事 例

上流工程における再利用を前提としたドメインモデルの使用法
―倉庫問題を例として―

A Domain Model for Reuse in Analysis Phase – Using a Warehouse Problem as an Example – by Shigeyuki KINOSHITA (Information Systems Technology Section, Information Systems Department, Kawasaki Steel Corporation).

木 下 茂 行[1]

1 川崎製鉄(株)情報システム部システム技術室

1. はじめに

オブジェクト指向開発においては，モデルの良否がシステム構築過程やでき上がったシステムの良否を左右する．したがって，よいモデルを再利用することは，開発の効率化とシステムの品質の両面から非常に有効であるといえる．また，最近ではビジネスオブジェクトやフレームワーク，パターンといった再利用技術が注目を浴びており[2)~4)]，上流工程での再利用の重要性の認識が高まっている．

我々は，日本IBMのユーザ団体である日本ガイドシェアのもと，その上位団体であるIUGC(International User Group Council)のオブジェクト指向プロジェクトにおいて，2年間に渡って再利用を中心に調査と研究を進めてきた．この成果物は，米国GUIDEのホームページ(http://www.guide.org/)に掲載されている．

この調査の結果，日本の各ユーザ企業においては，オブジェクト指向に対して，再利用性の向上や，生産性・柔軟性の向上といった期待が高いが，その本格的な適用については躊躇していること，再利用技術面では，負荷が大きくスキルが要求される上流工程については十分成熟しておらず，実際の開発局面ではまだ利用可能なものは少ないこと，などがわかった[1)]．

本稿では，再利用によるシステム開発を促進することを狙いとして，上流工程での再利用が容易な部品の特性とその作成の仕方を，例題に基づきできるだけ具体的に提示する．なお，これはビジネス・アプリケーションの，日常業務に近いもの(いわゆる基幹系システム)を対象に検討したものである．

2. 再利用の容易なモデルの要件

2.1 再利用の基本的な考え方

分析段階でのモデルの再利用の基本的な考え方を図-1に示す．分析段階でのモデルの再利用は，再利用の可能な汎用性をもった共通部分とアプリケーション固有の可変部分をいかにうまく分離し，共通部分を業務領域に対するドメインモデルとしていかに効果的に再利用するかにつきる．いい換えると，図-1で表現している考え方(アーキテクチャ)は，ある業務アプリケーションのモデルの共通部分と可変部分の分離の仕方，共通部分のカスタマイズの仕方を表現している．

a．共通モデルとしての再利用

ドメインモデルを自社の業務に適合させる形態での再利用である．この再利用では，ドメインモデルが対象とする業務を逸脱しない範囲での自社の固有業務に対するカスタマイズ，すなわちクラスの追加，サブクラス化による機能の追加，変更などを行う．

b．コンポーネントとしての再利用

ここでいう「コンポーネント」あるいは「コンポーネントとしての再利用」とは，単にソフトウェア部品としてのコンポーネントだけではなく，モデルとしての協調して振る舞うオブジェクト群(コンポジット・オブジェクト)を指している．上記a．に従って，自社の業務にカスタマイズしたドメインモデルを，ブラックボックス化したコンポーネントのモデルとして，可変部分(あるいは別の共通部分)から再利用する形態である．

各社固有の業務は共通部分の外の「可変部分」

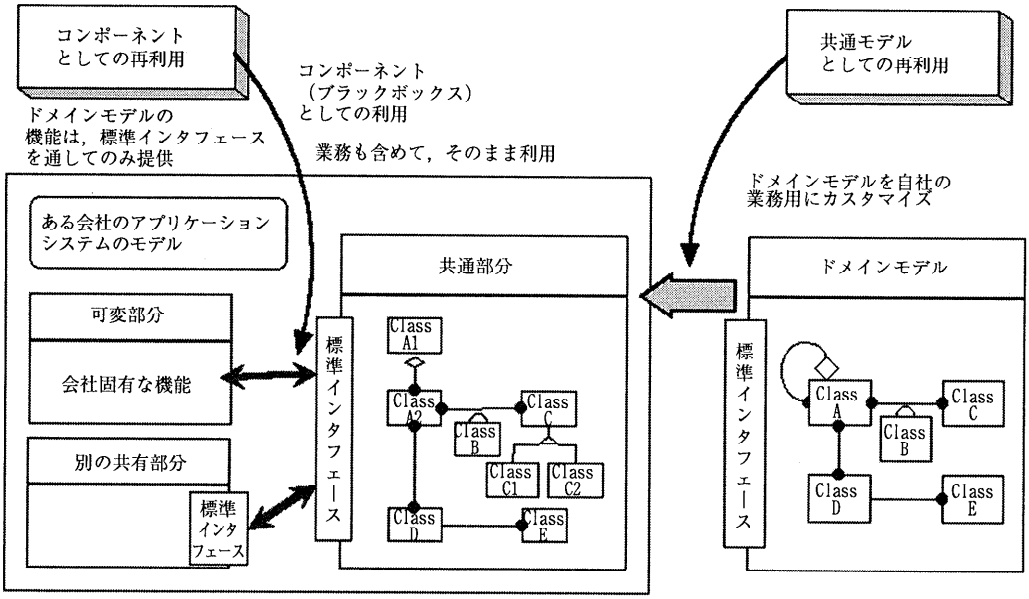


図-1 再利用の基本的な考え方（アーキテクチャ）

として実装し，共通部分が提供する標準のインタフェースを利用して，カスタマイズした共通部分の機能を利用する．

ドメインモデルは，その外の可変部分(あるいは別のドメインモデル)から，提供するインタフェースを通してのみアクセスを許すことで，内部を隠蔽する．このインタフェースは，ドメインモデルを操作する基本的なものをすべて備える必要がある．これにより，ドメインモデルの利用者(再利用する開発者)は，モデルの実装の詳細を知ることなく，分析に必要な共通部分のインタフェース(機能)だけを知ることでモデルの再利用が可能になる．また，インタフェースを明確にすることで，別の業務に対するのドメインモデルを組み合わせて再利用することも容易になる．

2.2 再利用しやすいモデルとは

我々は，事例に基づく再利用の検討を通して，再利用の容易性の要件を次のように整理した．

a．わかりやすいこと

上流工程でモデルを再利用するためには，そのモデルが何を表現したものであるかが理解しやすいことが要求される．優れたモデルであっても理解の難しいものであれば，それを利用するよりも最初から作った方がよいということも考えられる．

b．普遍性のある機能(業務)であること

BPR(Business Process Reengineering)などによる業務形態の変更や，業務形態の異なる他社でも変わらない基本的な業務の部分を中心に表現したモデルである方が，適合性が広い．

c．モデルの記述に一般性のあること(適合が容易であること)

モデルの静的な構造(静的モデル)が，一般性のある表現でなされており，実際の適用局面での固有事情に容易に適合できること，すなわちクラスの追加やサブクラス化による変更などが容易にできることが望ましい．

d．モデルが一意に解釈できること

モデルを再利用するためには，モデルのもつ意味やモデル作成者の意図が利用者側に正しく伝わる必要がある．また，1つのシステムを開発するモデルの利用者間でも，再利用するモデルに対する解釈が同じである必要がある．

e．追加・変更(差分プログラミング)が容易であること

実際の適用局面で，各社固有の(動的な)機能の追加や変更が容易なこと，すなわち追加・変更の個所の特定と変更の仕方の理解が容易なことが望ましい．

3. ドメインモデル作成のガイドライン

ここでは，上で述べたような要件をもったドメインモデルを作成するための考え方や留意点，ドメインモデルが備えるべき性質について述べる．

a．業務的に1つのまとまりのある単位

ドメインモデルは，1つの完結した業務機能だけを備えていることが望ましい．完結した業務単位にドメインモデルが作られることにより，モデルがわかりやすくなる．また，業務がモデル内で完結することにより，ドメインモデルの外部とのインタフェースが少なくなり，コンポーネントとしての再利用がしやすくなる．

ドメインモデルは単一の業務機能を果たす標準インタフェースが定義されたモデルである．実開発においては業務で要求される機能に応じたドメインモデルを組み合わせ，これらのドメインモデルをつなぐ「可変部分」を作成することで，アプリケーションを構築する．

カスタマイズ時にも，基本的にドメインモデルは1つの業務機能をもつという原則を崩さないことが重要である．カスタマイズ後もこの原則を維持することで，それらを組み合わせてのシステム構築を容易にするとともに，共通(ドメイン)モデル単位の機能の変更や業務の変更にあわせたドメインモデルの入れ替えが可能となり，システムの保守性が大幅に向上する．

b．物理的なもの(こと)を中心にした表現

前述の単機能は，できるだけ物理的なもの(こと)を中心にしたものの方が共通性が高いといえる．「物理的なもの(こと)」とは，物理的に存在するもの，およびそれらの間の関係，それらに対する直接的な操作，などである．これにより，上で述べた単一の業務機能の実現が容易になるとともに，共通なものと可変のものの切り分けの1つの基準を示すものとなる．

オブジェクト指向では，モデルは静的なモデル(オブジェクト関連図，クラス図，など)と動的なモデル(ユースケース，インタラクション図，など)で表現される．このうち，静的なモデルは業務に現れるオブジェクト間の物理的な関係を含む静的な関係を表現したものであり，同じ業務であれば企業間で共通性が高いといえる．一方，動的なモデルは，業務処理における各オブジェクトの振舞いを表現しており，各企業の個別性が入りやすい．このため，物理的なオブジェクトを中心に表現したクラスを中心に，静的な関係と物理的な操作をドメインモデル化するのがよい．

c．普遍性，一般性のある表現

カスタマイズを容易にするためには，できるだけ普遍性，一般性のあるモデル表現であることが望ましい．そのためには，前述の物理的なもの中心の表現においても，物理実態そのものをモデルで表現するよりも，それの実現している機能に基づいたものクラスとして表現するような抽象化が必要である．

d．クラスの責任の明確化

クラスとしてどのような責任(Responsibility)をもっているかを明確にしておくことがカスタマイズを容易にするとともに，カスタマイズ時に初期のモデルの考え方やよさを崩さないことになる．このためには，クラスに対する記述をきちんと残しておくとともに，実体がなくてもクラスの機能名(メソッド名とパラメータ)だけでも定義しておく方がよい．

e．平易なインタフェース

ドメインモデルのインタフェースは，その機能を表現したものである．ドメインモデルを再利用が容易なものにするためには，インタフェースがわかりやすい必要がある．また，分析段階でドメインモデルを再利用するためには，分析段階でのモデル化に適合したレベルでの，業務を表現したインタフェースである必要がある．ドメインモデル内部の処理に対しては，基本的にはインタフェースを提供せず，内部を隠蔽するのがよい．

さらに，外部とのインタフェースの考え方を利用者に明示して，ドメインモデルの利用時においてインタフェースの変更・追加を行うときにそれを守ってもらう必要がある．さもないと，内部の隠蔽が破られることになり，ドメインモデルの変更，バージョンアップなどの影響が外部に及ぶ．

f．利用形態の例示

ドメインモデルの理解を容易にし，再利用性を高めるためには，モデルの利用形態を例示するのがよい．利用形態の例示としては，ドメインモデル内部のもの(シナリオ，インタラクション図，など)と，外部からの利用を表現したもの(ユースケース，シナリオ)がある．前者は，ドメインモ

デルとしての再利用を行うときに，ドメインモデルの理解を助け，カスタマイズを容易にする．後者は，コンポーネントとしての再利用を行うときに，ドメインモデルをどのように使って業務を行うかを例示することで，利用を容易にする．

g．個別問題のビジネスルール依存部分の分離

機能の内容が個別の問題に依存する部分(ホットスポット[7),8)])をドメインモデルから分離することで，モデルの利用者は分析作業の焦点を早い時期に絞ることができ，モデルを再利用することによる利益を得ることができる．

h．構造のパターン化

一般的な構造のパターンをモデル内に取り入れることにより，モデルの構造を単純化し，再利用するために必須なモデルのわかりやすさを増すことができる．

i．実装依存構造の排除

オブジェクト指向の考え方では，分析から実装までが同じ形式でモデル化され，フェーズ間のギャップが少ないことが利点である．逆にいえば分析段階から実装時の実行効率やメモリ使用量などの実装についての考慮が入り込みやすいが，分析段階でのモデル化ではこれを排除すべきである．これらの要因は個別問題における実装環境や設計要因に大きく影響されるため，モデルが特定の問題や実装に依存したものとなり，再利用性を損なう．分析段階でのモデルは対象ドメインの構造を素直に反映したものであることが，モデルの理解可能性を高め，再利用をしやすくする．

j．モデル自身の再利用の過程での洗練

ドメインモデルをより再利用性の高いものにするためには，実問題にモデルを適用する過程での知見をモデルにフィードバックしていくことが必要である．モデル自体の粒度をより利用しやすいものにすること，ドメインモデルに必要なオブジェクトの選別，モデル内のオブジェクトの責任分担の調整などをモデルを利用した分析，設計の中で行うことで，ドメインモデルをより洗練されたものにすることができる．

4. 再利用しやすいドメインモデル導出の試み

我々は，倉庫管理業務を対象として，例題からのドメインモデル導出を試みた．ここでは，我々が検討した2つの例題の概略と，そこから導出した再利用性を考慮したドメインモデルの説明，ドメインモデルにおける再利用性への考慮点について述べる．

4.1 例題と構築したモデル

(1)例題の概略

対象とした例題は，文献5)の第13章「ケーススタディ：倉庫管理システム」(以下，Jacobsonの例題として参照)と，文献6)の第2章「Wally's Warehouse(A Warehouse Application)」(以下Coadの例題として参照)である．

a．Jacobsonの例題

ACME社という，複数の都市に倉庫をもち，客の品物の保管と倉庫間移動を請け負う倉庫会社のシステムである．客先からの荷物受入，倉庫間のトラック便による移動，客先の請求による荷物の送り出しを対象としている．客先の指示によらない倉庫間の品物移動があること，倉庫間には定期的なトラック便が想定されていることなどが特徴である．

b．Coadの例題

倉庫を1つだけもつ比較的小規模な卸売り業者のシステムである．手書き伝票で処理していた品物の入出庫や作業者の記憶に頼っていた保管場所管理が対象である．倉庫内部が保管棚の列，保管棚を区切った保管単位などの細かい保管場所単位に分割されている．

(2)ドメインモデル

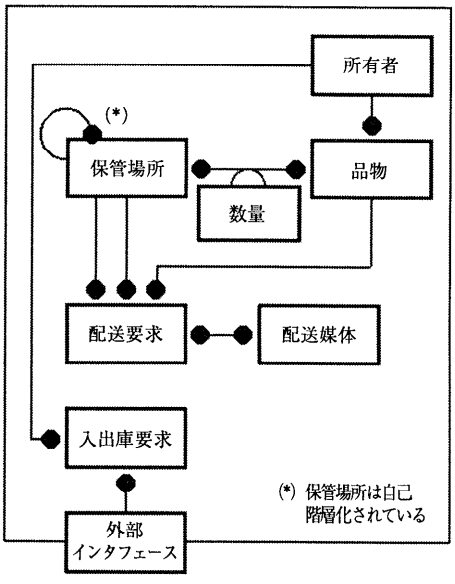
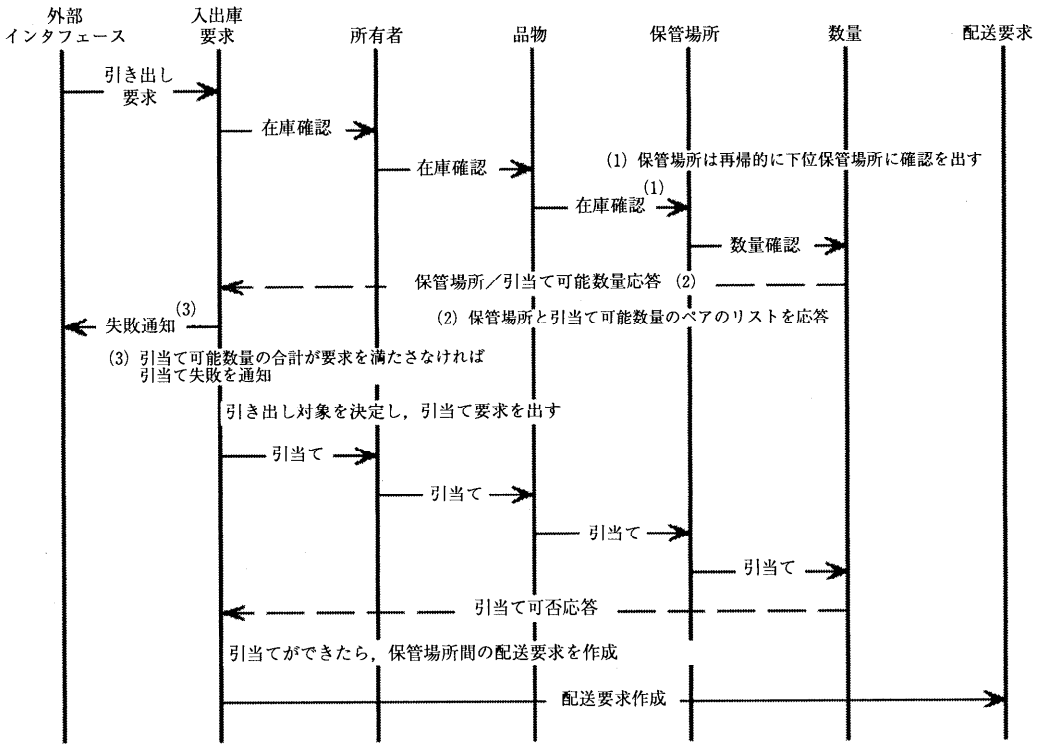


図-2 ドメインモデルのオブジェクト関連図

引き出し要求処理



預け入れ要求処理

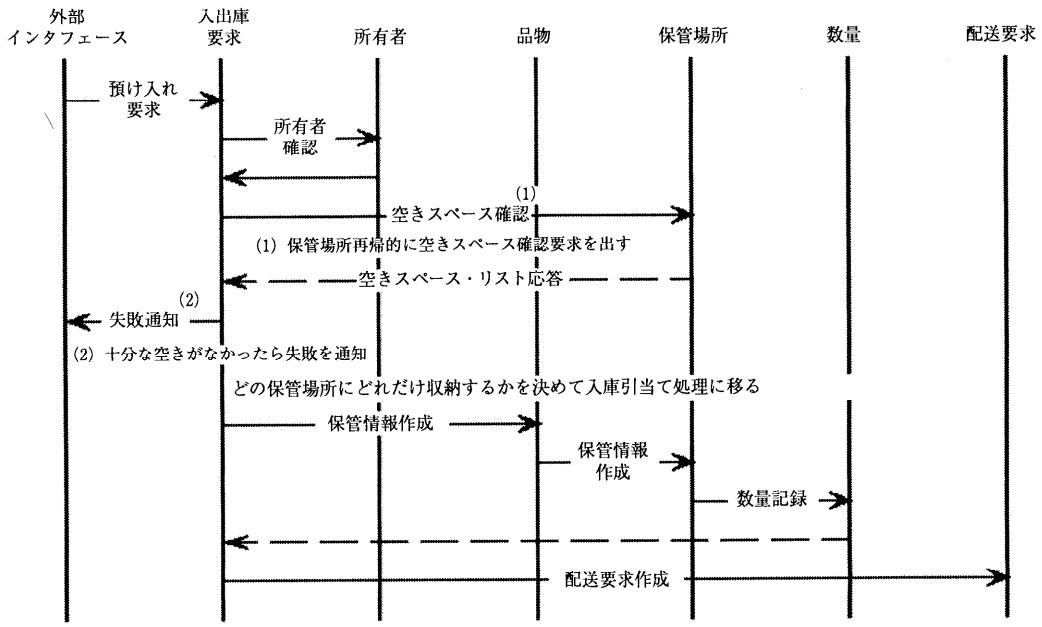


図-3 ドメインモデルのインタラクション図

図-2が今回作成したドメインモデルのオブジェクト関連図，図-3は，このモデルでの品物引き出しのインタラクション図である．モデルの導出にあたっては，まずJacobsonの例題のモデルを作成し，そのモデルをCoadの例題に適用することによって，当初のモデルの再利用性を検討するとともに，モデルの洗練を試みた．

4.2 ドメインモデル作成におけるガイドラインの適用

倉庫管理においては，貸し倉庫，製造業の倉庫，中間流通業の倉庫などが存在する．これらは，業態によって業務が異なるため，それぞれの全体のモデルは異なっている．しかし，「商品を預け入れ，保管し，引き出す」といった基本的な機能は共通であり，異なるのはその周辺にある受注管理，生産管理，輸送管理などである．ドメインモデルでは，「商品を預け入れ，保管し，引き出す」といった倉庫の基本的な業務機能を単位として作成した．

「置場」，「商品」などが実際の倉庫に物理的に存在するものであり，それらの間の関係(ある商品が置かれている置場，など)とともにドメインモデルとして表現した．また，商品の受入，払出し，置場間の移動(の指示)は，直接的な操作である．しかし，未入庫の商品に対する払出しの受付け(予約)は，「将来の在庫」という物理的に存在しないものに対する操作であり，ドメインモデルとして表現しない方がよい．これは，販売管理や生産管理など，倉庫固有の機能の外の機能であると考えられるからである．

「トラック」，「船」といったものは，物理的に存在するが，そのままクラスとして設定するのではなく，「輸送媒体」といった，トラックや船が実現している「輸送」という機能をクラスとして表現した．これにより，「輸送媒体」と表現されたクラスはトラックにも船にも(あるいはそれ以外の輸送媒体にも)適用が可能になる．

「商品を預け入れる」，「商品を払出す」，「在庫を確認する」，「置場の空き状況を確認する」などをドメインモデルのインタフェースとした．「倉庫内の置場の移動(指示)」や「入出庫の段取り」などは，倉庫内の業務であり，倉庫外からは知る必要がないものである．これらは，インタフェースを提供しないことで，モデルを平易にする．

「保管場所」に対する保管中の品物個数問合せのように，一般化がしやすく，個別問題へモデルを適用するときも，カスタマイズのかなりの部分が機械的な操作でできると考えられる部分と，「配送媒体」のスケジューリングなど個別問題の要件に大きく依存する部分がある．ドメインモデルには前者のみ取り込み，後者についてはインタフェースのみ定義することでドメインモデルからは分離した．

また「保管場所」については，“倉庫全体”→“個別倉庫”→“倉庫内の一番大きな区分”→…→“実際に「品物」を管理する区分”という再帰的構造をもつものとしてモデル化した．ここで再帰的というのは，たとえば「保管数量問合せ」というようなメソッドは，「保管場所」をカスタマイズした結果の上位階層のオブジェクトからは同じメソッドを直下の階層のオブジェクトに出して，戻ってきた値を戻すという，アルゴリズムの世界での再帰構造と同様のモデル化をしているためである．

5. おわりに

本稿では，上流工程での再利用の考え方と，例題に基づく再利用の容易なモデルの作成の仕方について述べた．

上流工程からのモデルの再利用は，現在のビジネスのスピードに対応して，「早く，安く」情報システムを構築する技術として，今後ますますその重要性を増すものと考える．本稿での提案が，その中で少しでも役に立てば幸いである．また本稿では割愛したが，再利用モデルを利用してのシステム構築とそのツールによるサポートについても，今後多いに議論され，技術が進展することを期待する．

参考文献

1) IUGCオブジェクト指向プロジェクトJGSグループ：オブジェクト指向開発における再利用技術の調査，日本GUIDE/SHARE(Nov. 1996)．
2) Sparks, S., Benner, K. and Faris, C.: Managing Object-Oriented Framework Reuse, IEEE Computer(Sep. 1996).(邦訳：フレームワークの再利用と管理法設計から現場配布まで，日経コンピュータ，1997.2.17)．
3) Hellenack, L. J.: Object-Oriented Business Patterns, OBJECT Magazine(Jan. 1997)．

4) Wooding, T.: Business Frameworks, OBJECT Magazine(Jan. 1997).
5) Jacobson, I., Christerson, M., Jonsson, P. and Overgarrd, G.: Object Oriented Software Engineering, Addison-Wesley(1992).(邦訳：オブジェクト指向ソフトウェア工学 OOSE，トッパン).
6) Coad, P.: OBJECT MODELS; Patterns, Strategies, and Applications, Prentice Hall (1995).
7) Pree, W.: Design Patterns for Object-Oriented Software Development, Addison-Wesley(1994).(邦訳：デザインパターンプログラミング，トッパン).
8) Pree, W.: Essential Framework Design Patterns, OBJECT Magazine(Mar. 1997).

IUGC オブジェクト指向プロジェクト JGS メンバ

増山陽平	東京ガス(株)情報システム部システム推進グループ
吉川浩史	東京ガス(株)情報システム部システム推進グループ
伊東義高	川鉄情報システム(株)情報技術センター技術開発部
松本好弘	住友金属情報システム(株)デザイン部第一システム技術室
常川郁代	(株)富士銀行システム開発部
三木亮二	新日鉄情報通信システム(株)技術部
古川陽一郎	日本アイ・ビー・エム(株)情報開発 CF ソリューション
矢崎朋夫	朝日新聞社(株)システム局システム部
田中一郎	日本アイ・ビー・エム(株)クロスインダストリー NCC 事業推進 NCC 技術センター
溝口直樹	日本アイ・ビー・エム(株)クロスインダストリー NCC 事業推進 NCC 技術センター

(平成9年9月18日受付)

木下 茂行(正会員)

1954年生．1979年京都大学大学院工学研究科情報工学専攻修士課程修了．同年川崎製鉄(株)入社．ソフトウェア開発技術，インターネット／イントラネット，データベース，エキスパートシステムなどの研究開発と社内適用に従事．現在，情報システム部システム技術室主査（課長）．人工知能学会，ACM 各会員．

e-mail:kinosita@system.kawasaki-steel.co.jp
URL: http://www.kawasaki-steel.co.jp/