Vol. 35 No. 5　　情　報　処　理　　May 1994

解　説

IPS

オブジェクト指向分析・設計

7. ディジタル移動通信システムへの適用†

梶原清彦††　浜田雅樹††

1. はじめに

オブジェクト指向技術はさまざまな分野で重要な役割を演じ始めている．本稿の対象である通信分野でも，管理情報の国際規格[1]などに利用されるようになってきた．また，ソフトウェア開発においても，プログラムの保守性を向上させる新しい技術として期待されている．しかし，オブジェクト指向技術をソフトウェア開発に利用するためには，パラダイムシフトにともなう多くの課題があると思われる．オブジェクト指向開発法は，このパラダイムシフトにともなうリスクの不透明性などにより，まだ広く普及していない．

本稿では，筆者らが提案したオブジェクト指向開発法 ROOD (Realtime Object-Oriented Design) をディジタル移動通信システムの開発に適用した事例を報告する．主な結果は以下のとおりである．

- オブジェクト指向開発法により比較的保守性の高いプログラムが開発できた
- オブジェクト指向開発法のためのオブジェクト管理などの経験蓄積と技法確立が課題である

以下，適用の概要として，対象システムと適用したオブジェクト指向開発法などについて述べた後，完成したプログラムや開発プロセスについて評価する．なお，オブジェクト指向開発法における基本用語は OMT 法[2]に準じている．

2. 適用の概要

2.1 適用の背景

（1） 適用の動機

適用対象システムは長期にわたる保守が予測されることから『保守が容易になるようにする』ためにオブジェクト指向開発法を適用することとした．

（2） 開発における各種制約

- 期間

要求の決定後から運用テストまで約3年という時間的制約があった．適用はその中の約1年にわたる分析工程，設計工程である．

- 開発言語

実装言語は大規模システム開発に適した言語である Ada[3] を採用した．この選択の主要な理由の一つは，オブジェクト指向開発法の適用に適しているからである☆．

- 実装環境

本適用対象のソフトウェアは，専用ハードウェア上で走行するため，汎用機の OS がもつ機能の一部（タイマや排他制御など）も開発対象であった．

- 開発者

開発者は類似システムの開発経験者から新人までの 20 人前後で構成されており，各開発者の設計あるいはプログラミング能力は比較的高かった．さらに開発者は，対象システムの要求を十分に理解していた（開発者が要求を記述したため）．

2.2 適用対象システム

（1） システムの構成

オブジェクト指向開発法を適用したシステムは，**図-1** に示すディジタル移動通信システムを構成する装置の一つである無線回線制御装置のソフトウェアである（以後，ターゲットシステムと記す）．

† Applying Object-Oriented Method to a Digital Cellulerphone System by Kiyohiko KAJIHARA and Masaki HAMADA (Software Research Laboratory, Software Laboratories, Nippon Telegraph and Telephone Corporation).

†† NTT ソフトウェア研究所　ソフトウェア基礎技術研究部

☆ Ada は，パッケージによるカプセル化の機能，抽象データ型，静的な多相機能，パラメータによって静的に特化できるモジュール機能などの特徴をもつ．したがって，動的な多相機能（プログラムでの動的で柔軟な制御）を必要としない場合や継承が不要である（再利用を重視しない）場合などでは，オブジェクトの表現が可能な言語である．

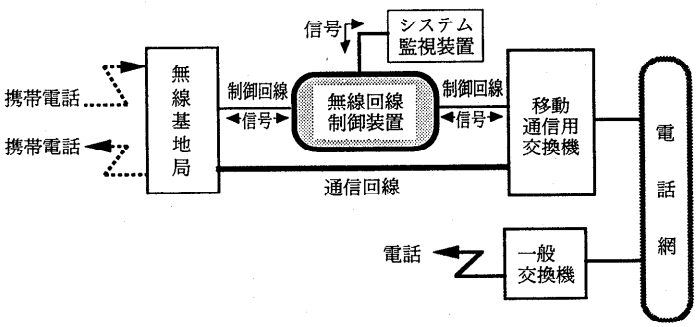


図-1 開発対象システム

（2） システムの機能

ターゲットシステムの機能は次の2種類のサービス群である*:

①通信サービス群

無線基地局または移動通信用交換機から信号を受け取り，通信回路の設定／開放／切替えサービスなどをおこなう.

②管理サービス群

リソースの状態報告／故障報告サービス，リソースや回線の試験サービスなどをおこなう.

通信サービス群と管理サービス群の比率は，サービス数で 1：3 程度，信号数で 1：1 程度である.

（3） リソースとプロトコル

通信サービス群および管理サービス群で扱うリソースには以下の種類がある.

- 論理的設備（無線周波数，タイマ，通信回線など）
- ハードウェアリソース(通信回線用パネル，システム運用用パネルなど)
- 履歴データ（リソースの利用率，故障履歴など）

また，両サービス群で用いられる信号のやり取り（プロトコル）を処理する部分の存在もターゲットシステムの重要な特徴である．以上のように，ターゲットシステムはリソースを管理する部分とプロトコルを処理する部分とに大きく分けることができる.

（4） オブジェクトへの対応付け

一つのサービスのプロトコル処理を一つのオブジェクトに*，それぞれのリソースを一つのオブジェクトにすることを方針とした.

2.3 ROOD 手法

筆者らは，ターゲットシステムの開発に先立ち，通信という分野で重視する特性をオブジェクト指向の枠組みでどのように考えるべきかを整理して，通信制御システムの開発での基本的な考え方と手順を規定したオブジェクト指向開発法 ROOD[4] を考案した**．独自の設計法を考案した理由は，適用開始時点では，まだ OMT 法や Shlaer/Mellor 法[5),6)]など実践的に利用できる手法が発表されていなかったこと，また通信分野に適したオブジェクト指向開発法もなかったためである．ROOD の詳細は文献 4) を参照してもらいたい.

（1） ROOD に基づく開発プロセス

今回の適用を例に ROOD に基づく開発プロセスを説明する（**図-2** 参照）.

0. 準備（参照モデルの作成）

図には示してないが，参照モデル（通信ソフトウェアのソフトウェアの枠組みを表したもの）をターゲットシステムの特徴をもとに作成しておく．本適用では，**2.2** で説明したプロトコル処理部分とリソース管理部分の分離をふまえ**図-3** に示す参照モデルを作成した.

I．分析

①初期オブジェクトの選択と②論理モデルの作成

通信サービス群と管理サービス群について（マルチビュー），実現すべき機能を論理的に記述したモデル（論理モデル）をそれぞれ別々に作成する．まず，事前に準備した参照モデルを参考にしてオブジェクトを選択する．以後，それぞれの論理モデルを，属性と操作をしだいに決める詳細化をはかりながら，論理モデルを洗練する.

③統合論理モデルの作成

両サービス群の論理モデルを統合し，サービス間の整合のとれた一つの論理モデルを作成する.

---

* これは通信の分野では一般的な分類である．通信サービス群は通信の利用者に通信回線（通信リソース）を提供するための機能であるのに対して，管理サービス群は通信回線を提供する側が，通信リソースを管理／運用するための手段として用いる機能である.

* ターゲットシステムの外部からの信号を操作とみなせば，それぞれのサービスは状態をもつオブジェクトとみなせる.

** 表記方法や詳細な手順は時間的制約から規定できなかった.

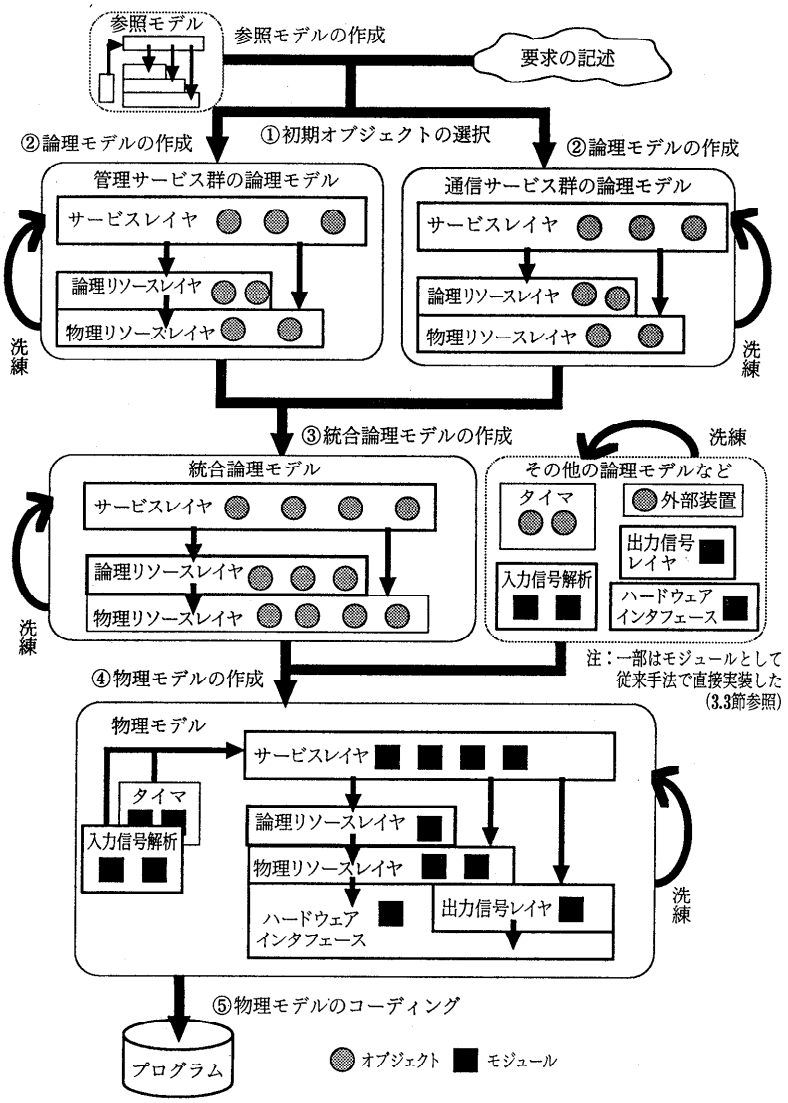


図-2　本適用における開発プロセス

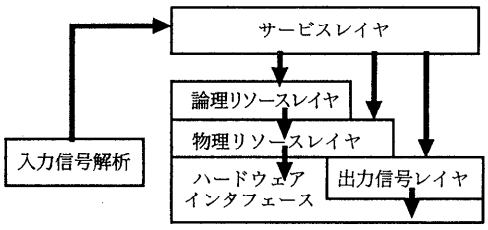


図-3　本適用における参照モデル

II. 設計

④物理モデルの作成

指定された実装環境上に論理モデルを実装する方法（物理モデル）を設計する．論理モデルはモジュール，プロセスなどでプログラムを構成する方法を記述したものである．物理モデルは対象システムで重視される特性，実装環境の特徴，実装言語の記述能力などを考慮して作成する．なおこの工程で，統合論理モデル以外で実装上必要になる部分の論理モデル（図-2 の「その他の論理モデルなど」）を作成し，統合論理モデルと組み合わせて物理モデルを設計する．

②～④の各作業では反復的にモデルを洗練する．

III. 実装

⑤物理モデルのコーディング

オブジェクトは，図-4 に示す方法によって Ada

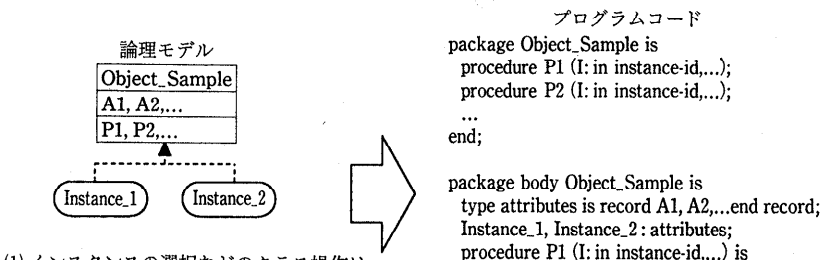


図-4 オブジェクトから Ada プログラムへの変換

のモジュール*として実装した.

（2） ROOD の特徴

• 参照モデル

参照モデルは，対象システムの特徴を踏まえてソフトウェアの構成と動作を単純化するためのもので，サブシステム構成，レイヤ構成，およびその間のインタラクションの枠組みを規定する.

• マルチビュー

ROOD では，異なる複数の利用目的（ビュー）から要求を分析し，のちに（要求間には関連性があるため）それらを統合することで，大規模システムにおける分析の複雑さを軽減する.

• 論理モデルと物理モデル，静的モデルと動的モデル

ROOD では，実現すべき機能とその実装方法を分けて設計する．これらの結果を記述したものを，それぞれ論理モデルと物理モデルと呼ぶ．また，論理モデルはオブジェクトがもつ構造と状態から設計される．それらを記述したものを，それぞれ静的モデルと動的モデルと呼ぶ.

2.4 教　育

（1） 教育方針

ほとんどの開発者は，オブジェクト指向技術の十分な知識をもたなかったため，開発者に対するオブジェクト指向技術の教育を以下の方法でおこなった．以下の教育は，オブジェクト指向技術および ROOD を熟知した教育担当者（著者）が本プロジェクトに参加しておこなった.

（a） 全開発者を対象とした，オブジェクト指向開発法の利用技法の教育（講義形式）

（b） オブジェクト管理者および開発者の中核となる技術リーダを対象とした，ROOD の適用法の説明（ディスカッション形式）

（c） 全開発者を対象とした，オブジェクト指向開発法の修得のための質疑応答（OJT 形式）

（2） 教育方法と期間

（a） 講義形式で開発が始まる 3 カ月前に合計三日間集中しておこなった．教材はオブジェクト指向の必要性や考え方，オブジェクト指向開発法を説明した OHP シート約 150 枚である.

（b） オブジェクト指向開発法を適切に進行する技法の習得のために，**2.3** で述べた各工程に入る前に数時間の議論をおこなった．その議論では，その時点の状況を踏まえながら，ROOD の適用方法の詳細やモデルの記述方法の詳細を決定した.

（c） 開発期間全体にわたり随時，開発者の質問に答える形式で教育をおこなった．多くの質問は分析や設計の結果の妥当性を確認するものであった.

（3） 結　果

（a）の説明会の結果は，オブジェクト指向開発法のメリットが理解され，動機付けができた程度であった（アンケートの結果から）．具体例が不足していたこと，参考にできる技術書がなかったため演習がなかったことなどの理由から，実際にオブジェクト指向で考えるレベルには達しなかった．また，（b）および（c）における教育では，開発にかかわる諸問題を解決することが最優先されたため，各開発者の理解がともなわない結果と

* 本記事では，Ada のパッケージという呼び名のかわりに，一般的な呼び名であるモジュールを用いた.

なってしまった．これは，開発者が彼らのそれまでのパラダイムである機能指向で考えた結果を優先させてしまう現象となって現れた．

以上の経験などからすると，新しいパラダイムであるオブジェクト指向への思考の切替えには，オブジェクトという概念への意識の集中と優秀な教育者による指導が不可欠である．したがって，初期の教育には OJT は不適切である．

3. 適用の結果とその評価

本章では，設計結果を簡単に紹介したのち，開発したプログラムとその開発プロセスについて，それまで用いていた機能分割法*の場合と比較し，評価する．なお，計測データの不足により定量的な評価はおこなっていない．また，ターゲットシステムは，それまでとは異なる開発言語での新規開発であるため，過去のプログラムの再利用はできなかった．さらに，大規模への最初の適用であるため，再利用と生産性については議論できない．

3.1 適用結果

表-1 に統合論理モデルと，それ以外の論理モデル（設計工程で分析したもの）におけるオブジェクトの種類と数，そしてプログラムにおけるモジュールの数を示す．

3.2 プログラムの定性的評価

本適用の目的である「保守性の向上」について評価する．また，信頼性と実行効率についても考察する．

（1） 保　守　性

以下の点から，プログラムの保守性は向上していると結論できる．

表-1 適用結果

統合論理モデル	
オブジェクトの種類	通信用リソース，ハードウェア構成部品サービスなど
オブジェクトの数	約 60 個
その他の論理モデル	
オブジェクトの種類	タイマ，インスタンス管理オブジェクト履歴オブジェクト，外部装置など
オブジェクトの数	約 300 個
プログラム	
モジュールの数	約 70 個
プロセスの数	5 個
モジュールのサイズ	300〜3000 lines 程度
手続きの数	5〜50 個程度／モジュール

☆ 機能をトップダウンに詳細化して，ソフトウェアを分析，設計する手法．

- プログラムの理解性向上

問題領域のリソースやプロトコルがオブジェクトとしてプログラムの構造に現れていること，およびプログラムの構造と動作の単純化により，プログラムの理解性が大幅に向上し，実行中に現れたバグの症状とプログラムとの対応付けが容易になった．

問題領域との対応による理解性向上は，すべてのオブジェクト指向開発法が，問題領域に存在するものや概念をモデル化する手法であるため，一般的に成り立つ．これに対してプログラムの構造と動作の単純化は，問題領域との対応だけから導かれるのではなく，（問題領域との対応を歪めることになるかもしれない）強制的な単純化を必要とすることが多い．Smalltalk の Model-View-controller モデルはその例といえる．

- 機能追加の容易化

現在までのところ，適用以後の新機能の追加は簡単におこなうことができた．これらの機能追加は，その新機能（サービス）がサービスレイヤのオブジェクトとして追加され，そのサービスにともなう新しいデータ（概念）などがリソースレイヤにオブジェクトとして追加され，また既存のオブジェクトとの調整をおこなっただけで実装できた．

これは大局的には，ソフトウェアの構成を図-3のようにサービスとリソースに分離したこと，局所的にはオブジェクトによるカプセル化によるコード変更がオブジェクトの外部に与える影響が小さかったことが理由である．ただし，このサービスとリソースの分離はオブジェクト指向から直接導かれるものではなく，対象システムの特徴を十分に把握した結果として得られたものである．

- 記述の集約による変更容易化

多相と総称をタイマオブジェクトの実装に利用することによって，抽象クラスに相当する記述の集約を実践した．この集約により，同一アルゴリズムで実装されなければならない部分の記述・修正ミスを減少させることができた．

通信制御という問題領域では，システムが扱う対象の種類や扱い方はパターン化される場合が多いので（本適用の場合では，リソース管理部分や信号送受信部分など），多くのパターンが抽象クラ

スによって記述できると思われる．この抽象クラス化により，変更がより容易になると思われる．

（2） 信 頼 性

今回の開発では信頼性に関するデータを収集しなかったので，本プログラムからは信頼性について妥当な評価はできない．しかし，開発中のバグ発生の特徴* から『上流工程の設計結果が，より洗練され，単純化されているほどプログラムのバグが少ない』という一般に言われていることと同じ結論が得られた．

一般に，オブジェクト指向開発法では，分析モデルや設計モデルの単純化が上流工程における洗練の目的である．したがって，オブジェクト指向開発法の利用により信頼性の向上が期待できる．

（3） 実 行 効 率

本適用では実行効率より保守性を重視したため，実行効率の改善はほとんどおこなわなかった**．実システムでの実行速度の実測はおこなっていないが，事前検討の結果[7] や実際に作成されたソースコードから判断すると，プログラムとしては機能分割法と比較して 10% 程度低下したと推測される．

この速度低下の最大の要因は，属性値へのアクセスが操作を経由しておこなわれることであると推定される（ソースコードからの判断）．これはオブジェクトによるカプセル化が原因であるため，速度低下する傾向を示すのはオブジェクト指向開発では比較的一般的な症状であると考えられる．ただし，モデルの単純化によって実行速度が向上する可能性もある．

3.3 開発プロセスの考察

プログラムの改善がなされても，開発の遅れやコストの増加をもたらすようでは妥当な開発方法とは言えない．本節では，本適用において開発の遅れなどをもたらした問題点とその原因を分析する．また，有益であったと判断できた事項についても考察する．

- 品質評価方法の欠如

本適用では，論理モデルや物理モデルの良し悪しが主観的・断片的にしか判断できなかった．

オブジェクト指向開発法による開発では，開発されるソフトウェアの構造と動作が従来の機能指向によるものとは異なる．したがって，従来の機能や制御などの特性にもとづいた品質の評価方法とは異なる評価方法が求められる．しかし，現在までのところ，広く認められた品質評価方法はない．分析モデルの品質評価方法は難しい問題である．けれども，それは最終的にはソフトウェアの基盤となることから，ソフトウェアの品質基準[8] である機能性，信頼性，使用性，効率性，保守性，移植性という観点と結び付けて評価すべきである．この分野は研究が少ないので（ここ3年間の OOPSLA での成果は文献9)だけである），当面はモデルの洗練経験を積んで，品質を向上させるためのノウハウを蓄積するしかないだろう．

- プロジェクト管理の不備

本適用では，品質以外にも進捗状況の把握，問題発生時の対応が，従来手法におけるプロジェクト管理のノウハウの応用では対処できなかった．そのため，十分なプロジェクト管理がおこなえなかった．これはパラダイムの変化が原因であると思われる．したがって，オブジェクト指向開発法のための進捗把握の方法，問題への対処方法の構築が必要である．しかし，大規模プロジェクトにおけるプロジェクト管理のデータの報告や研究が少ない現状では，事例報告[10] を参考にしながら，独自に経験を積んで，自前のプロジェクト管理技法を構築するしかないだろう．

- 参照モデルの利益

参照モデルは，**2.3** で述べた役割以外に，システム構成のイメージを事前に与えておくことによる開発者間の調整の容易化という効果を生んだ．さらに「実現できるか分からない」という不安をやわらげる効果もあった．

ROOD の参照モデルと同様のアプローチは，Shlaer/Mellor 法においてもドメイン・サブシステムとして提案されている．大規模なシステムでは，通信制御システムに限らず，分析工程からサブシステムへ分割することが必須であると思う．さらに，サブシステムやレイヤはプロジェクト管理における基本的な単位としても重要である．

- 部分的な非オブジェクト指向開発の利用

一部の開発者は従来手法での設計を用いたほうが早く高品質であると判断できたため，今回の開発では，入出力モジュールとハードウェアインタフェースモジュールを従来の機能分割法で設計・

☆ 論理モデルや物理モデルにおいて，うまく単純化できたと思われる部分ほどバグが少なかった．

☆☆ 主な改善として，インスタンスの静的メモリ割付けをおこなった．

実装した．これらは単なる変換ルーチンの集合であったため，オブジェクト指向開発法による実装の見込みと比較して，モジュールのインタフェースは異なるが，記述の基本的なまとまりやアルゴリズムの差は小さかった．したがって，保守性や信頼性の差はほとんどないと推測される．

本来なら，システム全体をオブジェクト指向設計法で開発するのが適切である．しかし，現状ではオブジェクト指向開発法を習得している開発者が多くないため，大規模なシステムでは既存の手法を用いた設計を部分的に認めざるをえない場合がある．その場合には，(お互いの影響を小さくするために) サブシステム間のインタフェースを調整した上で，入出力サブシステムのようなオブジェクト指向による表現以外にも妥当な表現方法が存在するサブシステムに既存の手法を適用するのが妥当であろう．

4. おわりに

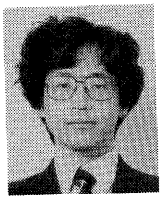
本適用は，最も重視していた保守性を向上することができ，比較的満足のいくプログラムが作成できた．また，オブジェクト指向開発法が通信制御プログラムの開発に適していることも確認できた．しかし，開発プロセスにおいては，開発の遅れや品質の予測ができないなどの問題が起こった．これはオブジェクト指向開発法におけるプロジェクト管理技術や品質管理技術の欠如が原因である．今後の現場での経験や適用データの蓄積，さらに実践的に利用できる技術の研究が必要である．

謝辞 本適用を決定し，実践していただいた移動体通信事業部・システム開発部（現在，NTT移動通信網(株)・研究開発部）の関係者各位に感謝いたします．

参考文献

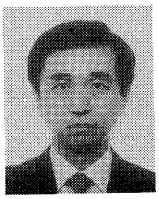
1) CCITT X.700 シリーズ ISO/IEC 10165 (1992).
2) Rumbaugh, J. et al. 著，羽生田栄一監訳：オブジェクト指向方法論 OMT，トッパン (1992).
3) JIS X 3009：プログラミング言語 Ada，JIS 規格書，情報処理—プログラミング言語編 (1993).
4) Yamazaki, S., Kajihara, K., Itoh, M. and Yasuhara, R.: Object-Oriented Design of Telecommunication Software, IEEE Software, Vol. 10, No. 1, pp. 81-87 (1993).
5) Shlaer, S. and Mellor, S. J. 著，本位田真一，山口亮訳：オブジェクト指向システム分析，啓学出版 (1990).
6) Shlaer, S. and Mellor, S. J. 著，本位田真一，伊藤 潔監訳：続・オブジェクト指向システム分析，啓学出版 (1992).
7) Hori, M., Yamazaki, S., Kajihara, K. and Aono, H.: Realtime OOD—Object-Oriented Design for Communication Control Systems, Proc. of TOOLS 3, pp. 199-206 (1990).
8) ISO/ITC 9126 Information Technology—Software product evaluation—(1991). JIS X 0129 (予定)
9) Shidamber, S. R. and Kemerer, C. F.: Towards a Metrics Suite for Object Oriented Design, Proc. of OOPSLA '91, pp. 197-211, ACM (1991).
10) Meyer, B. and Nerson, Jean-Marc ed.: Object-Oriented Applications, Prentice Hall (1993).

(平成5年11月9日受付)



梶原 清彦 (正会員)

1962 年生．1985 年山梨大学工学部計算機科学科卒業．1987 年同大学院修士課程修了．同年日本電信電話(株)ソフトウェア生産技術研究所に入社．汎用計算機 DIPS シリーズのソフトウェア開発環境の実用化を経て，現在，同社ソフトウェア研究所にてオブジェクト指向開発法の研究に従事．ACM，IEEE Software 各会員．



浜田 雅樹 (正会員)

1960 年生．1983 年慶応義塾大学工学部電気工学科卒業．1985 年同大学院修士課程修了．同年日本電信電話(株)入社．以来，ソフトウェア生産技術の研究・開発に従事．現在，NTT ソフトウェア研究所主任研究員．1990 年から 1993 年まで(株) ATR 通信システム研究所に出向．ソフトウェア再利用，オブジェクト指向技術に興味を持つ．1991 年情報処理学会奨励賞受賞．電子情報通信学会，IEEE 各会員．