてのチェックは不要である。

##### (エラー#2) 通信中状態間（遷移手順内）にあるプリミティブの誤り

網からの要求メッセージに対するユーザからの入力メッセージがない場合、この誤りとなる。このような誤りを検出するためには、プリミティブ個別に詳細にチェックする



必要がある。

### (2) 仕様誤りから見た修正方法

効率的で容易な誤り修正を実現するためには、仕様の広い範囲に影響を及ぼす誤り修正については、局所的な誤り修正の前に実行される必要がある。これは、先におこなわれた細部の修正が、その後の修正により無効になることを避けるためである。広い範囲に影響を及ぼす誤りは、上記のエラーのうちの、マクロなフローに関するエラー#1に属するものと考えられ、この誤り検出を最初に行なう必要がある。

### (3) 段階的検証方式

(1)の2つのタイプの誤りに対する検証は、効率的な誤り修正を考慮し、マクロなフローに関する誤りであるエラー#1に関する検証を最初に行ない、次に詳細に遷移手順内のプリミティブに関する誤りであるエラー#2に関する検証を行なうことにする。これを段階的検証方式と呼び、以下の2ステップで構成する。

[ステップ1] マクロ化情報シーケンス検証：通信中状態とその遷移の正常性の検証

[ステップ2] 遷移手順検証：遷移手順内のプリミティブ順序の正常性の検証

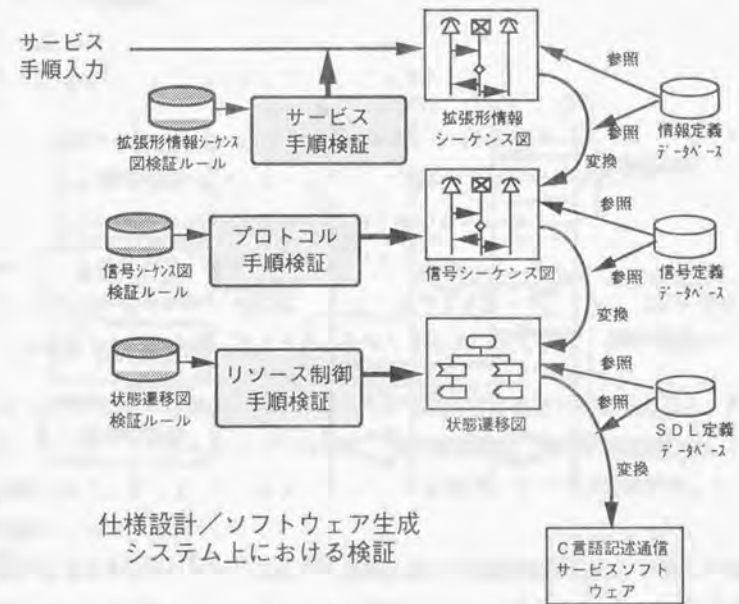


図 4.3 変換レベルに対応させた階層形仕様検証法

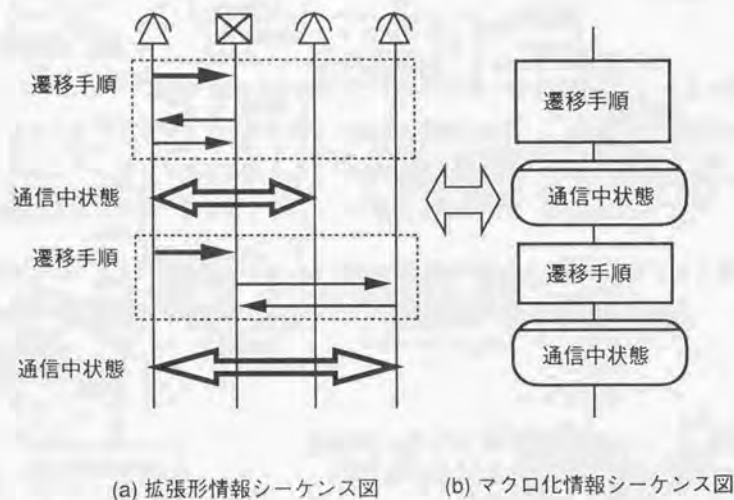


図 4.4 拡張形情報シーケンス図とマクロ化情報シーケンス図との対応

## 4.5 段階的検証方式

上述したように、段階的検証方式はマクロ化情報シーケンス検証と遷移手順検証の2ステップで構成される。以下では、それぞれについて、検証項目、検証ルール、検証アルゴリズムを中心に述べる。

### 4.5.1 マクロ化情報シーケンス検証

#### 4.5.1.1 検証項目

マクロ化情報シーケンス図は以下のルールを持つ。

- (i) 2者間の通信中状態は(a)通話中、(b)保留中、(c)初期状態、のうちの一つである。  
(Cを通信中状態の集合とすると、 $C = \{ \text{通話中}, \text{保留中}, \text{初期状態} \}$ )
- (ii) 二つの通信中状態間には一つの遷移手順がある。
- (iii) 一つの遷移手順には一つの遷移トリガがある。
- (iv) 通信中状態と遷移トリガは、一つの次通信中状態を決定する。(遷移関数は部分関数であり、遷移トリガの集合を  $T$  とすると、 $\delta: C \times T \rightarrow C$  で与えられる。)

最初のルールは、ユーザ2者間の通信中状態を定義する。もし2者以上のユーザがいる場合、このルールは各組のユーザに適用され、これらの状態の接続が全体としての通信中状態を表す。例えば、コールウェイティング状態は、ユーザa-b間通話中、ユーザa-c間保留中、のように表す。

2番目と3番目のルールは、通信中状態と遷移トリガが交互に現われることを意味する。このルールは分岐によって成立しない場合がある。つまり、遷移トリガと次通信中状態との間に分岐があり、この分岐がこの遷移を次通信中状態に到達することを妨げる場合、言い換えれば直前の遷移トリガを無効にする場合である。この場合、他の次通信中状態に到達するために、別の遷移トリガが無効になった遷移トリガの直後に記述される必要がある。このため二つの遷移トリガが連続して現われることになるが、これは誤りとして見なすことはできない。例えば、通信開始指示と通話中との間にユーザ話中による分岐があるとすると、この遷移では通話中に到達できない、つまりこの分岐は通信開始指示を無効にする。通信開始指示の直後には別の遷移トリガ、例えば通信終了指示が記述される。したがって、遷移トリガが二つ続けて記述されることになるが、これを誤りとはしない。



4番目のルールは、通信中状態と遷移トリガの関係が一つの次通信中状態を規定することを意味する。このことより、この次通信中状態とシーケンス図に記述された通信中状態とは一致しなければならない。以上より、マクロ化情報シーケンス検証の検証項目を以下の3つとする。

項目 (1-1) : 通信中状態と遷移トリガは交互に現われる。(遷移トリガを無効にする分岐がある場合を除く)

項目 (1-2) : 次通信中状態は現通信中状態と遷移トリガによって規定される。  
( $c \in C$ ) を通信中状態、 $t \in T$  を遷移トリガとすると、 $\delta(c, t) \in C$  となる。)

項目 (1-3) : 記述された次通信中状態と規定された次通信中状態は一致する。  
( $c' \in C$ ) を記述された次通信中状態とすると、 $\delta(c, t) = c'$  となる。)

#### 4.5.1.2 検証ルールの表現法

検証ルールを表すために、通信中状態と遷移トリガの簡単な表現を提案する。通信中状態  $c$  は二つのパラメータで構成し、 $c = (s, h)$  と表す。ここで、 $s \in \{1, 0\}$ 、また、 $h \in \{1, 0\}$  である。 $s$  は通話中で1を、それ以外のとき0をとる。また、 $h$  は保留中で1を、それ以外のとき0をとる。遷移トリガ  $t$  も二つのパラメータで構成し、 $t = [t_s, t_h]$  と表す。ここで、 $t_s \in \{s = s + 1, s = s - 1, s = s\}$ 、また、 $t_h \in \{h = h + 1, h = h - 1, h = h\}$  である。 $t_s$  は  $s$  に対する演算を表し、 $t_h$  は  $h$  に対する演算を表す。この表現を用いると、プリミティブは以下のように記述できる。

初期状態 :  $c = (0, 0)$

通信開始指示 :  $t = [s = s + 1, h = h]$

次通信中状態は、これらのプリミティブから、 $\delta(c, t) = (1, 0)$ 、つまり通話中となる。もし遷移が誤ったものであれば、次通信中状態  $c$  は  $c \notin C$  となる。したがって、項目 (1-2) の検証ルールは次通信中状態が  $c \in C$  であるかどうかだけを、例えば、 $s \in \{1, 0\}$  であることをチェックすればよい。ここで  $c$  と  $t$  はユーザ2者間の通信中状態と遷移トリガを表している。もし2者以上のユーザがいる場合には、これらの表現は各ユーザ間に適用され、各2者間の遷移は、互いに独立に扱われる。この表現法を使って通信中状態と遷移トリガを定義することで、状態遷移テーブルを用いずに項目 (1-2) の検証を

実行することができる。もし新しい通信中状態や遷移トリガが追加されたら、このプリミティブを上記表現法を用いて定義するだけでよく、検証ルールを修正する必要がない。

#### 4.5.1.3 マクロ化情報シーケンス検証のアルゴリズム

マクロ化情報シーケンス検証のアルゴリズムは以下の手順で行なう。

**Step 1** : 拡張情報シーケンス図からプリミティブを読み込む。このプリミティブが遷移トリガならStep 2へ、通信中状態ならStep 3へ進む。それ以外ならStep 1で次のプリミティブを読み込む。

**Step 2** : (項目 (1-1) の検証) マクロ化情報シーケンス図において、直前のプリミティブが遷移トリガかどうか調べる。もしそうであり、かつこの遷移トリガが無効にする分岐がなければ、読み込んだプリミティブは誤り。その他はStep 3へ進む。

**Step 3** : (項目 (1-2) の検証) 直前の通信中状態と読み込んだ遷移トリガから次通信中状態を計算する。もしこの状態が通信中状態として誤りならば、この遷移は誤り。正しければStep 1へ戻り、次のプリミティブを読み込む。

**Step 4** : (項目 (1-1) の検証) マクロ化情報シーケンス図において、直前のプリミティブが通信中状態かどうか調べる。もしそうであれば、読み込んだプリミティブは誤り。そうでなければ、Step 5へ進む。

**Step 5** : (項目 (1-3) の検証) 読み込んだプリミティブとStep 3で計算した通信中状態とを比較する。もし一致しなければ、読み込んだプリミティブは誤り。一致すればStep 1に戻り、次のプリミティブを読み込む。

Step 1で分岐のプリミティブを読み込んだ場合は、このアルゴリズムを再帰的に実行する。

#### 4.5.1.4 検証例

上記アルゴリズムを用いて、図 4.5に示すサービス仕様を以下のように検証する。

**Step 1** : 情報シーケンス図からプリミティブを読み込む。通信開始指示 ( $t = [s = s + 1, h = h]$ ) は遷移トリガなので、Step 2へ進む。

**Step 2** : マクロ化情報シーケンス図において、直前のプリミティブが通話中 ( $c = (1,$

0)) であるので、項目 (1-1) は満たされている。

Step 3: 次通信中状態を計算する。この場合、次通信中状態は通信中状態として誤り ( $\delta(c, t) = (2, 0) (\notin C)$ ) となるので、この遷移は誤りとなる。

## 4.5.2 遷移手順検証

### 4.5.2.1 検証項目

遷移手順検証では、メッセージ手順の基本的なルールとサービスの内容についてのルールとの2種類の検証ルールがある。

メッセージ手順の基本的なルールは、仕様に非依存であり、すべてのプリミティブがこの検証ルールに従う必要がある。このような検証ルールを共通検証ルールと呼び、以下の項目がある。

項目 (2-1) : 同じプリミティブは続けて記述されない。

項目 (2-2) : 要求メッセージと応答メッセージはペアを形成する。

項目 (2-1) は、拡張情報シーケンス図では同一プリミティブの連続は、一つのプリミティブで置き換えられることを意味する。

サービスの内容についてのルールは、プリミティブがサービス仕様の中で持つ意味から生じたものである。これらのルールを個別検証ルールと呼ぶ。各個別検証ルールは一つのプリミティブに属し、このプリミティブを検証するためだけに利用される。図 4.6 に共通検証ルールと個別検証ルールとの関係を示す。

### 4.5.2.2 検証ルールの表現法

4.4 で述べたように、遷移手順検証ではプリミティブの順序誤りを検出する。情報シーケンス図では、プリミティブの順序は二つのプリミティブの位置関係によって表現できる。このため検証ルールは、この位置関係についての情報を持っている必要がある。ルールに必要な情報は以下のようなものである。

#### (1) プリミティブ

ルールには二つのプリミティブ名が必要である。一つはそのルールが属しているプリミティブ、もう一つは検証するときに検索の対象となるプリミティブである。

#### (2) 位置関係

二つのプリミティブの位置関係は、以前もしくは以降に分けることができる。すなわち、PとP'をプリミティブとすれば、P'はPの以前または以降に存在する。しかしここでは、以前と以降の特別な場合として、それぞれ直前と直後を区別する。以降と直後では分岐を考慮しなければならない。分岐を含む位置関係は二つに分類できる。一つはすべて分岐に対して有効な関係、もう一つはいくつかの分岐に対して有効な関係である。図 4.7 にこれらの関係を示す。さらに、直前と直後は図 4.8 に示すように、同一ユーザ・網間の関係と異なるユーザ・網間の関係に分類できる。結果として、表 4.2 に示すように位置関係には全部で9とありあることになる。

#### (3) 誤りの原因

拡張情報シーケンス図では、誤りの原因は二つある。

(3-a) : 不必要なプリミティブの記述

(3-b) : 必要なプリミティブの抜け

最初の原因で誤りが生じた場合、その不必要なプリミティブを削除しなければならない。また、2番目の原因による場合では、その必要なプリミティブを追加しなければならない。この情報を用いて、検証はその仕様が正しいかどうかを決定する。

#### (4) 誤りのレベル

遷移手順検証では、検出した誤りを以下の二つに分ける。

(4-a) : Fatal error

(4-b) : Warning

Fatal error はサービスに非依存で、誤りが明らかなるものである。Warning はサービスに依存し、検証で誤りとは断定できないものである。

以上の情報を用いて、検証ルールを記述する。例えば、「着側応答要求は着番号入力以降の少なくとも一つの分岐に存在しなくてはならない。」という検証ルールの場合、このルールを以下のように構成する。

(1) プリミティブ : 着番号入力、着側応答要求

(2) 位置関係 : 以降のいくつかの分岐中



(3)誤りの原因 : 必要なプリミティブの抜け

(4)誤りのレベル : Fatal error

#### 4.5.2.3 遷移手順検証のアルゴリズム

遷移手順検証のアルゴリズムは以下の手順に従う。

**Step 1:** (項目 (2-1) の検証) 拡張形情報シーケンス図からプリミティブを読み込む。

もしこのプリミティブが直前のプリミティブと同じであれば、読み込んだプリミティブは誤り。読み込んだプリミティブが要求メッセージであれば、Step 2へ進み、入力メッセージであれば、Step 3へ進む。それ以外はStep 4へ進む。

**Step 2:** (項目 (2-2) の検証) 直後のプリミティブが、読み込んだプリミティブに対応する入力プリミティブかどうか調べる。もしそうでなければ、誤りとなる。Step 4へ進む。

**Step 3:** (項目 (2-2) の検証) 直前のプリミティブが、読み込んだプリミティブに対応する要求メッセージかどうか調べる。もしそうでなければ、誤りとなる。Step 4へ進む。

**Step 4:** もし読み込んだプリミティブが個別検証ルールを持つならば、Step 5に進む。もし持たないならば、Step 1に戻り、次のプリミティブを読み込む。

**Step 5:** ルールに指定されているプリミティブを検索する。もしこのプリミティブが指定された位置にあれば、Step 6へ進む。もしなければ、Step 7に進む。

**Step 6:** もしルールに指定された誤りの原因が不必要なプリミティブの記述であれば、Step 5で見つかったプリミティブは誤りであり、Step 8に進む。もし必要なプリミティブの抜けであれば、Step 4に進む。

**Step 7:** もしルールに指定された誤りの原因が必要なプリミティブの抜けであれば、このプリミティブは誤りであり、Step 8に進む。もし不必要なプリミティブの記述であれば、Step 4に進む。

**Step 8:** ルールに指定された誤りのレベルにより、検出された誤りが Fatal error か Warning かを決定する。

Step 5の検索中に分岐のプリミティブが現われたら、この検索手順は再帰的に実行される。

#### 4.5.2.4 検証例

上記アルゴリズムを用いて、図 4.9に示される仕様を以下のように検証する。

**Step 1:** 拡張形情報シーケンス図から着番号入力を読み込む。このプリミティブは直前のプリミティブと異なるので、項目 (2-1) は満たされる。このプリミティブは入力メッセージなので、Step 3に進む。

**Step 3:** 読み込んだプリミティブとペアとなる要求メッセージがあるので、項目 (2-2) の検証結果は正しい。Step 4に進む。

**Step 4:** 読み込んだプリミティブが個別検証ルール「着側応答要求は着番号入力の以降の少なくとも一つの分岐に存在しなくてはならない。」を持つので、Step 5に進む。

**Step 5:** 着番号入力の以降に着側応答要求を探す。着側応答要求が存在しないので Step 7に進む。

**Step 7:** 検証ルールにある誤りの原因が必要なプリミティブの抜けであるので、着側応答要求が誤りとなる。Step 8へ進む。

**Step 8:** 検証ルールにある誤りのレベルが fatal error なので、この検証結果は fatal error となる。

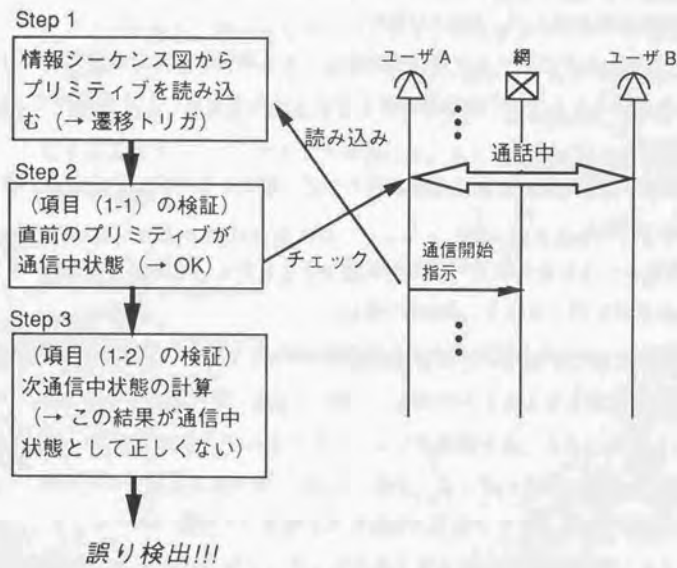


図 4.5 マクロ化シーケンス検証の例

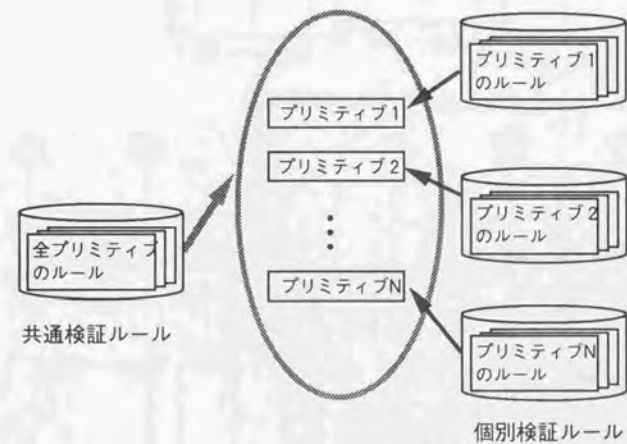
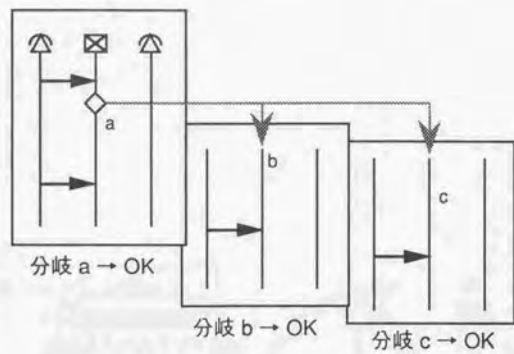
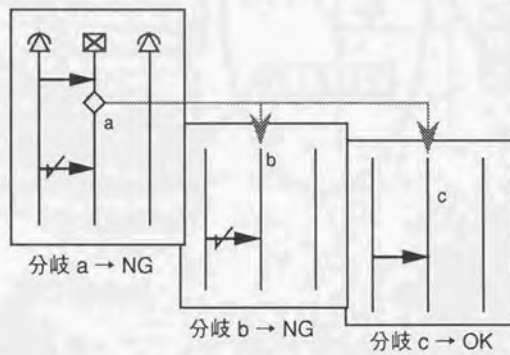


図 4.6 共通検証ルールと個別検証ルール





(a) 全分岐に対して有効な位置関係



(b) 少なくとも一つの分岐で有効な位置関係

図 4.7 分岐を含む位置関係

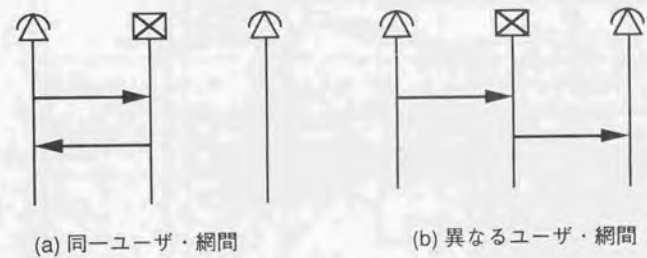


図 4.8 プリミティブの位置関係



図 4.9 遷移手順検証の例

表 4.2 位置関係の分類

位置	分岐	ユーザ・網間の関係
以前	——	——
以降	すべての分岐	——
	少なくとも一つ分岐	——
直前	——	同一ユーザ・網間
	——	異なるユーザ・網間
直後	すべての分岐	同一ユーザ・網間
		異なるユーザ・網間
	少なくとも一つ分岐	同一ユーザ・網間
		異なるユーザ・網間



## 4.6 検証結果の上位仕様逆変換表示法

### 4.6.1 下位レベル記述の通信サービス仕様の逆変換

一般に多段階の変換過程を持つ仕様の自動変換では、変換された仕様に対して下位のレベルで変更を加えた場合は、上位の部品単位での変更でないとは逆変換は基本的に不可能である。本論文では、仕様を入力・編集は、ステップ1の拡張形情報シーケンス記述に限定しており、非エキスパートのサービス開発者には見せないため、変換された下位のレベルで仕様に手を加えられることはない。このため、下位レベルの仕様と上位の情報シーケンス記述での仕様との関係付けは比較的簡単にできる。

一方、仕様検証では自動変換された下位のレベルであるステップ2やステップ3に対しても実施される。この際、誤りが検出された場合には、サービス開発者が理解できるレベルで通知できなければ、サービス開発者が誤りが存在する仕様上の箇所を知ることができない。つまり、ステップ2、3の検証結果による仕様誤りは、ステップ1の拡張形情報シーケンスレベルに逆変換して表示する必要がある。

以上を考慮し、下位レベルのステップ2、3における仕様検証で検出された誤りに相当する箇所をステップ1の拡張形情報シーケンス図上で表示する、逆変換表示法についてその実現方式を以下に述べる。

### 4.6.2 検証結果の上位仕様逆変換表示法

仕様の自動変換がステップ1→ステップ2、ステップ2→ステップ3と段階的におこなわれているため、仕様誤りの逆変換表示もこの逆方向に2段階で実現できる。

#### ステップ3/ステップ2逆変換表示

ここでは、本来ステップ2/ステップ3変換が1:nであるため、ステップ3のこのn個の記述要素内でどれかが誤りであった場合は、全てこの変換元の1つのステップ2部品に相当する。そこで、ステップ2レベルのこの対応部品を仕様誤り情報とすれば良い。ただし、変換知識データベースにおける変換部品を全て検索すると、検索時間が多く消費し、またステップ3の対象記述要素は他のステップ2部品にも対応するため、一意には決定できない。しかし、本論文における仕様検証法は、変換レベルに対応させた階層形仕様検証法であるため、必ず検証の前に仕様の自動変換を行なっている。そこで、この変換の際、ステップ2とステップ3の記述要素の対応情報（具体的には、ステップ

2の記述要素名+パラメータ値)をステップ3の仕様に付加情報として付与することにより、この情報をステップ2への逆変換表示情報として送出すれば良いことになる。

#### ステップ2/ステップ1逆変換表示

ステップ2/ステップ1の逆変換表示は、部品の記述要素の比が1:1であるため、より簡単である。ステップ3/ステップ2の逆変換表示と同様に、ステップ2/3の自動変換の際、ステップ2の仕様に各記述要素の対応情報を付加情報として付与し、この情報をステップ1への逆変換表示情報として送出すればよい。

以上2段階の逆変換表示を連続させることにより、自動変換した結果のステップ2、ステップ3の仕様における誤り情報をステップ1の拡張形情報シーケンス図上で表示可能となる。

## 4.7 複合検証における検証項目

複合検証は、通信サービス仕様間の矛盾を静的に検出することである。先に示したように、一般にはこのサービス仕様間の矛盾は、サービス競合と呼ばれている。従来のように、通信サービスの種類が限定されている場合には、全ての通信サービス仕様間を総当たりにして、競合の発生する組み合わせを人手により検出することが可能であった。しかし、INのようなカスタマイズドサービス提供のプラットフォームが導入されると、開発する通信サービス仕様の数が増すと共に、伝達網の既存サービスと比べてかなり仕様の複雑度も増加すると考えられ、検証の自動化が要求される。

自動検証する項目を検出するためには、複数通信サービス仕様間での競合の発生要因を明らかにすれば良いと考えられる。主なサービス競合の要因は、以下の4つに分類される。

- (i) 発側と着側のサービス仕様が同一トリガでそれぞれ独立に起動されて両立できない場合、選択手段がないために提供サービスが不確定となる場合。  
例) 発側CCBSサービスと着話中転送サービス
- (ii) 同一入力（フッキング、PB信号等）により指示されるサービス動作が、仕様間で異なり、不確定となる場合。
- (iii) 発／着の各ユーザが起動したサービス仕様で、各ユーザの意思間の相違により論理矛盾が発生する場合。  
例) 着側の転送で発側のスクリーニング条件で制限される人／地域に着信、着信者課金要求と発信者課金要求等
- (iv) 複数の通信サービス仕様が同一の仮想リソースにアクセスし、仮想リソースの状態遷移動作に矛盾を引き起こす場合。

## 4.8 通信サービス仕様間競合検証方式

### 4.8.1 サービス競合検証方式

前節で述べた通信サービス仕様間の競合について、競合発生要因対応に検証方式を述べる。

#### (1) 同一トリガ起動の場合

(i) については、一般的に起動トリガは通信サービス仕様では陽に記述するため、同一起動トリガのサービス仕様は予めリストアップすることは容易である。このリストアップされた各通信サービス仕様間に優先度を付与しておき、新しく仕様が追加される毎に優先度の付与及び変更を行なうことで、起動すべき通信サービス仕様は一つに特定できる。ただし、両仕様間に優先度を送受可能な仕組みが予めサービスプラットフォームの中に必須となる。

#### (2) 同一入力不確定動作及び同一の仮想リソースアクセスの場合

(ii)、(iv) については、両仕様間で同一入力または同一仮想リソースアクセスによる状態遷移があるかどうかを検出すれば良く、手法としては両サービスの仕様を状態遷移図でツリー展開して、同一の入力待ちとなる状態（ツリーではノード）または同一仮想リソースアクセスを実行する遷移枝を検出する。この手法としては、従来より検討されている、到達可能解析法が適用できるため、これを使用する。

#### (3) ユーザ間の意思相違の場合

(iii) については、情報シーケンスには陽に表現されない、網処理機能で表現されるものが中心である。この場合は、特にスクリーニング条件の組み合わせでユーザの意思が表現されるため、スクリーニング条件の全組み合わせをツリー展開して、その葉の部分にサービス仕様中で設定されているサービス機能（CCITTではService Feature と呼ばれる。コールウェイトイング、着信転送、CCBS等）を設定する。この構成のツリーの形式で同一枝の葉について、他の通信サービス仕様間との矛盾を検出することにより、ユーザ間の意思相違を検出することができる。

### 4.8.2 検証対象限定手法

以上の方式のうち、特に(2)と(3)は、カスタマイズドサービス提供においては、一般



的に検証すべき対象の通信サービス仕様の数が膨大になると考えられるため、このままでは現実的ではない。4.3節でも述べたように、予め競合発生の可能性がある仕様の候補を何からの方法で限定した上で、上記手法を適用すれば実用的なものになると考えられる。そこで、競合発生の可能性の候補を検出するためには、以下の方法を考える。

#### 競合可能性判定法

##### (i) サービス競合属性

通信サービス仕様において、サービス競合にとってキーとなる情報（これを以下、サービス競合属性と呼ぶ）を抽出したものを表 4.3に示す。このサービス競合属性の要素のうち、同一の要素が、もし異なる通信サービス仕様間に存在した場合は、両者の仕様の間で競合が発生する可能性が極めて高くなるものである。

##### (ii) 判定法

各通信サービス仕様の自動変換された状態遷移記述の仕様において、特に仮想リソースに対する制御メッセージを基に、上記サービス競合要素を自動抽出する。これは、通信サービス仕様が新規に作成される毎に、競合判定リスト上に追加していく。追加する際、リスト上で同一のサービス競合要素を持つ、既作成済みの通信サービス仕様を検出し、競合可能性候補として選択する。

表 4.3 サービス競合属性の要素

要素	意味
課金条件	発信者課金、着信課金、 $\alpha$ 課金等
接続タイプ	①論理サービス—仮想リソースの制御なし
	②単純接続制御—1回の接続制御
	③接続形態変更制御
着信先変更	着信転送先のアドレスデータの変更
フッキング信号受信	アナログでのサービス切り替え指示信号
ファシリティ信号送受	ISDNでのサービス切り替え指示信号

## 4.9 検証方式の試作と評価

以上述べてきた、検証方式のうち、単体検証において新たに開発した段階的検証方式について試作システムを作成し、評価を行なった。

### 4.9.1 段階的検証方式試作システム

段階的検証方式を適用した検証システムをSUN3上に試作した。検証システムは、ステップ1の言語形式によるサービス仕様に対して、1つずつプリミティブを読み込む、仕様入力部とルールに基づき段階的検証を実施する検証処理部、及び検証結果をシーケンス図上に検証箇所とエラーメッセージと共に表示する、検証結果出力部により構成される。これを図4.10に示す。また、この検証結果の表示例を図4.11に示す。

### 4.9.2 試作結果と考察

#### (1)検証能力

上述の段階的検証システム上で、図4.12に示すような実験方法で以下の項目について測定した。

$E_S$ : 検証システム上で検出される誤りの個数

$C_S$ : 検証システム上で誤りがなくなるまでに要する修正の繰り返し回数

$E_M$ : 検証システム上での誤り検出終了後、人手で検出される誤りの個数

対象としたサービス仕様は、特定のサービスを記述したものではなく、プリミティブをランダムに並べたものを使用している。また、プリミティブは、一般通話と標準的な付加サービスを記述するために必要なものを用いた。プリミティブの種類と検証規則の数は以下の通りである。

プリミティブの種類: 38

検証規則数: 59

#### 修正の回数と検出される誤りの割合

修正の繰り返し回数と検証システムで検出される誤りの割合との関係について、いくつかのサービス仕様について実験した結果を図4.13に示す。修正の繰り返し回数が1回でないのは、修正により追加、削除、変更されたプリミティブに関して新たに誤りが検

出されるためである。1回目の検証で大半の誤りは検出されており、その後も4回目までの修正ですべての誤りが検出された。検証システム上での誤り検出終了後におこなった人手による誤り検出では、検証システムでは検出できなかった誤りが検出された。

#### 検証能力

本検証システムの検証能力 $\delta$ を、通信サービス仕様に含まれる誤りのうち、検証システムで検出できるものの割合で評価するものとし、次式で算出した。

$$\delta = E_S / (E_S + E_M)$$

検証実験結果による $\delta$ の値を図4.14に示す。この結果、誤りの大部分が検証システム上で検出されることが確認できた。プリミティブの仕様上の絶対的位置に対する知識による規則を追加することにより、さらに検証能力が向上可能である。

#### (2)検証に要する時間

検証システム上で検証を起動させてから終了するまでの時間を測定した。記述量と時間の関係を図4.15に示す。測定では、従来作成した通信サービスで、最も複雑な仕様である電話会議サービスに相当する基本通話の40倍の記述量の場合で約4.4秒であり、通常のサービスではこれ以下で検証できることが示された。



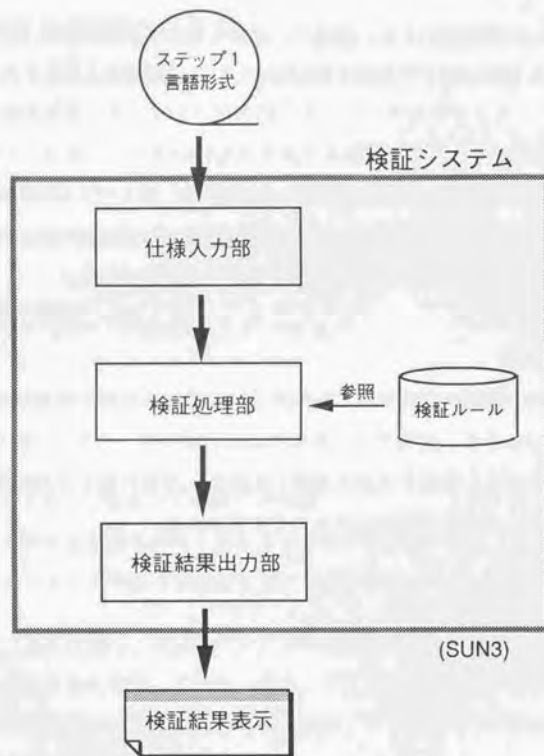


図 4.10 検証システムの構成

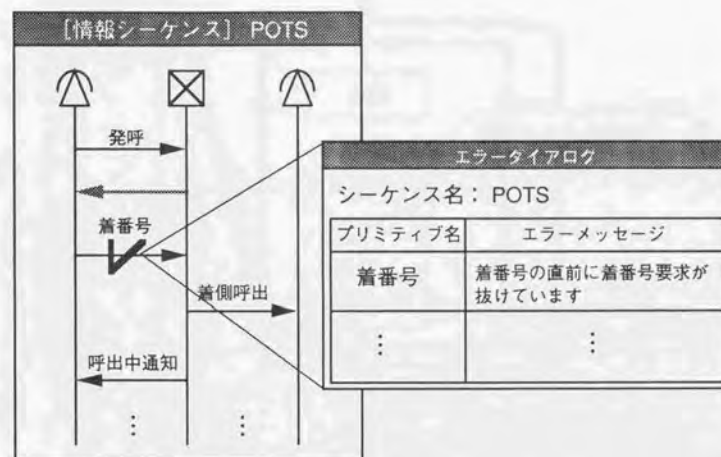


図 4.11 検証結果の表示例

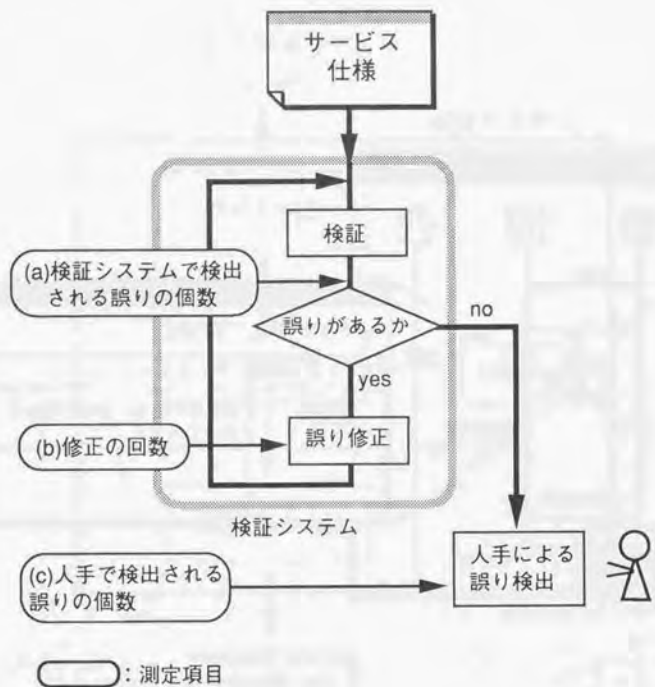


図 4.12 試作システムによる検証実験方法

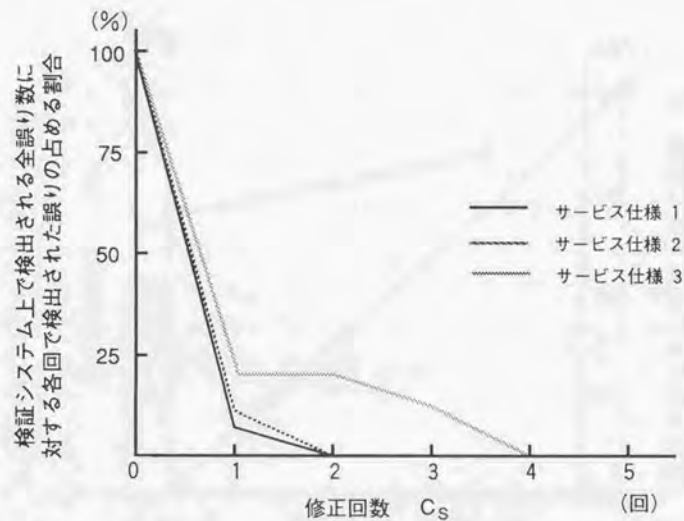


図 4.13 修正の回数と検出される誤りの割合



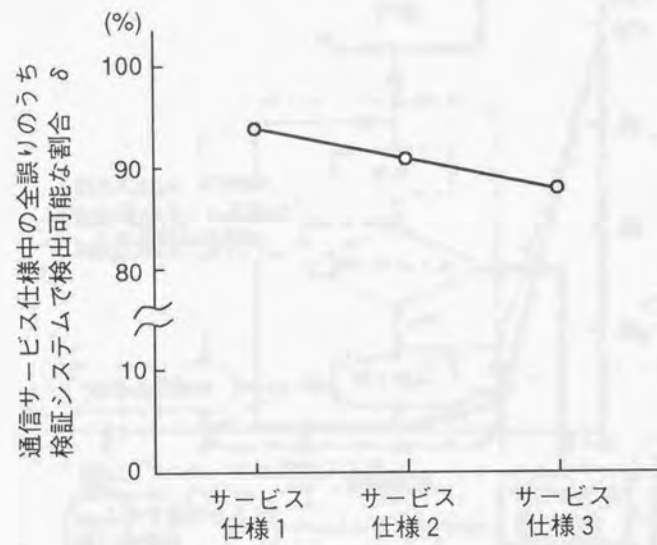


図 4.14 検証能力の評価

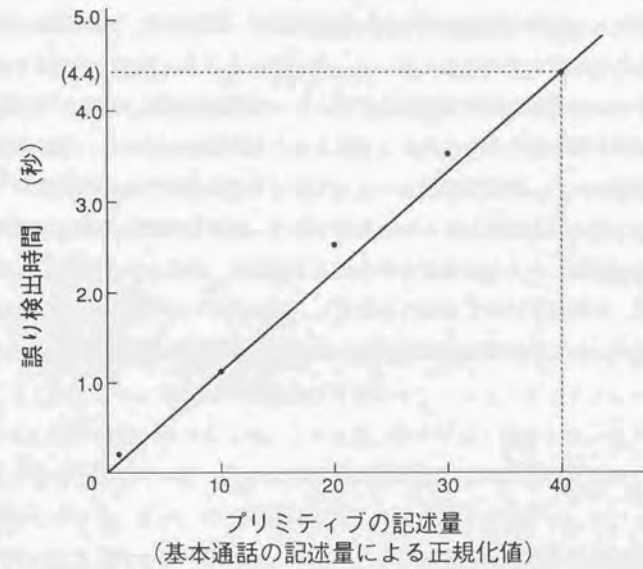


図 4.15 検証に要する時間

## 4.10 むすび

通信サービス仕様の検証技術のうち、次の2つの主要技術課題について検討結果を述べた。

- (1) 階層形仕様検証法
- (2) サービス手順検証法

まず最初に、検証の深さの観点から、構文検証、論理検証、意味検証、価値検証の4つがあることを明らかにした。

記述レベルに応じた階層形検証法では、サービス手順検証、プロトコル手順検証、リソース制御手順検証の3つのステップがあることを明らかにした。このうちのサービス手順検証について、拡張形情報シーケンスをマクロに見たマクロ化情報シーケンス検証とミクロに見た遷移手順検証との2段階で行う、段階的検証方式を提案した。両検証法について、検証ルールと検証アルゴリズムを開発し、検証システムの試作を行なった。この結果、本検証方式の正常動作を確認し、通常のサービス仕様はSUN3ワークステーション上で、約4秒以下で検証可能であることを明らかにした。

## 第5章 通信サービス仕様の試験・評価技術

### 5.1 まえがき

第4章の通信サービス仕様の検証技術は、論理的な誤りを静的に検出するものであるが、静的な仕様検証だけで全ての誤りが検出されるという保証はなく、動的な面で発生する誤りは、実際に記述した仕様を動作させて調べる必要がある。また、サービスの要求仕様との整合性という、より深いレベルの検証は自動仕様検証では困難であり、実際にサービス動作を人間が確認することで初めて、サービスの要求仕様との整合性が保証できたとと言える。これらは、従来の通信ソフトウェア開発における試験工程のうち、通信サービス仕様に関するものの試験工程であると言えるが、実際のターゲットマシン用のソフトウェアの製造の前に試験を行なう点が新しいと言える。これは、情報処理の分野で最近盛んになっている、ソフトウェアのラビッドプロトタイプング技術と言えるものであり、通信サービスソフトウェアに対するラビッドプロトタイプング方式を、本論文では新たに開発する。

一方、従来の通信サービスの開発は、サービス提供側の立場からサービスの機能中心に検討されてきており、ユーザに対する操作性の高いマン・マシンインタフェースの提供という点からの検討が不十分であった。この結果、ユーザは、高度なサービス機能の良さ、便利さを味わう前にサービス自身に拒否感を持ってしまい、利用を止めてしまうという事態が生じていた。また、サービス提供後にユーザの声を受けて、マン・マシンインタフェースの改善を含むサービス仕様の改善を行なうための手戻りが多く発生し、開発工数を増加させていた。

また、通信サービスのマン・マシンインタフェースは、ディスプレイの入出力を前提としたサービスについては検討が進んでいるが、ダイヤルボタンの入力と可聴音信号、音声による出力のみにより高度な通信サービスを実現しなければならない、電話系のサービスについては従来ほとんど検討されていなかった。このため、電話系サービスにおけるマン・マシンインタフェースの設計では、机上検討でガイドラインもなく、各設計者の思い付きに頼って検討が行なわれていた。しかし、本論文で提案するサービス呼制御方式でも最初は電話系サービスが核であり、操作性の高い誰でもが使いやすいと感じるマン・マシンインタフェースの設計が必須である。

本章では、通信サービス仕様の試験技術として、ラビッドサービスプロトタイプング



方式とこれによるサービスプロトタイプシステムの構成法、及びこのサービスプロトタイプシステムを用いたサービス仕様評価技術について述べる。

## 5.2 通信サービス仕様の試験・評価技術の概要

通信サービス仕様の誤りのうち、仕様検証で検出されない誤りやサービス要求仕様との整合性をチェックするための試験技術、及び操作性の高いマン・マシンインタフェースを中心とした評価技術を明らかにする(図5.1)。通信サービス仕様の試験は、前章でも述べたように、仕様検証における意味検証に相当し、評価は一番レベルが深い価値検証に相当するものである。

通信サービス仕様の試験技術については、サービス仕様レベルで動的な試験が可能となる、ラビッドプロトタイプ方式の実現技術として、以下の(1)の課題を明らかにする。また、通信サービス仕様の評価技術については、比較的定量的評価がし易く、改善効果も高く、通信サービス仕様の品質に影響が大きい、マン・マシンインタフェースについてのみ、本論文では検討を行なう。マン・マシンインタフェース以外の評価項目については、マン・マシンインタフェースのモニタ評価における感想聴取により取得できると考えられるため、ここでは検討の対象としない。マン・マシンインタフェース設計における問題点を分析することから、(2)から(4)の3つの技術課題について実現技術を明らかにする。

- (1) 通信サービス仕様の試験環境構成法
- (2) マン・マシンインタフェースの評価尺度・分析手法
- (3) ユーザによるマン・マシンインタフェースモニタ評価方法
- (4) マン・マシンインタフェース設計のガイドライン

(4)のマン・マシンインタフェース設計のためのガイドラインは、(3)のモニタ評価方法に従い実際の被験者によるモニタ評価結果をもとに作成する。

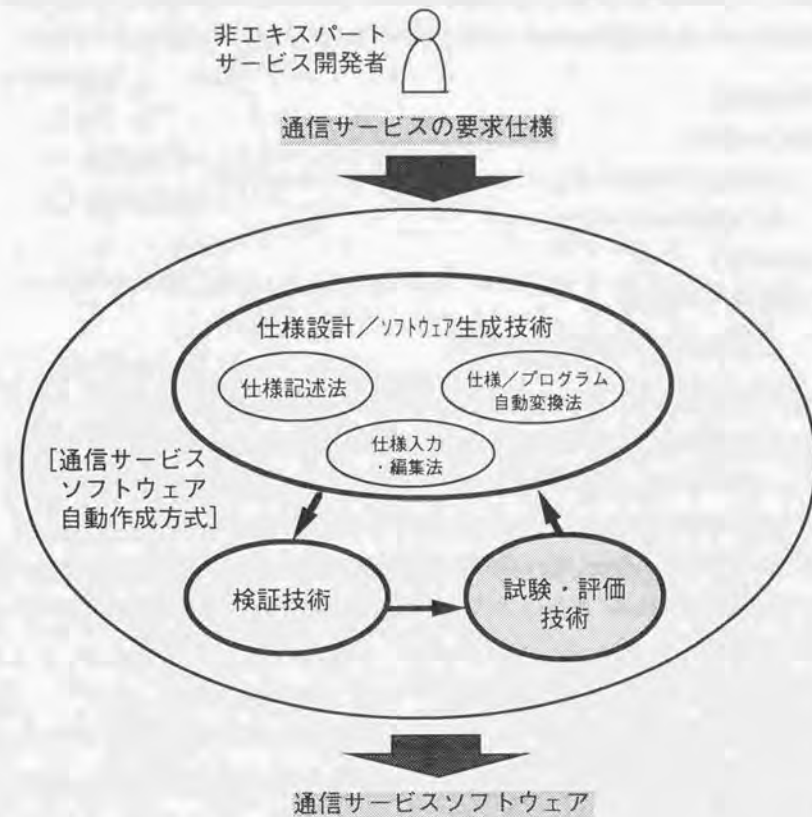


図 5.1 通信サービス仕様の試験・評価技術

## 5.3 通信サービス仕様の試験法

### 5.3.1 概要

通信サービス仕様の試験では、仕様検証では検出できない、以下の2レベルの検証項目に相当するものが対象となる。

#### [論理検証]

##### ○動作の正常性

- ・仕様検証で論理的に正しいと判定されたが、動作不良の検出
- ・準正常動作の漏れの検出

#### [意味検証]

##### ○要求仕様との整合性

- ・サービス要求との非整合動作の検出
- ・システムによる余分なサービス動作(要求外動作)付加の検出

これらは静的な検証では実現困難であるため、動的な試験により実現する必要がある。しかし、従来の試験は、作成されたプログラムを走行させてその正常性をチェックするものであり、この場合のように仕様を直接走行させて試験する方法は検討されていない。このため、仕様試験の環境を新たに構成する必要があり、この構成法を以下に示す。

一方、上記2つの検証に加えて、同様に仕様検証では実現されない検証として以下の価値検証がある。

#### [価値検証]

##### ①利便性、有効性の検証

- ・提供サービス機能が便利で効果的なサービスであるかどうかのオピニオン評価

##### ②操作性の検証

- ・提供するサービスのマン・マシンインタフェースの使い勝手が良いかどうかのオピニオン評価

価値検証は主観的評価であり実際のサービスユーザが評価しなければ、サービス開発者だけの思い込みで決定されてしまい、できるだけ多くのユーザによる評価が求められる。このため、価値検証に相当する評価法の開発は、ユーザによる評価法を確立することに

あり、5.5節以降に通信サービス仕様のマン・マシンインタフェース評価法として述べる。

### 5.3.2 通信サービス仕様試験環境の構成法

通信サービス仕様の試験環境を現状の技術で実現することを考えると、次の2つが挙げられる。

- ①実マシンによるプロトタイプ実験
- ②計算機シミュレーション

これらはいずれも机上検討よりは優れているが、それぞれ以下の欠点を持っている。

#### (①の欠点)

いずれの検証項目も実現可能であるが、プロトタイプの作成を支援する環境がなく、また作成の際実マシンの知識が必要となり、簡単かつ迅速な仕様検証とならない。

#### (②の欠点)

いずれの試験項目も実現可能であり、かつ汎用計算機の持つソフトウェア開発支援環境をそのまま利用できるため、シミュレーションも簡単に実現可能である。しかし、シミュレーションであるため実際のハードウェアが動作せず、サービスの具体的なイメージがつかみにくいため、サービスの有効性及び操作性の評価は十分といえない。特に、ユーザによる評価を行う場合は実際のサービス動作が必須と言える。

上記2つの手法の分析より、通信サービス仕様の試験には次の2点が要求される。

#### (要求1)

サービス仕様がダイナミックにハードウェアを伴う動作をし、ユーザからは本物のサービスが動作しているように見え実際に体験できる。

#### (要求2)



上記動作を実現するサービスのプロトタイプ作成を支援する環境があり、簡単かつ迅速な仕様検証ができる。

上記の要求を満たす試験環境は、マン・マシンインタフェースを提供するための交換装置と、サービス仕様を簡易記述し交換装置を制御する汎用コンピュータで構成される。また、この構成による試験方式は、各種の通信サービス仕様のプロトタイプを迅速に作成し動作を実現して体験できることから、情報処理分野で実現されているラビッドプロトタイピングの概念を考慮して、ラビッドサービスプロトタイピング方式と呼ぶ。また、本方式を実現する構成をサービスプロトタイピングシステムと呼び、図 5.2 に示す。以下、サービスプロトタイピングシステムの構成法を述べる。

### 5.3.3 サービスプロトタイピングシステム構成法<sup>[34]・[37]</sup>

#### 5.3.3.1 汎用コンピュータ呼制御方式

サービスプロトタイピングシステム上で作成するサービス仕様はプロトタイプであるが、ダイナミックにハードウェアを伴う動作をし、本物のサービスが動作しているように見える必要があり、ある程度のハードウェア動作に対する実時間性の確保が必要である。また、この際汎用コンピュータの開発支援環境、走行環境の良さを失わないように考慮する必要がある。

一方、サービスプロトタイピングシステム上でサービス呼制御を実現するためには以下の処理機能が必要となる。

- ①リソースの状態監視、通知処理
- ②リソースの管理処理
- ③呼状態遷移の制御処理
- ④リソースの駆動処理

①の処理は、実時間性の強い処理であり、汎用OS下におけるソフトウェア制御では実現しにくい。また、④の処理はリソース駆動のための制御情報を汎用コンピュータから交換装置に対して、逐一細かいレベルで送出していたのでは、I/Oの転送速度等の制約から実時間性が保証されない。そこで、交換装置には上記①と④の実時間性の高い処理機能のみを与え、受動的な動作をする装置とし、実質的な呼制御処理機能である②と③の機能は汎用コンピュータに付与することにより、汎用コンピュータを制御の主体と

した機能分担とする(図 5.3)。

#### 5.3.3.2 呼制御処理機能と交換装置・汎用コンピュータ間インタフェース

交換装置と汎用コンピュータそれぞれに付与する具体的呼制御処理機能と交換装置・汎用コンピュータ間インタフェースを示す。交換装置・汎用コンピュータ間インタフェースは、交換装置から汎用コンピュータへの制御信号は、状態変化通知メッセージと呼び、汎用コンピュータから交換装置への制御信号は制御動作指示メッセージと呼ぶ。

##### (1) 交換装置の呼制御処理機能

###### ①リソースの状態監視、通知処理

周期監視により、以下のリソースの状態変化を検出し、状態変化通知メッセージとして組み立てる。

- ・加入者(加入者回路)：オフフック、オンフック、フッキング
- ・PB信号レシーバ：ダイヤル数字受信
- ・発着トランク：応答、着信

状態変化通知メッセージの例を表 5.1(a)に示す。

###### ②リソースの駆動制御処理

制御動作指示メッセージを必要な部品(各リソースの駆動ルーチン)に展開し、各対応リソースを制御する。制御動作指示メッセージの例を表 5.1(b)に示す。

###### ③汎用コンピュータとのメッセージ送受

状態変化通知メッセージを汎用コンピュータに送出し、汎用コンピュータからの制御動作指示メッセージを受信する。

##### (2) 汎用コンピュータの呼制御処理機能

###### ①リソースの管理処理

汎用コンピュータ内の状態遷移制御処理部からのリソース捕捉要求により、個別リソースの捕捉、共通リソースの割当を行う。また、同様にリソース解放要求により、個別リソースの解放、共通リソースの割当解除を行う。

###### ②状態遷移制御処理

交換装置からの状態変化通知メッセージを受けて、対象とする通信サービスソフトウェアに基づき状態遷移制御を実行し、それに伴う遷移タスクを制御動作指示メッセージとして交換装置に送出する。



### 5.3.3.3 処理ソフトウェア構成法

#### (1) プロセス構成

汎用OSの提供する並行プロセス走行環境を用い、各処理機能を以下の個別のプロセス構成で実現した。この構成を図 5.4に示す。

##### ①交換装置インタフェースプロセス (NIP: Network Interface Process)

交換装置と制御装置のインタフェースのための処理で、上述の状態変化検出メッセージを交換装置より受信し、駆動制御指示メッセージを交換装置へ送出する。

##### ②リソース管理プロセス (RMP: Resource Management Process)

プロセスを含めたリソースの捕捉/解放、共通リソースの割当/解除のための処理を行う。

##### ③サービス制御プロセス (SCP: Service Control Process)

サービス仕様に基づくサービスソフトウェアを記述し、これに基づく状態遷移制御処理を行う。具体的な状態遷移制御処理は、状態変化検出メッセージを受けて状態遷移し、それに伴うリソース制御処理を駆動制御指示メッセージとして、交換装置へ送出する。

##### ④運転管理プロセス (OMP: Operation Management Process)

保守者とのマン・マシンインタフェースを持ち、運用情報の収集、サービスオーダーの投入等の処理を行う。

#### (2) サービス呼の1コール1プロセス処理方式

並行プロセス走行環境を取り入れたことにより、同時複数制御動作が可能となるため、制御対象とするリソースを限定することによるサービスソフトウェア記述の容易性を図り、サービス制御プロセス(SCP)は、1プロセスが1コールのみを取り扱う、1コール1プロセス処理方式とする。具体的には、呼の発生によりプロセスが生成されてその呼に割り当てられ、呼の消滅によりその割り当てられているプロセスを消去する。このイメージを図 5.5に示す。これにより、従来、通信サービスの仕様を設計する際に使用してきた1コールスルーの仕様をそのまま書き下したサービスソフトウェアを、そのままSCP上に適用することで、サービス動作が実現される。

#### (3) ハードウェアリソースの仮想化表現

交換装置内のハードウェアリソースは、全て端子アドレス(端子内通番)で表現し、

リソース物理名と呼ぶ。これは、汎用コンピュータからの制御がリソースの種別を意識せずに統一的な形式で可能としたためである。汎用コンピュータ内では、リソースについて以下の点を考慮する。

- ①リソースの物理的配置、構成は意識せず、使用可能なリソースの種別、個数のみを論理的リソースとして管理する。
- ②RMPがリソースを集中して種別毎に管理する。
- ③SCPでは呼に関係するリソースのみを管理。

これより、メッセージに伴ってNIP経由で入力されたリソース物理名は、まずRMP内で仮想化されてリソース共通論理名(リソース種別名、リソース種別内通番)に変換される。これは、SCP、OMPに対してもこのリソース共通論理名が使用される。さらに、サービス制御プロセス(SCP)では、サービスソフトウェアの記述性、理解性の向上のため、対象とするリソースのみに限定して表現されることが望ましい。このため、SCP内のRMPとのインタフェース処理部において、リソース呼対応論理名(リソース種別名\_呼内識別子)に変換する。この例を表 5.2に示す。

### 5.3.3.4 サービス制御プロセス(SCP)の構成法

#### (1) SCPの処理構造

SCPの処理は図 5.6に示すように、大きく分けてリソース管理プロセス(RMP)とメッセージの入出力を行うRMPインタフェース部とサービスソフトウェアに基づく呼の状態遷移制御を行うサービス制御部とで構成される。SCPの処理は、通常休止状態にあるが、RMPからのオーダー受信かタイムアウトで割り込みがかかり、RMPインタフェース部でリソースの共通論理名/呼対応論理名変換を経た後、サービス制御部が駆動される。サービス制御部からメッセージの送出を行うときは、RMPインタフェース部を駆動し、リソースの呼対応論理名/共通論理名変換後、RMPあてのメールボックスへ書き込みを行う。

#### (2) SCPにおけるサービスソフトウェア記述

上述のサービス制御部は、具体的にはサービスソフトウェアが記述される。ここでは、従来より通信サービスソフトウェアの設計に一般的に用いられている状態遷移記述を図 5.7のように作成し、これをC言語で表現することによりサービスソフトウェアを記述する手法を開発した。

##### ①状態遷移図と状態遷移処理構造



図 5.7中の各枝に記述されているパラメータを伴ったオーダは、+（オーダ）がRMPからの受信オーダであり、（オーダ）がRMPへの送出オーダを示している。また、符号なしのラベルはダイヤル分析やタイマ処理の共通部品を示している。この状態遷移図に示された制御を処理構造で表すと図 5.8のようになる。ここで、状態遷移図中の状態名は状態番号に置き換えている。

### ②C言語状態遷移記述によるサービスソフトウェア記述法

図 5.8の処理構造は、C言語のswitch、case文を用いて状態番号で判定・分岐させるswitch文の中に受信オーダの判定・分岐させるswitch文を入れ子にした構造で、簡単に表現できる（状態遷移手順と呼ぶ）。また、受信オーダのテーブル（分析テーブルと呼ぶ）とRMPへのオーダ送出テーブル（タスクテーブルと呼ぶ）を配列データ定義として表すことにする。一方、共通部品はC言語記述の関数として作成し、状態遷移記述内で通常の関数呼び出しとして用いる。図 5.7の状態遷移図をC言語によるサービスソフトウェアとして記述した例を、状態遷移手順を図 5.9に、分析及びタスクテーブルを図 5.10 に示す。

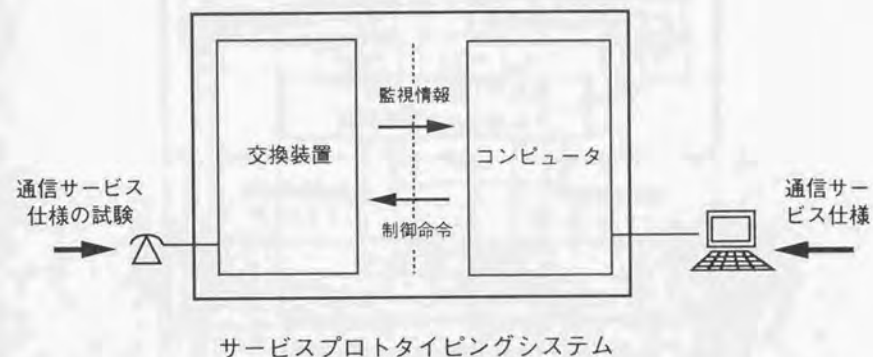


図 5.2 通信サービスのラピッドプロトタイピング方式

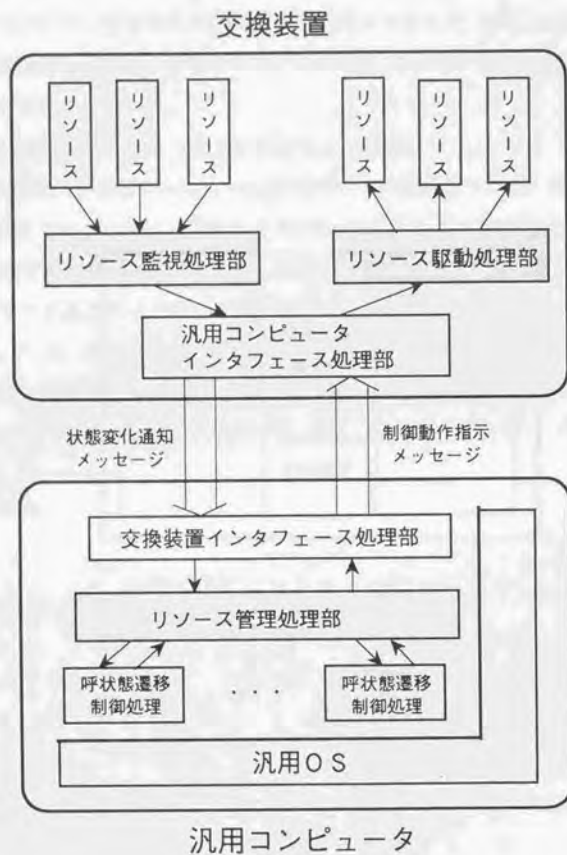
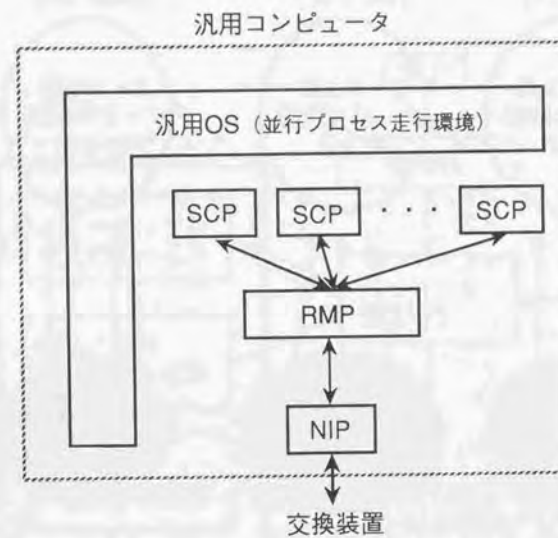


図 5.3 汎用コンピュータ呼制御方式



- SCP : サービス制御プロセス
- RMP : リソース管理プロセス
- NIP : 交換装置インタフェースプロセス

図 5.4 処理ソフトウェア構成



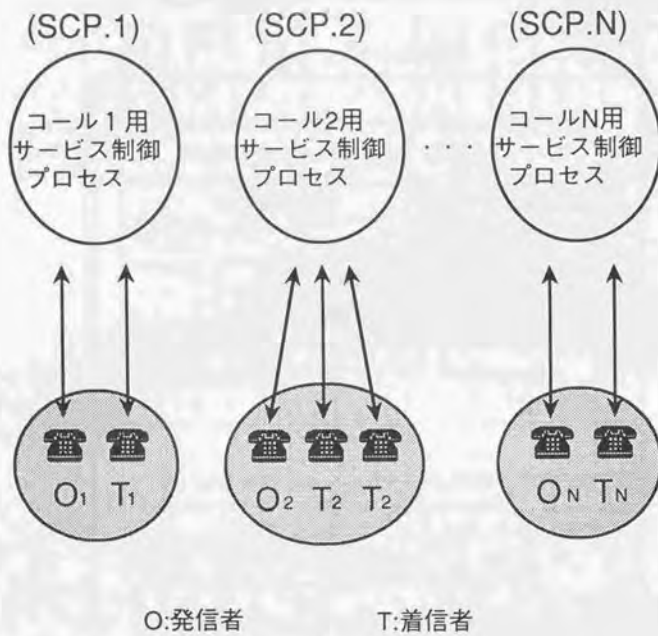


図 5.5 1コール1 プロセス処理方式のイメージ

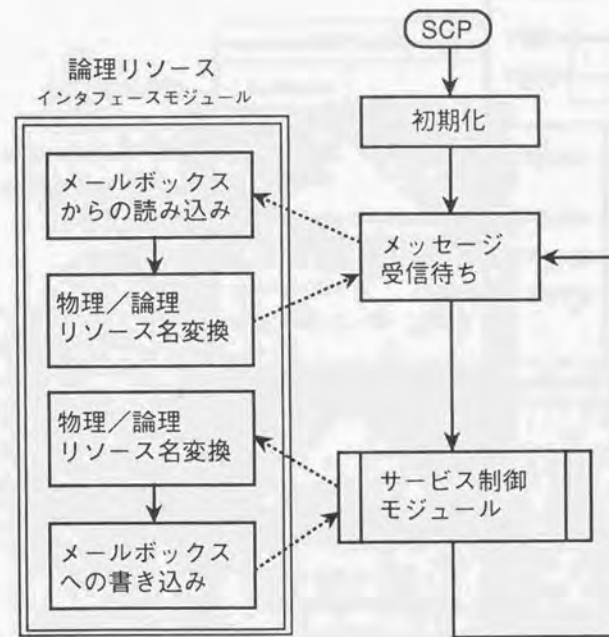


図 5.6 SCPの処理構造

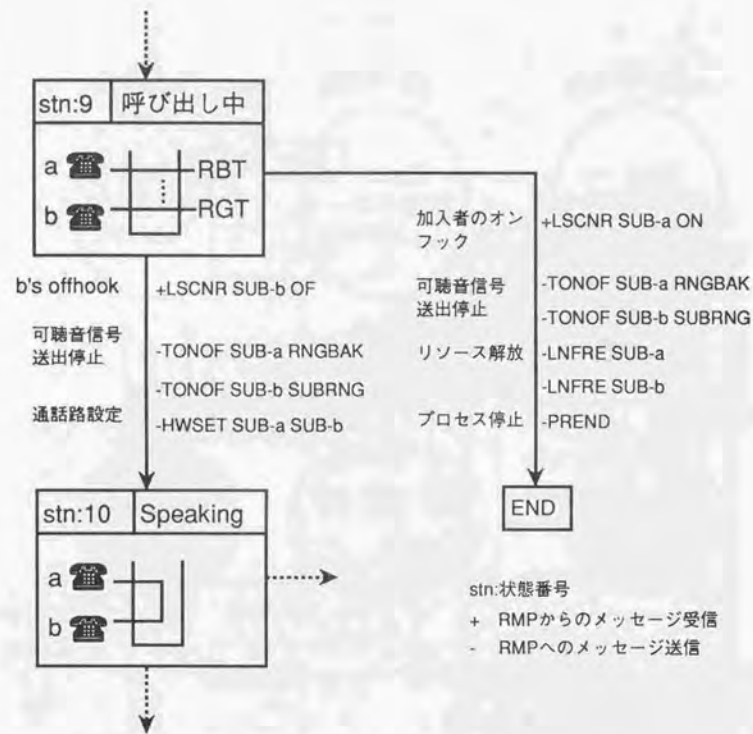


図 5.7 状態遷移図例 (呼出中から通話中への遷移)

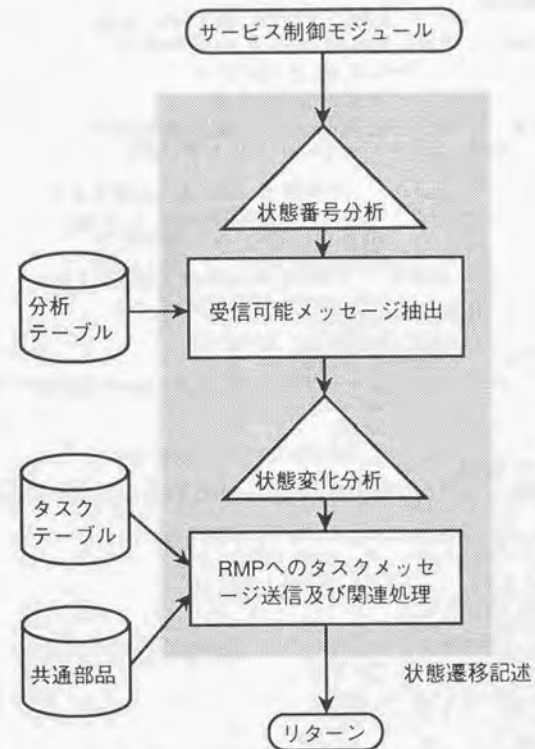


図 5.8 状態遷移処理構造



```

scp_std ()
{
    switch (stn)
    {
        case 0: /* Idle; Waiting for SUB_a's offhook */
            .
            .
        case 9: /* Calling; Waiting for SUB_b's offhook */
            switch (input_order (INORDER_BUF, st9))
            {
                case 0: /* SUB_b's offhook: TASK 9_0 */
                    task_routine (tsk9_0);
                    stn = 10; /* Next state number */
                    break;
                case 1: /* SUB_a's onhook: TASK 9_1 */
                    task_routine (tsk9_0);
                    stn = -1;
                    break;
                default: /* ERROR */
                    perror ("### ERROR !!! stn = 9 ###");
                    stn = -1;
                    break;
            }
        case 10: /* Speaking; Waiting for SUB_a's or SUB_b's onhook */
            .
            .
    }
    if (stn < 0)
    {
        process_end (); /* Calling for SCP end routine */
    }
}

```

図 5.9 C 言語状態遷移記述法によるサービス制御手順記述—状態遷移手順

(1) 分析テーブル

```

char st9 [ ] [RORDER_MAX] =
{
    "LSCNR SUB_b OF",
    "LSCNR SUB_a ON",
    "x",
};

```

(2) タスクテーブル

```

char tsk9_0 [ ] [SORDER_MAX] =
{
    "TONOF SUB_a RINGBAK",
    "TONOF SUB_b SUBRNG",
    "HWSET SUB_a SUB_b",
    "x",
};
char tsk9_1 [ ] [SORDER_MAX] =
{
    "TONOF SUB_a RINGBAK",
    "TONOF SUB_b SUBRNG",
    "LNFRE SUB_a",
    "LNFRE SUB_b",
    "x",
};

```

図 5.10 C 言語状態遷移記述法による通信サービスソフトウェア記述—分析及びタスクテーブル

表 5.1 交換装置・汎用コンピュータ間インタフェース

(a) 状態変化通知メッセージ例

メッセージ	意 味
L S C N R	加入者の変化情報通知 (オフフック、オンフック等)
T S C N R	トランクの変化情報通知 (応答、着信等)
D I R E C	受信ダイヤル数字の通知

(b) 制御動作指示メッセージ例

メッセージ	意 味
H W S E T	指定リソース間の通話路設定
H W R E L	指定リソース間の通話路解除
T O N O N	指定リソースへの指定トーンの送出開始
T O N O F	指定リソースへの指定トーンの送出停止

表 5.2 ハードウェアリソースの仮想化表現

リソース	リソース物理名	リソース共通論理名	リソース呼対応論理名
加入者	8	SUB.08	SUB_a
	12	SUB.12	SUB_b
PB信号受信回路	179	PBR.03	PBREC_1
三者通話トランク	160	TWT.00	TWTRK_1



## 5.4 通信サービス仕様試験システムの試作と評価

前節で提案したサービスプロトタイプシステムによる試験システムを試作し、評価を行った。

### 5.4.1 試作システムの構成

交換装置の主な構成要素を以下に示す。本論文での対象は、主に電話系の通信サービスであるため、電話系サービスを実現するための装置構成となっている。

- (i) 交換装置内処理用プロセッサ
- (ii) 通話路スイッチ
- (iii) 加入者回路
- (iv) トランク・・・発着トランク、三方路トランク、トキトランク
- (v) 可聴信号発生回路（リングング回路も含む）
- (vi) 可聴信号受信回路（ここではPB信号レシーバ）
- (vii) 音声応答、蓄積装置

具体的には、NTTの小型PBXであるEP-12（最大96加入者収容）を改造して実現した。また、汎用コンピュータはDEC社のワークステーションで、汎用OSとしてVMSを搭載しているVS-3500とSunMicro Systems社のワークステーションで汎用OSとしてUNIXを搭載したSUN3/470のどちらでも動作可能とした。交換装置と汎用コンピュータとの接続は、16bitパラレルのDMAインタフェースによるバス接続とし、汎用コンピュータがどちらの場合でも全く同一のインタフェースなるように設定した。以上の構成を図5.11に示す。

### 5.4.2 試作結果

#### (1) 制御ソフトウェア

交換装置内のソフトウェアについては、8ビット・マイクロプロセッサのアセンブラ言語を用いて約4Kステップで実現した。一方、汎用コンピュータ内のソフトウェアは表5.3に示す規模（SCPの結果は(2)で詳細に示す）で実現した。特に、オーダ送受については、NIPでは17オーダを処理し1オーダ当たり約35行を要している。また、RMPでは35オーダを処理し、1オーダ当たり約107行を要している。RMPがNIPに比し、約3倍の処理量を要しているのは、NIPはオーダ送受制御が主な機能である

のに対し、RMPはそれに加えてリソース管理機能を実現しているためである。

#### (2) サービスソフトウェア記述

表5.4に示した各通信サービスの通信サービスソフトウェアを記述するのに要した期間を記述量（状態遷移記述+分析テーブル+タスクテーブル+各サービスで新規に作成した共通部品）と共に図5.12に示す。全て14日以内で実現されており、作成量（C言語プログラムの行数）に比し、かなり短期間で実現できている。これは、サービス制御手順の記述をSCPの記述で実現できるようになったことで、記述性が向上したと評価される。また、短期間で作成量が多いことは、機械的に記述できる部分が多いことを示しており、言い換えると通信サービスソフトウェア作成の自動化が実現しやすい条件が整備されたと考えられる。

### 5.4.3 サービスプロトタイプシステムを用いた試験法の効果

サービスプロトタイプシステムを用いた、通信サービス仕様の試験法の効果について、本方式を通信サービスソフトウェア開発に適用した際の開発期間短縮効果で見積ることとする。

通信サービス仕様の試験法が開発工程に与える効果は主として従来の試験工程の短縮であり、ここでは試験工程の期間短縮だけを評価する。開発期間短縮効果を $\rho$ とすると次式で表現される。

$$\rho = C_T \cdot \alpha (1 + \beta) - C_p$$

ただし、

$C_T$ : 全開発期間に対する試験工程期間の割合

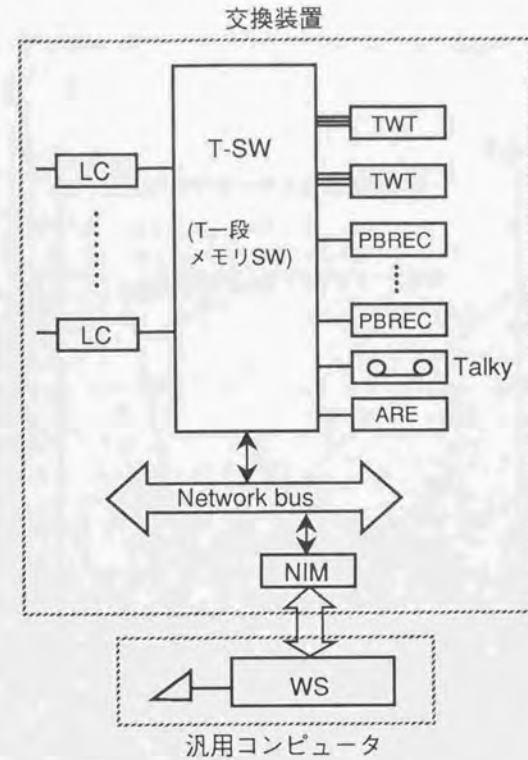
$C_p$ : 全開発期間に対する動的仕様検証に要する期間の割合

$\alpha$ : 試験工程で検出されるバグのうち、サービス仕様に依存するバグの比率

$\beta$ : 試験工程での仕様変更に伴う、変更箇所以外への波及効果

$C_T$ をサービス実用化における経験値から0.5、また同様に全開発期間は2年程度とすると、動的仕様検証の経験により2週間以内で実現可能な見通しより、 $C_p$ を0.02とすると、また、検出バグのサービス仕様依存比率 $\alpha$ は、一般的に10%以下であることを考慮して、0.03~0.09で変化させる。また、仕様変更に伴う波及範囲 $\beta$ は、手戻りによる影響の大

さを考慮し、0.5~2.5まで変化させる。この場合の開発期間短縮効果 $\rho$ の値を図5.13に示す。これにより、従来のサービス開発工程にそのまま適用しても、開発期間は約15%程度まで短縮が図れることが期待される。



- LC : 加入者回路
- TWT : 三方路トランク
- PBREC : PB信号受信回路
- ARE : 音声応答装置
- NIM : 交換装置インタフェースモジュール (プロセッサ)
- WS : ワークステーション

図 5.11 試作システムの構成



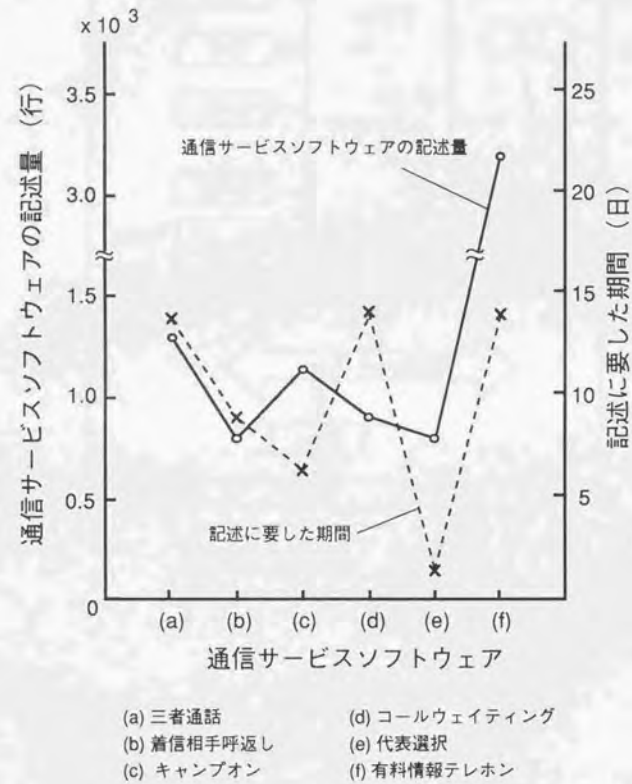


図 5.12 通信サービスソフトウェア記述量と記述に要した期間

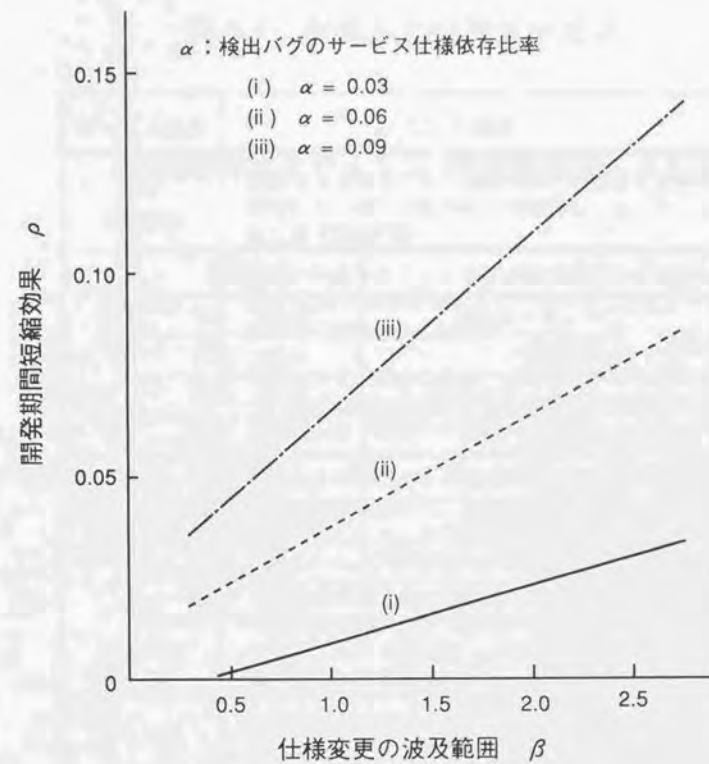


図 5.13 開発期間短縮効果

表 5.3 交換装置内 ソフトウェア作成規模

プロセス		作成規模
NIP	メッセージ送受	0.59 K行
	共通処理	0.30 K行
RMP	メッセージ送受	3.76 K行
	共通処理	0.19 K行
	リソースデータ	1.07 K行
OMP	メッセージ送受	0.32 K行
	表示制御・入出力制御	1.56 K行
	表示データ	0.21 K行

表 5.4 記述した付加サービス

サービス種別	サービスの概要
(a) 三者通話	発信者 a が着信者 b と通話中に、発信者 a が b を保留にし、第二の着信者 c を呼出し、a、b、c の三者で通話可能
(b) 着信相手 呼び返し	直前の不完了となった通話または完了した通話の相手番号を3桁の特殊番号をダイヤルすることにより、自動的に呼返すことが可能
(c) キャンプオン	相手加入者が話中の場合、網に登録することにより、網が相手加入者の空きを検出し、自動的に発着両加入者を呼出して接続
(d) コールウェ イティング	発信者 a が着信者 b と通話中に、別の加入者 c よりの着信が通知され、b を保留にした後、c と通話が可能となり、以下交互に切替通話可能
(e) 代表選択	予め複数の加入者が選択順序を付与されてグループとなり、かつ先頭の番号が代表番号として公にされ、着信があった場合に話中遭遇により順次選択して接続
(f) 有料情報 テレホン	情報提供者の提供するテレホンサービスを通話料金に加えて、情報料金を代行して徴収する。情報聴取後、情報提供者の直接呼出しやメッセージ録音がメニュー選択可能



## 5.5 通信サービス仕様におけるマン・マシンインタフェース評価法

### 5.5.1 通信サービス仕様におけるマン・マシンインタフェース設計の問題点

#### 5.5.1.1 電話系サービスに内在する問題点

電話系サービスは、そのユーザのほとんどがアナログの電話機を用いて提供されるサービスであり、サービスを実現するための制御情報の入出力の点で大きな制約が課せられている。入力は0～9と\*、#の12個のPBダイヤルボタンだけであり、出力は可聴音信号と音声ガイダンスのみである。これがマン・マシンインタフェースに与える影響を表5.5に示す。

#### 5.5.1.2 電話系サービスの開発における問題点

電話系サービスの開発における問題点は、既開発済みのサービスから見ると以下の2つがあげられる。

- (i) ユーザにとって受け入れられないサービス仕様
- (ii) 操作性を改善するための手戻りによる開発工数の増加

(i) については、特にそのマン・マシンインタフェースが難解であると思われる電話会議サービスについて、契約ユーザを対象に行なったアンケート調査の56名の回答より得られた結果より、以下の点が明らかとなった。

- ・評価 68%が使いにくい
- ・要望 ①操作手順の簡略化  
②誤りのチェック機能  
③ガイダンスのコンパクト化

(ii) については、通信サービスの制御ソフトウェアバージョンアップに伴う変更である。電話会議サービスにおける一回目のソフトウェア変更においては、変更分のうちサービス仕様の変更に関するものが約63%も占め、このうちの約34%（全体では21%）が操作性に関するものである。

(i)、(ii)ともサービス仕様におけるマン・マシンインタフェースの設計のまずさに起因している問題である。これは従来のサービスにおけるマン・マシンインタフェースの

設計は、技術者が思い込みや想像で設計を行ないユーザの意見をほとんど反映させていなかったためであり、具体的には次の3つの要因に分析される。

- ①マン・マシンインタフェース設計のための統一的な指針がない。
- ②マン・マシンインタフェースをユーザが事前に理解し意見を反映させる手段がない。
- ③マン・マシンインタフェースの良さを把握するための一般的評価法がない。

#### 5.5.1.3 マン・マシンインタフェース設計の課題

前節で分析した問題点の要因を解決し、操作性の高い通信サービス仕様のマン・マシンインタフェースを設計するためには、次の3点が求められる。

- (1) マン・マシンインタフェース設計のための統一的ガイドライン
- (2) 通信サービス仕様の操作性がユーザもしくはサービス開発者自身で評価可能な  
擬似環境
- (3) マン・マシンインタフェースの評価尺度とモニタ評価方法、モニタ評価結果の  
分析手法

このうち、(2)の擬似環境については、通信サービス仕様の動的な試験システムとして開発したサービスプロトタイプシステムを適用する。本システムを用いて、対象とする通信サービス仕様のマン・マシンインタフェースを実際のサービス提供とほぼ同一の条件で体験させて、操作性の評価を実現する。(3)については次節で、本モニタ評価方法に基づくモニタ評価結果については5.6で示す。また、(1)については、5.6の結果に基づき5.7で述べる。

### 5.5.2 サービスプロトタイプシステムを用いたマン・マシンインタフェースの評価法

#### 5.5.2.1 マン・マシンインタフェースの評価尺度

通信サービス仕様におけるマン・マシンインタフェースの良さを示す尺度としては、操作性に加えて、理解性、嗜好性の3つが重要であると考えられる。そこで、この3つの尺度を具体的に測定するためのデータを以下に示す。

### (1) 操作性

操作性については、以下の3つのデータを取得する。

- 入力操作時間
- 操作の学習効果（習熟度）
- 感想聴取

入力操作時間は、操作手順が簡明か／複雑かを示す指標として、一般的に用いられており、本論文でもこれを用いる。また、操作がわかりやすく、覚えやすいといった指標として、操作の学習効果（習熟度）も一般的であり適用する。本論文でのマン・マシンインタフェース仕様の評価における特徴として、被験者の発話や質問に対する応答を全て記録し解析する、プロトコル解析手法と呼ばれる手法を取り入れることにする。このため、操作性に関しても被験者の感想を求め、記録する。

### (2) 理解性

理解性については、以下の3つのデータを取得する。

- 誤り回数
- トラブル発生箇所の検出
- 課題の到達度

誤り回数は、マン・マシンインタフェース仕様の解かりにくさを示す指標として、一般的であり、本論文でも適用する。また、例えば誤りを起こさなくとも、何らかのトラブルが発生した箇所は、理解性の点で問題がある所であり、その箇所を把握する必要がある。仕様の評価をするために、被験者に与えた課題に対してどの程度到達したかも、仕様の理解度を図るために重要な指標と言える。

### (3) 嗜好性

マン・マシンインタフェースはかなり主観的な要素が大きく、客観的なデータとは別に、その仕様に対する嗜好を尋ねるために、

- 感想聴取

を行なう必要がある。

### 5.5.2.2 ユーザによるマン・マシンインタフェースのモニタ評価法

前述したように、情報処理分野の製品開発に適用されている、プロトコル解析手法を取り入れた、モニタ評価法の具体的な手順を準備とモニタ評価に分けて示す。

#### 5.5.2.2.1 モニタ評価の準備

##### (1) 課題の設定

- ① サービス仕様として設計した全ての機能（メニュー）毎に評価課題を設定する。
- ② 学習効果を見る場合、同一機能に対して複数回の課題設定を行なう。
- ③ メニュー選択のツリー構造や入力データ形式等比較評価すべきものは、それだけを取り出して部分的なサービス仕様の比較評価を実現可能にし、対応する課題を設定する。

##### (2) 被験者の選択

利用者として想定される年齢層を対象に、未経験者及び経験者を任意に選択する。

#### 5.5.2.2.2 モニタ評価方法

##### (1) モニタ評価環境及び採取データ

サービスプロトタイプシステム上でサービス仕様を動作させ、被験者には端末であるPB電話機により操作させる。評価データを以下のように採取する。

[撮影] VTRカメラ4台—被験者付近全景、電話機ダイヤル付近、  
被験者の顔、課題・マニュアル付近

(評価データ) 操作時間、操作時の顔の表情

[録音] 電話機から直接マイク入力

(評価データ) モニタ評価中の被験者の発声、トラブル時の助言／応答、  
課題終了時毎の質問応答

[操作入力記録] プロトタイプシステム内のコンピュータ上にファイル記録

(評価データ) 操作入力データ（PBコード）、メニュー選択ツリーの操作軌跡  
この模様を図 5.14 に示す。

##### (2) モニタ評価手順

- (i) モニタ評価目的を口頭で説明し、被験者の不安感、緊張感を取り除く。
- (ii) 操作マニュアル、課題を配付し、時間を限定せずに理解してもらう。



(iii)課題の順番に従い、試験を実施する。

(iv) 被験者の隣にアドバイザーを配置し、被験者が操作に行き詰まった状態（トラブル）に陥った時、助言を与える。

(v) 各課題終了後にアドバイザーが質問形式で操作性、利便性等の感想を聴取する。

### 5.5.2.3 モニタ評価結果の分析方法

上述のモニタ評価方法により得られた結果に対して、以下の手順で分析を行なう。

- (i) 各課題毎に操作時間が長い箇所、エラーの発生箇所を検出し、原因を分析する。
- (ii) トラブルの発生箇所を検出し、アドバイザーの助言／対話を基に原因を分析する。
- (iii) 比較評価項目については、操作時間、誤り発生率及び操作の感想等により、操作性を評価する。
- (iv) (i)～(iii)の結果を基に、以下の点を検討する。
  - ・マン・マシンインタフェース仕様改善の要／不要
  - ・仕様改善必要時の改善方法
  - ・比較仕様からの最適仕様の選択

### 5.5.2.4 モニタ評価結果のフィードバック方法

モニタ評価結果を上記方法により分析した結果、対象の通信サービス仕様を改善する必要があると判断された場合、具体的に通信サービス仕様どのようにフィードバックをさせて改善するかを以下に示す。以下では、電話系の通信サービス仕様のマン・マシンインタフェースの評価では、重要な要素となる3つを選んで説明する。

#### (1) メニュー分岐数

メニュー分岐数は、通信サービス仕様のツリー構造を規定するものであり、分岐数が変わると大きくツリー構造が変わることになる。これは、仕様設計においては情報シーケンス図のツリー構造に反映し、図 5.15 (a)に示すように、分岐記述要素において変更後の分岐条件と対応する分岐後の各情報シーケンス図を記述することになる。図は、2分岐から5分岐に変更する例を示している。分岐条件の変更は、図中に示すように、網内処理機能の記述における戻り値データの個数と値を変更することにより実現する。

#### (2) 操作コード

操作コードは、ユーザが電話機より入力するダイヤル数字の組み合わせであり、情報シーケンス上では、操作コード入力プリミティブの直後の、分岐記述要素における分岐条件に対応する。これは、(1)で示した方法で実現される。

#### (3) ガイダンス表現

音声ガイダンスは、入力操作を促すために用いられるのが主であるため、上記(2)の操作コードとペアで変更されることが一般的である。情報シーケンス図上では、ガイダンスを表現するプリミティブのパラメータとして、ガイダンスプリミティブ上で開かれるサブウィンドウ上で、図 5.15 (b)に示すような形式で記述される。

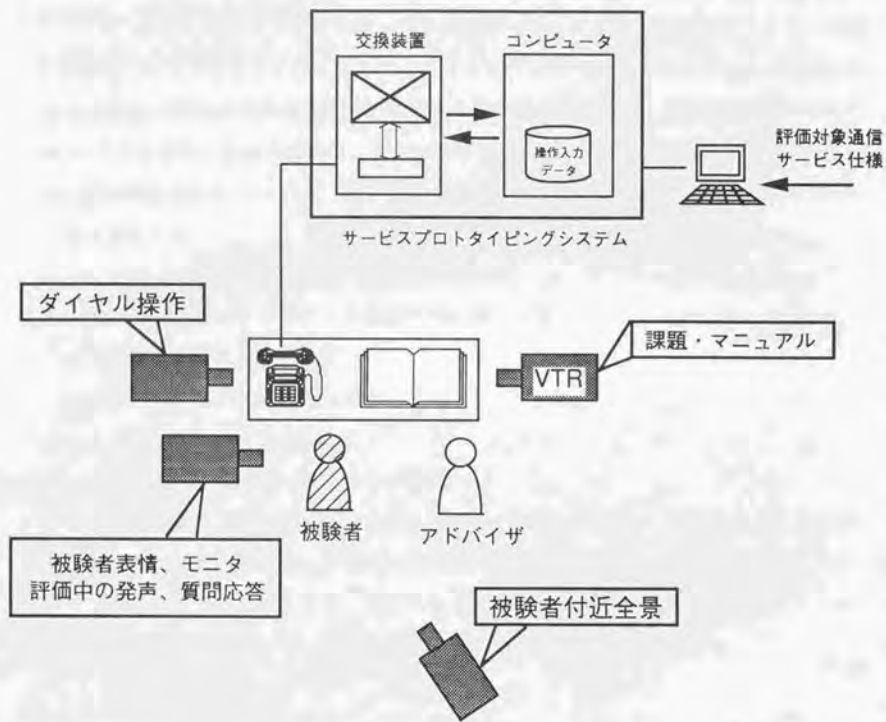
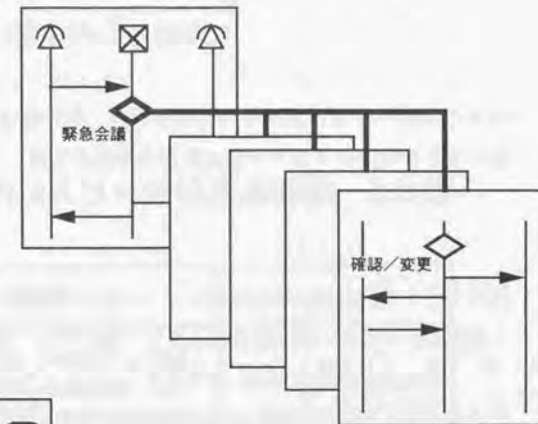


図 5.14 モニタ評価環境

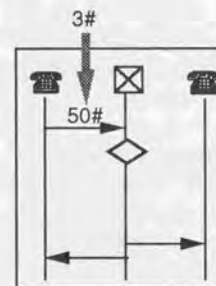
(1) メニュー分岐数

(例) 2分岐→5分岐

シーケンス図の  
木構造に反映



(2) 操作コード



信号シーケンス図のプリミ  
ティブ (メッセージ) に反映

(3) ガイダンス表現

ガイダンス信号  
プリミティブの  
パラメータに反映



図 5.15 仕様の評価結果のフィードバック方法



表 5.5 電話系通信サービスに内在する問題点

入出力方法	問題点
12個のPBダイヤルボタンによる入力	<ul style="list-style-type: none"> <li>・指示コード等が*、#と数字の組み合わせになり、多種類のコード作成には限界がある。</li> <li>・無意味な数字列や大量な各種識別番号番号を入力しなければならず、誤りやすい。</li> </ul>
可聴音信号、音声ガイダンスによる出力	<ul style="list-style-type: none"> <li>・ユーザによる信号音識別には限界がある。</li> <li>・音声ガイダンス情報は、揮発性であり内容を忘れやすい。</li> <li>・情報に冗長性があり、どれが重要かを表示できないため、重要な情報を聞き逃しやすい。</li> <li>・情報はシーケンシャルに出力されるため、必要な情報だけを最初に取り出すことができず、取り出すまで集中していなければならないため、ユーザの心理的負担が大きい。</li> </ul>

## 5.6 サービスプロトタイプリングシステムを用いた仕様評価法によるモニタ評価結果と考察

### 5.6.1 高度化電話会議サービス仕様のモニタ評価

#### (1) 被験者

事前に行なったアンケート結果に基づき、利用者の多い年齢層として、男性では30代、40代、女性は20代、30代から、現在提供中の電話会議サービスの未経験者及び経験者を任意に12名選択した。

#### (2) モニタ評価課題

①図 5.16に示す高度化電話会議サービスの仕様（ここでは操作性が特に問題となる会議予約部分のみ）について、緊急会議、予約会議、定例会議短縮登録、会議予約の確認・変更、会議予約の取消の5つのサービスメニューの各々について課題を設定した。

②既提供サービス、机上検討で問題となった以下の点を部分仕様を作成して課題を設定した。

- (i) 連続会議メンバ番号入力方法
- (ii) メニュー選択の最適分岐数
- (iii) 予約操作の学習効果

### 5.6.2 モニタ評価結果

#### (1) サービスメニューの課題

①予約確認メニューで終了のガイダンスが抜けていたために、被験者は平均55秒間も受話器を置かない状態が続いた。これは、終了ガイダンスを追加することにより、簡単に改善される問題である。

②「会議予約」と定例的な会議形式をパターンとして登録する「会議登録」の差が、被験者には理解されず、混同する操作が頻繁に（一つのメニューにつき、2回～4回／1被験者）発生した。会議予約を「会議申込み」、会議登録を「会議短縮登録」と表現を変えても両者の混同が発生することから、短い熟語で内容を表現するよりも長くなってもより表現としての類似度ができるだけ小さいものにする必要があると思われる。

③予約の確認後、予約の変更を「8#」の操作でできるようになっているが、「8#で変更ができます・・・」のガイダンスの後、予約の確認情報が流れ無音でユーザからの指示コードを待つため、被験者は「8#」の入力タイミングが理解できず平均約160秒間操作を待ってしまっていた。これは、ユーザの操作タイミングをガイダンスで明示することと、操作タイミングで入力促進音（例えば「ピッ」音）により操作を誘導することで解決であると思われる。

## (2) その他の課題

### ①連続会議メンバ番号入力方法

電話会議サービスでは、ホストを除く最大29名の会議参加メンバの電話番号を入力しなければならないが、既存のサービスではこの点で操作性の悪さを指摘するユーザの声が多かった。本モニタ評価での結果について、メンバ対応毎の入力操作時間を図5.17に示す。最も操作時間の要した時点のメンバ数は、被験者により6人～12人とばらついており、個人差が大きいが少なくとも安全側としては5名程度の入力で操作を一旦区切るのが、操作性の点からみて望ましいと言える。

### ②メニュー選択の最適分岐数

メニュー選択の分岐数を2つと5つの場合を電話会議の予約と会議形式の登録を比較評価した結果を図5.18に示す。この結果では、選択数による有意さはあまり見られず、5分岐程度までは操作性の点でそれほど問題はないと思われる。

### ③予約操作の学習効果

図5.19は、サービスメニュー毎の2回の平均操作時間を比較したものである。操作経験は1回だけでも大幅に減少しており、学習効果は期待できる。

## 5.6.3 考察

1回目の机上検討によるマン・マシンインタフェースの設計の場合と2回目の本設計方式によるマン・マシンインタフェースの設計の場合について、電話会議予約に要する操作時間を測定した。電話会議サービスの未経験者の操作時間の平均は、1回目で約14.9分、2回目で約6.7分であった。即ち、操作時間はおおむね半分に短縮されている。また、事前に行なった現在の電話会議サービスの利用者へのアンケート調査結果では、

習熟者による操作時間の平均値は約7.1分であった。これは、2回目の改善では未経験者でもわずか1回の経験で、習熟者の操作時間より約0.4分短い時間で実現できるようになったことを示しており、本評価法の適用により通信サービス仕様のマン・マシンインタフェースの操作性や理解性が向上したと言える。



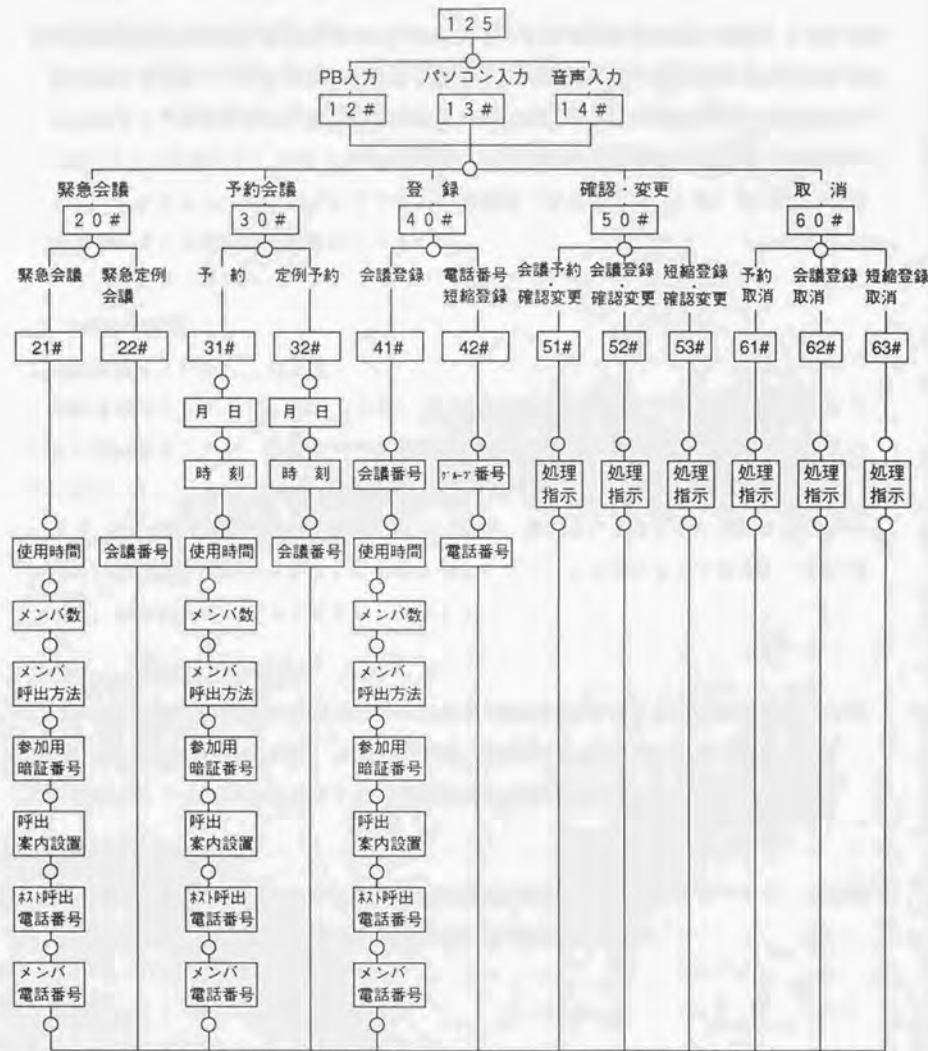


図 5.16 高度化電話会議サービスの仕様  
(予約部分)

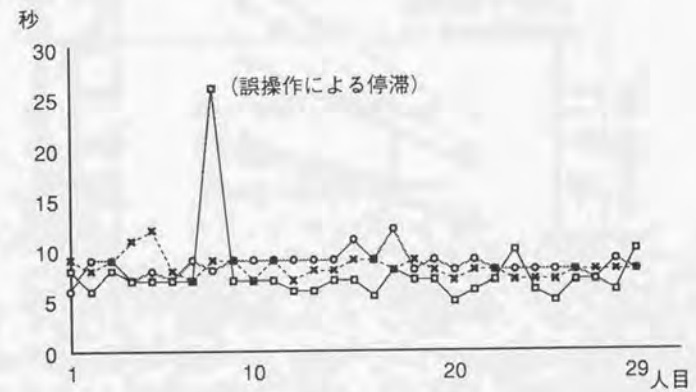


図 5.17 連続会議メンバー番号入力操作時間

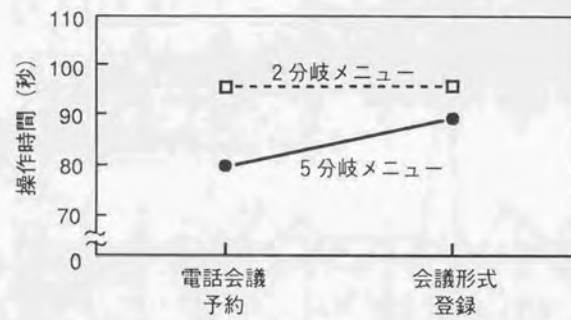


図 5.18 メニュー選択分岐数による操作時間の比較

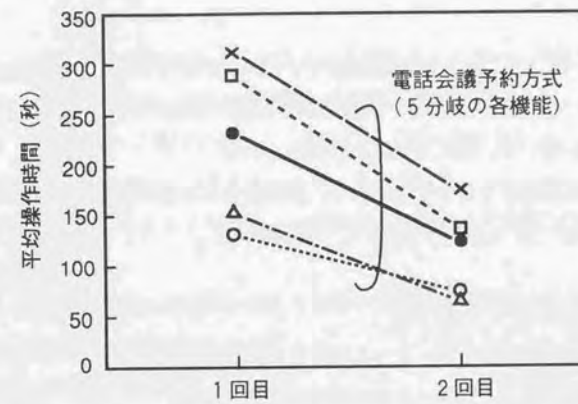


図 5.19 予約操作の学習効果



## 5.7 マン・マシンインタフェースの設計ガイドライン

既存電話会議サービスに対するアンケート調査結果、予備モニタ評価、及び高度化電話会議サービスに対するモニタ評価のそれぞれから得られた結果を基に、通信サービスのマン・マシンインタフェースを設計する際に有効と思われる汎用的事項についてガイドラインとしてまとめた。これを第3章で提案した仕様設計方式のStep 2におけるガイドラインとして使用する。

### 5.7.1 一般的指針

- (1) 対象とする新サービスを、ユーザが馴染みやすいメンタルモデルとして構成し、これに基づくマン・マシンインタフェースの設計が重要である。
- (2) 入力順序の設計しだいで、誤入力を防止できる。
- (3) 提供するサービス機能はできるだけ制限し、顧客層に合わせた機能を切りだしを図って、各種アプリケーションサービスとして構成する。

### 5.7.2 操作手順の設計ガイドライン

- (1) 分岐数が少ない方が望ましいが、5～8程度なら許容範囲である。
- (2) 単純な連続数値入力は、1分半～2分間連続が限界である。
- (3) システムとの対話による作業が一定のテンポでリズムカルに行なわれれば、操作に対するユーザの負荷は少ない。
- (4) 入力コードの意味づけは、一貫性が必要である（例えば、#と\*の使用法）。ただし、入力方法が大きく異なる状況ではこのルールから外れていても問題がない。
- (5) 複数項目に対する連続ガイダンス途中で、個別の項目に対する入力を求める操作は、ユーザに理解されにくい。
- (6) 操作の正誤の確認は即時に行ない、いくつかまとめてあとから行なう方法は避ける。
- (7) エラーメッセージを充実させる。
  - ① 誤操作の発生した場所を告知する。
  - ② 誤操作の原因を告知する。
  - ③ 誤操作の修正方法を告知する。
  - ④ 一定の時間間隔でエラー・メッセージを繰り返す。
- (8) 指示コード選択ガイダンスの冒頭で、概要説明をする。例) 「利用の形態を選択

してください。……」

- (9) 指示コード選択ガイダンスの冒頭で、いくつの選択項目が準備されているかを予め認知させる。例) 「以下の○個の項目から選んでください。」

### 5.7.3 音声ガイダンス表現の設計ガイドライン

- (1) 1文で1操作を指示する。

例) 「すぐにご利用になる時は20#を、予約の時は30#を、……」

↓

「すぐにご利用になる時は20#を押してください。予約の……」

- (2) 同音語との誤解が発生しやすい箇所は、大和言葉を使う。  
例) 「操作」→「操作する」、「指定」→「割り当てる」
- (3) ポーズやイントネーションにより、メリハリをつける。
- (4) 操作が復帰可能であることを示す。一番最初まで戻れる方法と一つ前に戻れる方法の両方が必要である。
- (5) 動作を指示する用語は別の言い換えを行わず最小限の数にする。
- (6) 操作の正誤の確認は即時に行なう。
- (7) 用語は以下の点を留意する。
  - ① 語彙数を限定する。
  - ② 日常用語を特別な意味で使用する場合は、制約条件を明確にする。
  - ③ システム記述言語、専門用語、業界用語は使わない。

## 5.8 むすび

通信サービス仕様の試験・評価技術のうち、次に以下の4つの技術課題について実現技術を明らかにした。

- (1) サービスプロトタイプシステムの構成法
- (2) マン・マシンインタフェースの評価尺度・分析手法
- (3) ユーザによるマン・マシンインタフェースのモニタ評価方法
- (4) マン・マシンインタフェース設計のガイドライン

マン・マシンインタフェースの評価尺度については、操作性、理解性、嗜好性の3つの観点から抽出した。また以下の3つの手順で行なう分析手法を示した。

- (i) 操作時間の長い箇所、エラーの発生箇所の原因分析
- (ii) トラブルの発生箇所の原因分析
- (iii) 操作時間、誤り発生率、操作の感想よりの比較評価

モニタ評価は、サービスプロトタイプシステム上で通信サービスの仕様を動作させ、被験者の動作を逐一VTRに記録すること、また、モニタ評価課題を設定し、学習効果を測定するものは複数回繰り返し、比較評価すべきものは部分的な仕様を抜き出した課題を実施する等の一連の方法を明らかにした。

モニタ評価法を高度化電話会議サービスの通信サービス仕様に適用した結果、操作性の改善が大幅に図れることを示した。また、この結果を基に、通信サービス仕様のマン・マシンインタフェースを設計するためのガイドラインを作成した。

## 第6章 インテリジェントネットワーク用通信サービスソフトウェア作成環境への適用

### 6.1 まえがき

第2章で提案した、通信サービスソフトウェア自動作成方式について、第3章から第5章までその実現技術について述べてきた。本章では、通信サービスソフトウェア自動作成方式を、現在、NTTで研究開発中の、インテリジェントネットワーク (IN: Intelligent Network) の通信サービスソフトウェア作成環境に適用することを検討する。INでは、通信サービスソフトウェア (INではサービスシナリオと呼ばれる) の開発がサービス要求に合わせて頻繁に行なわれるために、できるだけサービスシナリオの開発が簡単になる作成環境の構成が望まれていた。

本章では、NTTが既に提案しているINのアーキテクチャを前提として、サービスシナリオ作成環境への各構成技術について、それぞれ適用法を述べる。

### 6.2 INアーキテクチャ

#### 6.2.1 階層形アーキテクチャ

NTTが既に提案しているINアーキテクチャは、以下の3つの階層で構成される<sup>[40]-[42]</sup>。

##### (i) サービス管理階層

カスタマまたはサービス開発者が作成したサービスシナリオをもとにサービス制御機能が解釈・実行できる形式 (実行形式) に変換する機能と提供される各種サービス機能の使用権限や運用状況を管理する機能を持つ、サービス管理ノードで構成される。これは、第2章で提案した制御方式には含まれない、非呼対応の処理で、サービスシナリオの翻訳機能と既作成のサービスシナリオ実行形式と運用情報を管理する機能を実現する。

##### (ii) サービス制御階層

サービスシナリオに基づき、伝達機能を用いて多様な通信サービスの制御を実行する。

##### (iii) サービス動作階層

サービス制御機能の指示に従って種々の通信形態を実現する。



上記階層を構成するサービス管理ノード、サービス制御ノード、サービス動作ノードは、それぞれNSSP (Network Service Support Point)、NSP (Network Service control Point)、SAP (Service Action Point) と呼ばれる。これを図 6.1に示す。

### 6.2.2 サービス制御メカニズム

NTTが現在開発中のINでは、サービスシナリオを作成者のスキルに対応させ、カスタマとサービス開発者に分けて、以下の2つのレベルで構成し、それぞれSシナリオとNシナリオと呼ぶ。

#### Sシナリオ<sup>[43]</sup>

ユーザが使用する網機能や網リソースを契約の範囲内で定義する。また、時間帯、発信地域、ダイヤル番号等に応じて対応するサービス要素機能 (Nシナリオ記述) へ振り分ける、スクリーニング手順を記述する。

#### Nシナリオ

網制御機能要素 (伝達レイヤ制御インタフェースのメッセージに相当) を組み合わせ、ユーザに提供されるサービス要素機能であり、サービス開発者が記述する。

この2種のサービスシナリオの作成からサービス動作の実行までを実現するサービス制御のメカニズムを図 6.2に示す。Sシナリオは、ユーザ対応のスクリーニング等の定義データが中心であり、実質的なサービスシナリオはNシナリオに記述される。

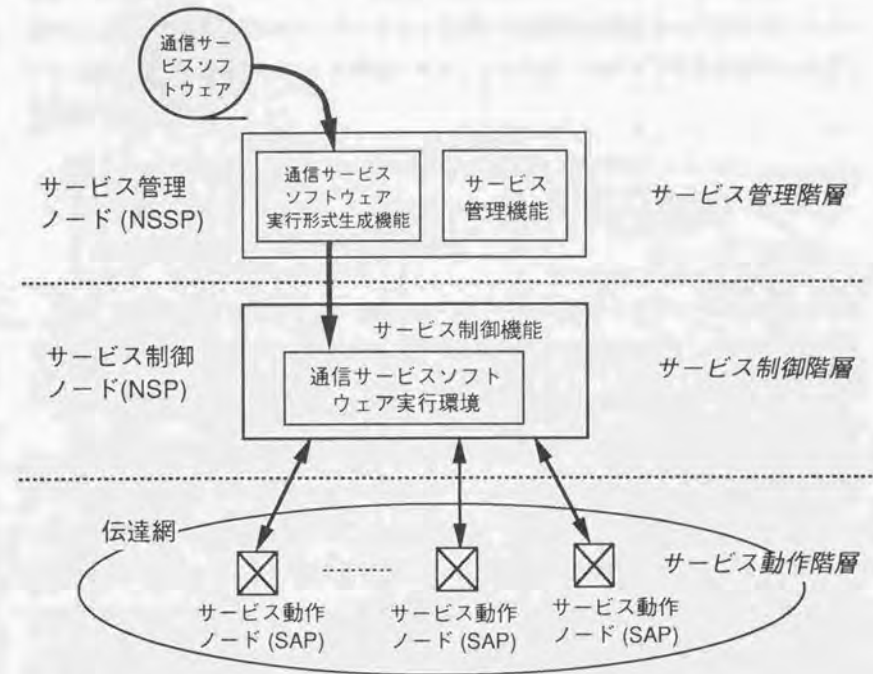


図 6.1 INアーキテクチャ

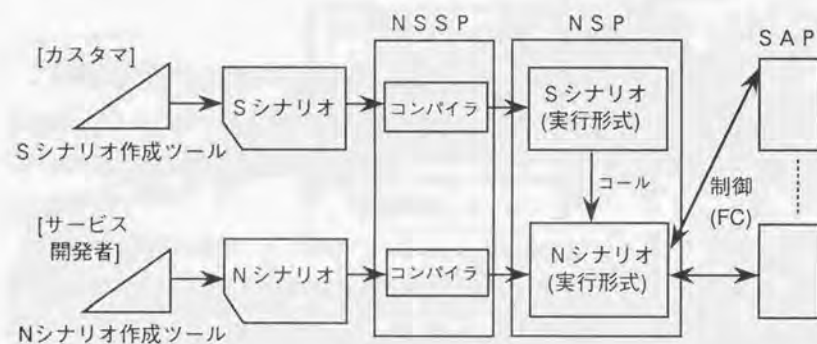


図 6.2 IN におけるサービス制御メカニズム

## 6.3 仕様設計/ソフトウェア生成技術のINへの適用

### 6.3.1 INサービスシナリオ作成環境に対する要求条件

6.2で示したように、INにおける実質的なサービスシナリオである、Nシナリオはサービス開発者が作成する。Nシナリオを作成するサービス開発者のスキルとしては特に限定せずに、交換サービスについての基礎的知識しか持っていないサービス開発者に対する非エキスパートから、INのアーキテクチャに関する詳細な知識を持った設計のエキスパートまで幅広く対象とすることを前提とする。このため、次のような要求条件を考慮する必要がある。

- (i) サービス開発者のスキルレベルに合わせた仕様入力インターフェース。
- (ii) 低いスキルレベルのサービス開発者が入力した仕様をより高いスキルレベルのサービス開発者が修正や変更が可能。

また、INのサービス制御では、従来のサービスでは実現できなかった課金条件の設定等のきめ細かい指定ができ、これをNシナリオに表現する必要がある、次の要求条件が求められる。

- (iii) きめ細かいサービス制御を指定するためのサービス記述範囲の拡大。
- (iv) IN提供サービスで対象とする仮想リソース制御の記述が可能。

### 6.3.2 仕様設計/ソフトウェア生成技術適用法の概要

要求条件 (i)、(ii) に対しては、それぞれのステップで仕様入力可能な階層形の仕様記述構成であり、仕様入力から1ステップ毎に変換するステップ・バイ・ステップ方式であることが望まれる。これは、既に第3章で示した3ステップの記述の各変換過程で、各スキルに合わせた複数の入力インターフェースを見せることにより実現する。すなわち、ステップ1に加えて、ステップ2、ステップ3でも仕様入力可能なように、信号シーケンスIED及び仮想リソース制御SDL図IEDを設けることにより、第3章で提案した技術がそのまま活用できる。

要求条件 (iii) に対しては、着信に対するスクリーニングや課金等の詳細な条件を入力可能な構成が必要であり、入力インターフェース構成法の記述要素の拡大と、計算機支援の知的入力/編集方式におけるIEDの構成、機能を拡大する必要がある。



要求条件 (iv) に対しては、ステップ3の記述要素の具体的部品名であるプリミティブを、IN用の仮想リソース制御メッセージに対応するプリミティブにすることに加えて、ステップ2からステップ3を生成するための新たな変換アルゴリズムが必要となる。

そこで、以下では詳細条件仕様入力法とステップ2-3変換アルゴリズムの実現方式について述べる。

### 6.3.3 詳細条件仕様入力法

INでは、網リソースの割当アルゴリズムをリソース対応に固定的ではあるが選択可能となること、また課金条件や通信情報属性をサービスシナリオで指定可能となることを実現する。以下、この2つに関する実現方式を述べる。

#### (1) 仮想リソース割当アルゴリズム指定方式

INにおける共通の仮想リソース（専用リソースでも複数選択可能なものも含む）は、以下の4種類である。

- ・専用中継線
- ・音声蓄積装置
- ・会議ブリッジ装置
- ・分配装置

これらのリソースを割当てるアルゴリズムは、以下の4種類が埋め込まれる。

- (i) 先頭方式：常に先頭から空きのものを選択
- (ii) ラウンドロビン方式：決められた順番に割当てる
- (iii) 比率方式：予め設定された比率に従い、選択する
- (iv) ダイレクト方式：直接仮想リソース id を指定

この場合のメニュー選択による指定方式の構成を図 6.3 に示す。IEDの網構成エディタのなかの仮想リソースを指示すると、これらの仮想リソース情報ウィンドウが開き、さらにサブウィンドウとして割当アルゴリズム選択メニューが開くこととする。

#### (2) 通信中パラメータ指定方式

課金条件や通信情報属性は通信中に必要とされるパラメータであり、通信中をしめす記述要素とリンクさせて指定パラメータを入力できるウィンドウを開く方式とする。ま

た、通信中においては仮想リソースの接続状態、情報の方向、伝達レイヤでの即時課金は必須情報であるため、通信中記述要素から開くメインのウィンドウを構成する。

課金条件では、以下の4項目を指定する。

- ・課金先
- ・課金方式
- ・登算時期
- ・算出方法

また、通信情報属性には提供サービスがISDNの場合を考慮して、以下の4項目を指定する。

- ・コネクションタイプ
- ・ベアラ能力
- ・低位レイヤ整合性
- ・高位レイヤ整合性

これらの指定を可能とするウィンドウ構成を図 6.4 に示す。

### 6.3.4 仕様変換アルゴリズム

#### (1) 仕様設計/ソフトウェア生成技術適用における問題点

第3章で示したステップ3記述で用いられる入出力命令は、INにおいてはサービス動作ノード (SAP) 内のリソースをきめ細かく制御するものである。Nシナリオにおける命令体系は、マクロな命令体系で行われている。これをCCITTのI.130の規定に合わせて、表 6.1 (a) に示したメッセージのレベルをEF (Elementary Function)、及び表 6.1 (b) に示したINのメッセージをFC (Functional Component) と呼ぶ。すなわち、INにおける通信サービス仕様のステップ3記述は、FCを入出力とする状態遷移で記述する。

ステップ2からステップ3への変換において、ステップ2記述の記述要素がEFの組み合わせによって一意に決定する場合 (図 6.5 (a)) には、ステップ2記述に含まれる個々の記述要素に着目して順次置換することにより実現できる。しかし、Nシナリオのステップ3記述では、図 6.5 (b) のように複数の記述要素を一つのFCによって実現したり、ステップ2記述の記述要素を実現するFCを一意に決定できない場合があるため、記述

要素を順次置換する第3章で示したアルゴリズムは適用できない。

## (2) 仕様変換アルゴリズム

Nシナリオのステップ3記述生成のために、ステップ2記述を通信パスを確立するための手順ととらえ、通信中状態に着目した変換アルゴリズムを導出する。通信中以外の信号プリミティブは、まとめてインタラクション部分と呼ぶ。

### 変換アルゴリズム

- <1> ステップ2記述から情報チャンネルによる通信中の状態（以下、通信中状態と呼ぶ）の抽出
- <2> 通信中状態の状態遷移における仮想リソース制御を実現するFCの導出
- <3> 信号シーケンスのインタラクション部分の一部付加
- <4> 終了処理手順の導出

上記アルゴリズムは、具体的には以下の手法で実現する。

- ① <1>はステップ2記述の記述要素のうち通信中状態を示すものを交換用データベースである信号定義データベース中にマーキングしておくことで実現できる。
- ② INにおける制御対象の仮想リソースはレグという単位で規定されており、通信中状態の遷移はレグの遷移の組み合わせで表現できるので、<2>は通信中状態を構成する個々のレグについて状態遷移をおこなうFCを選択し組み合わせることによって実現できる。
- ③ <3>は信号シーケンスと比較し、FCで表現されていないインタラクション部分を付加する。
- ④ <4>は、終了のための仮想リソース解放処理を実行するFCの制御パターンを選択し、付加する。

以上の点を考慮した、変換アルゴリズムの具体的なイメージを図6.6に示す。

## (a) 専用中継線

仮想リソース情報	
グループID <input type="text"/>	<input checked="" type="radio"/> 専用 <input type="checkbox"/> 一般
個別ID <input type="text"/>	<input type="checkbox"/> 専用 <input checked="" type="radio"/> 一般
leg0	<input type="checkbox"/> 専用 <input checked="" type="radio"/> 一般
<input type="button" value="割当方法..."/>	

割当方法
※専用指定時のみ有効
[アルゴリズム]
<input checked="" type="radio"/> 先頭 <input type="radio"/> ラウンドロビン <input type="radio"/> 比率 <input type="radio"/> ダイレクト
[捕捉指定]
<input checked="" type="radio"/> 即時 <input type="radio"/> 予約

## (b) 会議ブリッジ装置

仮想リソース情報	
グループID <input type="text"/>	<input checked="" type="radio"/> 専用 <input type="checkbox"/> 一般
個別ID <input type="text"/>	<input type="checkbox"/> 専用 <input checked="" type="radio"/> 一般
leg0 接続数 <input type="text"/>	<input type="checkbox"/> 専用 <input checked="" type="radio"/> 一般
<input type="button" value="割当方法..."/>	

割当方法
※専用指定時のみ有効
[アルゴリズム]
<input checked="" type="radio"/> 先頭 <input type="radio"/> ラウンドロビン <input type="radio"/> 比率 <input type="radio"/> ダイレクト
[捕捉指定]
<input checked="" type="radio"/> 即時 <input type="radio"/> 予約

図 6.3 仮想リソース割当アルゴリズム指定方式



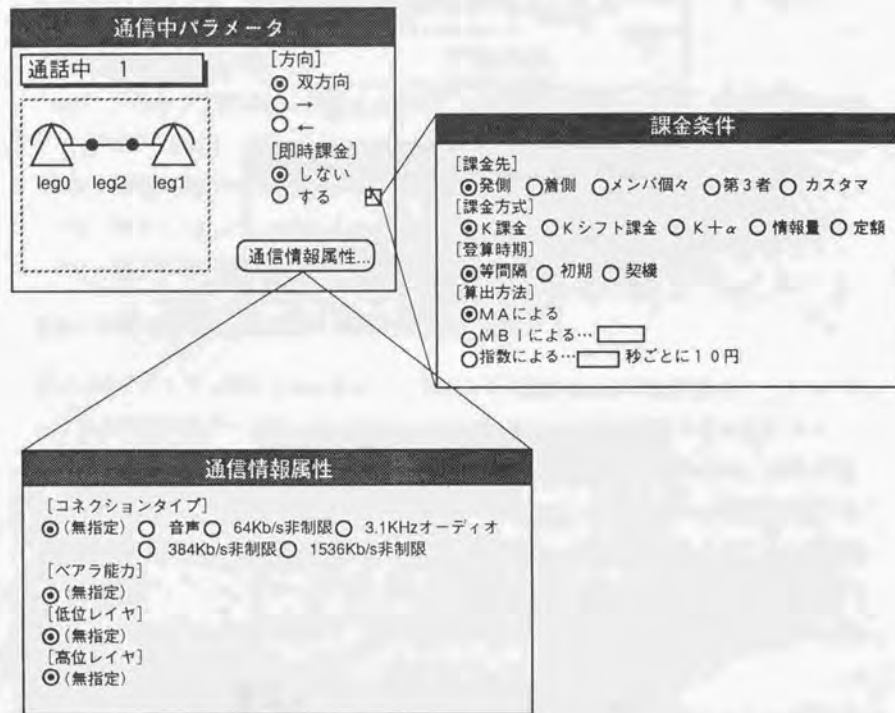


図 6.4 通信中パラメータ指定方式

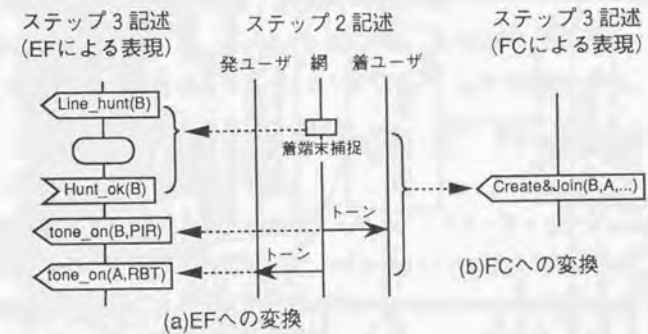


図 6.5 ステップ2記述からステップ3記述への変換例

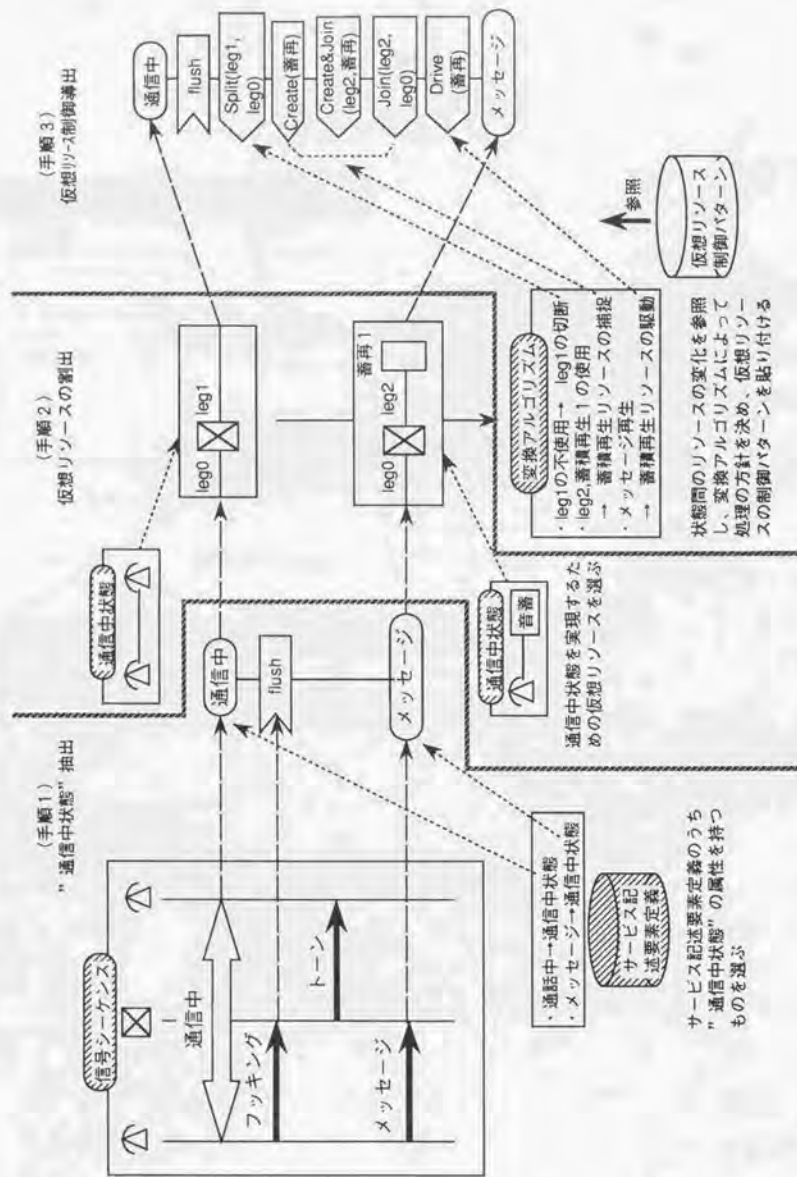


図 6.6 変換アルゴリズム

## 6.4 サービス仕様検証、試験・評価技術のINサービスシナリオ作成環境への適用

### 6.4.1 通信サービス仕様の検証技術適用法

通信サービス仕様に対して、静的な仕様検証については第4章で述べた、変換レベルに応じた階層的検証法が適用でき、サービス手順検証については第4章で提案した、段階的検証方式が適用可能である。

### 6.4.2 通信サービス仕様の試験・評価技術適用法

基本的に第5章で提案した、サービス仕様の試験・評価技術は全て変更なく適用可能である。ただし、INでは作成したサービスシナリオは、以下の特徴を持つ。

- (i) ユーザの定義したスクリーニングデータを扱うSシナリオとその他全てを扱うNシナリオに分離して記述される。
- (ii) 通信サービスソフトウェアが動作するサービスプロトタイピングシステムが動作する伝達レイヤ制御メッセージは、EFレベルのメッセージであるが、Nシナリオで記述する伝達レイヤ制御メッセージはマクロなFCレベルのメッセージである。

このため、SシナリオからNシナリオをコールするメカニズムと、FCレベルのメッセージをEFレベルのメッセージに展開する機能を追加することにより、IN用の通信サービス仕様のプロトタイピングを実現することができる。

## 6.5 INサービスシナリオ自動作成環境

上記技術を適用して、インテリジェントネットワーク用の通信サービスソフトウェアであるサービスシナリオを自動作成するための環境を構築した。作成環境は、図6.7に示すように、以下の4つのシステムにより構成されている。

- |                |   |
|----------------|---|
| サービス仕様記述システム   | : 3つのステップで構成され、シーケンス図またはSDL図で計算機支援により、通信サービス仕様を入力・編集する。 |
| サービスシナリオ生成システム | : 各記述のステップで入力された通信サービス                                  |



仕様を各変換知識データベースを参照しながら、プログラミング言語記述のサービスシナリオに自動変換する。

**サービス仕様検証システム** : 各記述ステップでの入力・編集後、各ステップに合わせた検証方式により仕様の自動検証を行ない、仕様誤りを検出する。

**サービスプロトタイピングシステム** : 通信サービス仕様のラピッドプロトタイピング（擬似的に動作させる）により、仕様を動的に試験・評価する。

本作成環境は、SUNワークステーションとサービスプロトタイピングシステムを構成するための交換装置とで構成され、OSはUNIX、ウィンドウシステムはX-Windowを用いている。現在、この環境上で、インテリジェントネットワーク用のサービスシナリオの開発を行なっている。

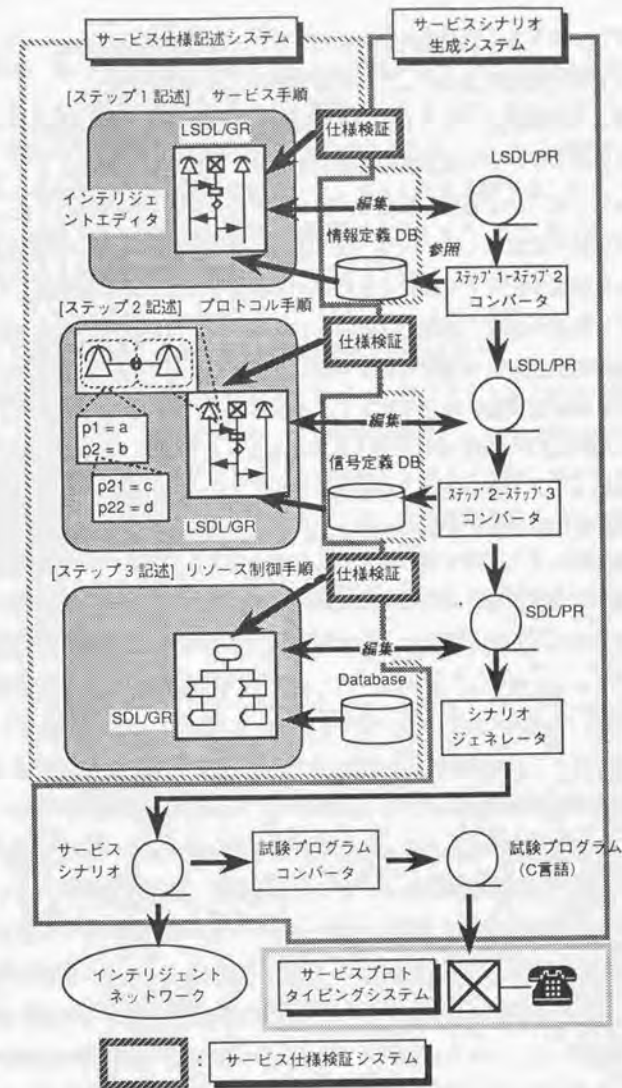


図 6.7 INサービスシナリオ自動作成環境

## 6.6 むすび

通信サービスソフトウェア自動作成方式を、NTTが提案し開発を開始しているINのサービスシナリオ作成環境に適用するための実現技術について述べた。

通信サービスの仕様設計/ソフトウェア生成技術の適用では、きめ細かいサービス制御を指定するためのサービス記述範囲の拡大のため、詳細条件入力法として、仮想リソース割当アルゴリズム指定方式と通信中パラメータ指定方式を明らかにした。また、IN仮想リソース制御の記述のため、FCレベルの伝達レイヤ制御メッセージで記述されたステップ3の状態遷移記述を生成するための仕様変換アルゴリズムを導出した。

通信サービス仕様の検証、試験・評価技術の適用では、検証技術については階層形仕様検証法、段階的検証方式の適用可能性を示した。また、サービス評価については、サービスシナリオの評価試験を行なうテストベッドである、サービスプロトタイプングシステムに、SシナリオからNシナリオをコールするメカニズムと、FCレベルのメッセージをEFレベルのメッセージに展開する機能を追加し、S、Nシナリオ形式のサービスシナリオでサービスプロトタイプングシステム上での評価試験を実現可能とした。

## 第7章 結言

本論文では、非エキスパートであるサービス開発者がサービス要求に合わせて自由にサービス仕様を設計し、ターゲットとなる通信システムのサービスプラットフォーム上で走行する、通信サービスソフトウェアを自動作成する方式についての研究結果を論じた。具体的には、通信サービスの仕様設計/ソフトウェア生成技術、通信サービス仕様の検証技術、及び通信サービス仕様の試験・評価技術を明らかにした。以下では、本論文で述べた研究内容と今後の展望・課題について述べる。

### (1) 第2章「通信サービスソフトウェア自動作成方式の提案」

従来の通信ソフトウェア開発の問題点を挙げ、通信ソフトウェア開発自動化への課題を明らかにした。また、通信ソフトウェア開発を効率化し自動作成の前提となる、通信システムにおけるサービスプラットフォームの構成とその上で走行し個々のサービスを実現する通信サービスソフトウェアについて示した。この前提の基に、サービス開発の自動化を推進し、非エキスパートによるソフトウェア開発を実現する、通信サービスソフトウェア自動作成方式を提案した。本方式の実現技術として、仕様設計/ソフトウェア生成技術、仕様検証技術、仕様試験・評価技術の3つを明らかにし、それぞれの主要技術課題を抽出した。

### (2) 第3章「通信サービスの仕様設計/ソフトウェア生成技術」

通信サービスの制御手順を非エキスパートなサービス開発者が作成できるような簡単なユーザインタフェースのみを見せ、目的の通信システム上で走行する通信サービスソフトウェアを生成するための、仕様設計/ソフトウェア生成技術について述べた。

仕様記述法については、メッセージシーケンス記述と網内処理機能記述の2つで構成される拡張形情報シーケンス記述法を提案した。本記述法におけるメッセージシーケンスを記述するため、新たに図形式と言語形式の両方を持つ、階層形サービス仕様記述言語MSDL (Message Sequence Description Language) を開発した。

仕様入力/編集法については、サービス仕様の入力を非エキスパートにも可能にするため、高機能なマン・マシンインタフェースを持ち、知識を用いた入力支援を実現する、IED (Intelligent Editor) を構成した。サービス仕様記述に関する知識を登録するデータベ



ースとして、ステップ1とステップ2に関連する情報定義データベース、ステップ2とステップ3に関連する信号定義データベースを設定した。ここに登録されている知識により、リソース状態による分岐の発生箇所、条件は、定義環境が自動的にサービス仕様設計者に提供可能となった。

仕様/プログラム自動変換法については、上記2つのデータベースに変換知識として、5種類を埋め込み、これに基づいてステップ1からステップ2、ステップ3と自動変換する仕組みを確立した。

試作システムによる評価結果により、通信サービスソフトウェアを直接汎用言語で記述する場合と比較して、本方式の仕様入力法では、約10%以下で記述できることを明らかにした。

### (3) 第4章「通信サービス仕様の検証技術」

通信サービス仕様検証技術として、サービスの仕様入力ステップに合わせて、階層的に検証を行なう階層的仕様検証法を提案し、そのうちの従来検討されていない、サービス手順の仕様検証方式について述べた。ここでのポイントは、シーケンス図で記述されるサービス仕様の大きな流れをマクロに捉えて検証する、マクロ化シーケンス検証と、マクロな検証対象の各要素を構成する個々の細かい記述要素間の正当性を検証する遷移手順検証の2段階で検証することにより、検証システムの実現性が高まること、検証法の理解性を向上すること、及び誤りの重要度の高いものを早く検出することにより仕様修正の効率化が図られることが挙げられる。

検証システムの試作により、本検証方式が高い検証能力を示し、実用的な時間内に検証を完了することが可能であることを明らかにした。

### (4) 第5章「通信サービス仕様の試験・評価技術」

自動作成された通信サービスソフトウェアに対して、実システム上で動作させる前に、要求仕様どおりの動作が実現されているかどうかを簡単にサービス動作を擬似体験して試験できる環境として開発した、ラピッドサービスプロトタイプシステムの構成法を示した。これを通信サービスソフトウェアの開発で使用することにより仕様レベルの誤りの検出、修正ができ、実システム上での試験を軽減できること、また仕様へのフィードバックを早期にできることにより、開発工数が削減されることを明らかにした。

また、本システムを用いて作成したサービスのマン・マシンインタフェース仕様を評価する手法を確立した。評価尺度については、操作性、理解性、嗜好性の3つの観点から抽出し、長い操作時間箇所の原因分析、トラブル発生箇所の原因分析等により行なう分析手法を確立した。お客様によるモニタ評価は、サービスプロトタイプシステム上で通信サービスの仕様を動作させ、被験者の動作を逐一VTRに記録すること、また、モニタ評価課題を設定し、学習効果を測定するものは複数回繰り返し、比較評価すべきものは部分的な仕様を抜き出した課題を実施する等の一連の方法を明らかにした。

モニタ評価法を高度化電話会議サービスの通信サービス仕様に応用した結果、操作性の改善が大幅に図れることを示した。また、この結果を基に、通信サービス仕様のマン・マシンインタフェースを設計するためのガイドラインを導出した。

### (5) 第6章「インテリジェントネットワーク用通信サービスソフトウェア作成環境への適用」

通信サービスソフトウェア自動作成方式を、NTTが提案し開発を開始しているINのサービスシナリオ作成環境に応用するための実現技術について述べた。

通信サービスの仕様設計/ソフトウェア生成技術の適用では、きめ細かいサービス制御を指定するためのサービス記述範囲の拡大のため、詳細条件入力法として、仮想リソース割当アルゴリズム指定方式と通信中パラメータ指定方式を明らかにした。また、IN仮想リソース制御の状態遷移記述を生成するための仕様変換アルゴリズムを導出した。

通信サービス仕様の検証、試験・評価技術の適用では、検証技術については階層形仕様検証法、段階的検証方式の適用可能性を示した。また、サービス評価については、サービスシナリオの評価試験を行なうテストベッドである、サービスプロトタイプシステムに、S、Nシナリオ形式のサービスシナリオが動作する擬似サービスプラットフォーム環境を構築し、評価試験を実現可能とした。

本論文では、通信サービスソフトウェア自動作成技術のうち、サービス仕様間の動的な競合検証方式やセキュリティについての検証法については今後の課題である。また、自然言語やアイコンにより、サービス要求を定義し要求仕様を自動作成するような、最上位レベルでの仕様定義法についても今後の課題である。

本論文で提案した通信サービスソフトウェア自動作成方式は、インテリジェントネットワークやコンピュータ制御形のPBXの導入による、カスタマイズドサービス提供の要求に伴い、今後ますます需要度を増す技術である。将来は、AI技術の進展をより積極的に取り入れ、サービス仕様設計を知的支援することにより、カスタマ自身が複雑な通信サービスソフトウェアを自動作成できるような環境構築への発展も期待される。本研究の成果が、今後のこれらの研究のための手掛かりとなり得れば幸いである。

## 謝 辞

本論文をまとめるにあたり、懇切なる御指導ならびに御助言をいただいた、東京大学工学部電子情報工学科 秋山稔教授、同 水沢純一助教授に深謝いたします。また、本論文に関して有益な御指摘と御助言、温かい励ましをいただいた、東京大学生産技術研究所 安田靖彦教授、学術情報センター 浅野正一郎教授、東京大学生産技術研究所 石塚満助教授、東京大学生産技術研究所 相田仁助教授に心から御礼申し上げます。

本研究は、NTT交換システム研究所における研究業務の一環として進めたものであり、本研究の機会を与えて下さったNTT幹部の皆様方に深く御礼を申し上げます。特に、LSI研究所 池田博昌所長（前交換システム研究所長）、塚田 啓一氏（前ヒューマンインタフェース研究所長）、通信網総合研究所 青木利晴所長、交換システム研究所 石川宏所長、交換システム研究所高機能処理研究部 鈴木滋彦部長には、研究に対する励ましと御助言をいただきました。

また、本研究を進めるにあたり、有益な討論、御協力を頂いた、人事部 岡田和比古部長（前交換システム研究所高機能処理研究部グループリーダー）、交換システム研究所 交機能処理研究部 伊藤弘グループリーダー、重松直樹グループリーダー、ソフトウェア研究所 市川晴久グループリーダー、交換システム研究所高機能処理研究部 石川和範主幹研究員、水野修主任研究員、近藤好次主任研究員、岡本光弘社員に深く感謝いたします。特に、本研究の当初より現在に至るまで討論や協力をいただいている、水野主任研究員、岡本社員には心より謝意を表します。

本研究のサービス評価実験及びINへの適用検討にあたって、多くの方々の協力、励ましをいただきました。特に、交換システム研究所 菱沼千明部長（前東京技術開発センタ所長）、和泉夏樹主任研究員、研究開発技術本部 吉田孝課長、サービス開発本部 有上俊男社員、ソフトウェア研究所 荒野高志研究主任、堀正弘社員に深く感謝いたします。

本研究の遂行、本論文の執筆は、以上の方々をはじめとする多くの関係の方々を支えられて実現することができました。ここに心より感謝の意を表します。



## 文献

### 全般

- [1] 石川 (宏)、石川 (秀) : "ネットワークの高度化とサービス展開", NTT技術ジャーナル, Vol.1, No.9, pp.4-8 (1989).
- [2] 藤野、花田 : "ソフトウェア生産技術", 電子通信学会 (1985).
- [3] 水野他 : "特集: 通信システムの形式記述技報の標準化", 情報処理, Vol.31, No.1 (1990).
- [4] 交換・通信ソフトウェア仕様記述言語の標準化と技術展望, 信学会専門講習会, pp.33-46 (1987).
- [5] 秋山、水沢、吉田、田中 : "インテリジェントネットワークとネットワークオペレーション", コロナ社 (1991).

### 第2章

- [6] 大久保 : "CTRON入門", オーム社 (1990).
- [7] 水沢、岡田 : "インテリジェントネットワーク", 電子情報通信学会誌, Vol.74, No.74, pp.1190-1197 (1991).
- [8] J. Gilmour and R. D. Gove: "Intelligent Network/2", IEEE Communication Magazine, Vol.26, No.12, pp.8-11 (1988).
- [9] CCITT Recommendation Q.1200 series (1992).
- [10] 丸山、久保田 : "オブジェクト指向による交換プログラムの構成法", 信学論, Vol.J74-B-I, No.10, pp.757-768 (1991).
- [11] 有澤 : "ソフトウェアプロトタイピング", 近代科学社, ソフトウェア工学ライブラリ16 (1986).
- [12] 市川 : "通信ソフトウェアを対象としたCASEの現状と動向", 情報処理, Vol.31, No.8, pp.1049-1056 (1990).

### 第3章

- [13] CCITT Recommendation Z.100 (1988).

- [14] CCITT Recommendation Z.120 (1992).

- [15] CCITT Recommendation I.130 (1988).

- [16] 水野、高橋、伊藤 : "新サービスプロトタイピングにおける仕様記述法の考察", 信学春期全大, 1871 (1987).
- [17] 水野、新津、伊藤 : "新電話サービスプロトタイピングにおける仕様記述法の検討", 信学技報, SE87, pp.7-12 (1987).
- [18] 水野、新津、伊藤 : "階層形プロトタイプ仕様記述言語 (LPDL) の検討", 信学情報・システム部門全大, 528 (1987).
- [19] 水野、新津 : "階層形サービス仕様記述法の一検討", 信学春期全大, B-334 (1989).
- [20] 水野、新津 : "サービス記述手順に基づく階層形サービスソフトウェア生成法の検討", 信学技報, SSE89-63 (1989).
- [21] 新津、水野 : "非エキスパートを指向したサービス仕様記述法", '90信学春期全大, SB5-1 (1990).
- [22] Y.Niitsu, O.Mizuno: "Interactive specification environment for communication service software" IEEE J-SAC, Vol.8, No.2, pp.181-188 (1990).
- [23] O.Mizuno, Y.Niitsu: "Layered service software generation based on service description procedures", ICC '90, 303.2 (1990).
- [24] 水野、新津 : "メッセージシーケンス図入力による通信サービス仕様設計方式", 信学論, Vol.J74-B-I, No.12 (1991).
- [25] 角田、若原 : "プロトコル仕様の時系列表現と状態遷移の相互変換", 信学情報・システム全大, 526 (1987).

### 第4章

- [26] P. Zafiropuro, C. West, H. Rudin, D.D. Cowan and D. Brand : "Towards analyzing and synthesizing protocols", IEEE Trans., COM-28, pp.651-660 (1980).
- [27] M. Itoh and H. Ichikawa : "Protocol verification algorithm using reduced reachability analysis", IECE Trans., Vol.E66, No.2, pp.88-93 (1983).
- [28] 岡本、新津 : "サービス手順段階的検証の検討", 秋季信学全大, B-417 (1990).
- [29] 岡本、新津 : "サービス手順の意味検証法の検討", 秋季信学全大, B-338 (1991).

- [30] 岡本、新津：“サービス手順検証法の検討”、信学技報、IN91-109 (1991).
- [31] M. Okamoto and Y. Niitsu：“A verification scheme for service specifications described by information sequence charts”、IEICE Trans., Vol.E75-B, No.10 (1992).
- [32] Y. Kakuda, Y. Yasuhara and M. Norigoe：“An acyclic expansion algorithm for fast protocol validation”、IEEE Trans. Soft. Eng., Vol.SE-14, No.8, pp.1059-1070 (1988).
- [33] J. Yamazaki, K. Miyazaki, T. Suzuki and H. Takano：“SVEX: Switching program verification expert system”、ICC'90, pp.336-340 (1990).

## 第5章

- [34] 新津、伊藤、近藤：“新サービス開発用プロトタイプツール (NEWSTER) の構想”、信学技報、SE85-116 (1985).
- [35] 新津、伊藤：“新電話サービス開発におけるプロトタイプ手法の一検討”、情報処理学会プロトタイプと要求定義シンポジウム、pp.97-102 (1986).
- [36] 新津、水野、伊藤：“新電話サービス開発用プロトタイプツールNEWSTERの試作”、信学技報、SE86-76 (1986).
- [37] 新津、水野、水沢：“通信サービス仕様検証方式”、信学論、Vol.J72-B-I, No.4, pp.331-342 (1989).
- [38] 新津、水野：“新サービス用ソフトウェア作成自動化技術の開発進む”、NTT技術ジャーナル、Vol.1, No.9 (1989).
- [39] 新津、水野：“交換サービスソフトウェア仕様作成検証システム” NTT R&D, Vol.39 No.7 pp1061-1072 (1990).

## 第6章

- [40] 岡田、重松、佐藤：“インテリジェントネットワークにおけるサービス定義・制御の階層的構造”、信学技報、SSE90-17, pp.25-30 (1990).
- [41] K. Okada, K. Sato and Y. Kondo：“Hierarchical software definition structure for Intelligent Network”、IEEE GLOBECOM'89, 27.6.1-5 (1989).
- [42] 江崎、大宮、重松：“インテリジェントネットワークにおける伝達網の仮想化とその制御法”、信学論、Vol.J74-B-I, No.11, pp.949-958 (1991).

- [43] N. Uchida and A. Miura：“Customer-defined service model and definition method for Intelligent Networks”、IEEE ICC'91, 30.4.1-5 (1991).
- [44] Y. Niitsu and O. Mizuno：“Unified service creation environment for service scenario driven communication nodes”、IEICE JC-CNSS'90, pp. 43-47 (1990).
- [45] 新津、水野、岡本：“INにおけるサービス自動作成環境構成法”、信学技報、SSE90-116 (1991).
- [46] 水野、新津：“INサービスシナリオ作成ツールにおける仕様記述・変換法の検討”、春期信学全大、B-557 (1991).
- [47] 水野、新津、岡本：“INサービスシナリオ作成環境における仕様記述変換システムの試作検討”、信学技報、SSE91-159 (1992).
- [48] O. Mizuno, Y. Niitsu and M. Okamoto：“Service specification and service logic program generation for Intelligent Networks”、ICCC'92, pp.430-440 (1992).
- [49] Y. Niitsu, O. Mizuno and M. Okamoto：“Computer-aided Stepwise service creation environment for Intelligent Network”、IEEE ICC'92, 317.4.1-5 (1992).
- [50] Y. Niitsu and O. Mizuno：“Computer-aided stepwise service creation for the Intelligent Network”、IEICE Trans., Vol.E75-B, No.10 (1992).



