Java 技術と適用技術の現状

Current State of Java Technology and Its Applied Technique

原　　潔

要　約　本技報に納められている論文は，Java を実システムに適用した体験をまとめたものである．特に業務システムへの適用を中心にしている．本論は，二つことを目的に書かれている．一つは，Java 技術が特に基幹系システムを担当している技術者にはまだ十分には広く知られているものにはなっていない現状を踏まえ，各論文の主題を理解しそこに現れる用語・概念の簡単な理解を得られるように Java の発展の歴史に触れている．これは本特集を読みやすくするためのものである．もう一つはこれからのコンピューティング環境での重要な技術となるコンポーネント開発技術について触れている．これはこれからのソフトウェア開発への提言である．

Abstract　This report describes two thing as its purpose. As it seems that Java technology is not still familiar to system engineers, especially to engineers for legacy enterprise system, the terms and concepts which appear in the subjects included in this report and the history of development of Java is mentioned. It helps to understand the subjects. Another is topics about component development technology in computing environment in the future. This is a proposal to Software Development in the future.

1. は じ め に

1990 年代の終わりにあたって振り返ってみると，この 10 年間はビジネス構造の大きな変換の渦の中にあった．それは情報産業のビジネスモデルを変化させるとともに新しい情報技術の誕生とその成長を育んできた．Unix 機や Windows 機の普及によるクライアント/サーバという新しいコンピューティング環境を基盤にした情報システムのダウンサイジングもしくはダウンコスティングがその端緒であった．続くインターネットの普及とグラフィカルな Web ブラウザの出現により大域な分散コンピューティング環境がグローバルなビジネス環境への変化に対応する情報技術として期待され分散コンピューティング技術が注目されてきている．このような 1990 年代の半ばに出現したのが，本特集で取り上げている Java である．

日本ユニシスでの Java への取り組みは 1995 年にさかのぼるが，Java が一応業務でも使えるように環境が揃ってきた 1997 年からの，主に業務システムへの Java 適用を実践的に行ってきた成果の一部を本論文集に集めた．各論文は Java 技術の単なる評論ではなく実践に裏付けられた知見の報告となっている．

本稿は，二つのことを目的に書かれている．一つは，Java 技術が特に基幹系システムを担当している技術者にはまだ十分には広く知られているものにはなっていない現状を踏まえ，各論文の主題を理解し，そこに現れる用語・概念の簡単な理解を得られるように Java の発展の歴史に触れている．これは本特集を読みやすくするためのものである．もう一つは，これからのコンピューティング環境での重要な技術となる

コンポーネント開発技術について触れている．これは，これからのソフトウェア開発への提言である．

2. Java 小史と Java 適用状況

2.1 新しいコンピューティング環境

Unix 機や Windows 機によるクライアント/サーバという新しいコンピューティング環境を基盤にした情報システムの構築は，ハードウェアの性能向上に加えその低価格化により初期の導入・開発費用を低減したが，運用・保守費用をいれるとかえって旧来のメインフレーム上でのシステムと変わらない，あるいはそれ以上の費用がかかることも判明してきた．その原因には，ネットワークに接続するコンピュータの種類と数が比べものにならないほど多くなりコンピューティング環境のインフラが複雑になってきたこと，並行して進んできた情報産業ビジネスモデルの水辺分散型への変化により，採用する基本ソフトウェアを複数のベンダから導入せざるを得なくなりソフトウェア構成が複雑になってきたこと，また新しいソフトウェアのバージョンアップが早くしかもしばしば非互換であり十分な品質を期待できないことなどが上げられる．そしてこのような基盤の上で情報システムを構築する技術の獲得が追いついていないことが上げられる．

メインフレーム中心時代のシステムは均質なものであったのに対し，クライアント/サーバのオープン時代のシステムは非均質なものになってきたのである．メインフレームでは，コンピュータを選定した段階で，使用するソフトウェアのセットとシステムを構築するためのプラットフォーム・アーキテクチャは決まっていた．しかし，オープンの時代では，ソフトウェアの選定とプラットフォーム・アーキテクチャの決定から行わなくてはならない．このことがシステムの構築と運用・保守を困難なもの，あるいは費用を高いものにしている．そのような非均質なクライアント/サーバシステムでの各システムの相互接続性，相互運用性を保証する技術として OMG（Object Management Group）が業界標準として定めてきたのが CORBA（Common Object request Broker Architecture）である．実装とインタフェースを分離するオブジェクト指向の考えのもと IDL（Interface Definition Language）を規定し，TCP/IP の上にオブジェクトレベルの通信を可能にするプロトコル IOP（Inter-ORB Protocol）を提供し，分散オブジェクト技術のもとで複雑なネットワークの物理構成からアプリケーション開発を独立にすることを可能にした．この CORBA はどちらかというと広域な大規模ネットワークシステムでその効果を発揮するものとなっており，基幹業務がなかなかダウンサイジングできない状況においてその適用はあまり進んで行かなかった．このような状況で出現したのが Java である．

2.2 Java 小史

Java が初めて公開されたのは，1995 年の SUN 開発者会議においてであった．このときの主題は，アプレットとアニメーティング Web ページであった．このころはインターネットが普及し始め，特に Mosaic に発端するグラフィカルなインタフェース（GUI）でインターネットをアクセスする Web 技術の出現が新しいビジネスを予感させ騒がれていた時期である．これまでは静的な情報しか扱えなかったのに対し，

Java は Web ページに読み込まれて実行されるプログラムを可能にした．この時デモされたのは株価の変動をリアルタイムに Web ページ上に表現するというものであった．今日のインターネットビジネスの基本を示している．Web ページでプログラムを実行できるということは Web の表現力を格段に高いものにした．加えてアプレットという実行時にサーバからクライアントにダウンロードされるという実行形態は，多数のクライアントにプログラムを配布するという問題に対し，新しい解決を提供した．

翌年，JavaOne と改称した 1996 年の Java 開発者会議での主題は，Java 仮想機械（JVM）と「write once and run anywhere」を実現する汎用言語としての Java であった．JVM により Java のプログラムはどこでも稼働可能となり，プラットフォームによらず一度作成したプログラムがどこでも実行できるということは，開発・保守の生産性を上げるものとして期待された．この年に Java でアプリケーションを作成する環境が整って来て，実システムを目指した適用が増え始めた．当時日本ユニシスで開発に入っていた CORBA プロダクトである SYSTEM ν［nju：］のクライアントプログラム作成の言語としても，いち早く Java を対象とした．当時はまだクライアントプログラムに対する関心が高い時期であったが，「write once and run anywhere」という特性は，Java を単なるクライアントプログラムの開発言語というよりむしろサーバプログラムに採用したら効果があるのではないかと期待された．寿命が長いのはビジネスロジックを実装したサーバプログラムであるからである．そのため日本ユニシスでは，Java の業務システムの適用に向けてその可能性と方針を追求するために 1997 年に Java プロジェクトを社内に組織し実証を進めた．

1997 年の JavaOne 会議での主題は，JavaBeans と Java コンポーネントモデルであった．この年 Java からデータベースを扱うための JDBC 仕様も確定しベンダからプロダクトが提供され始め，業務システムを作成するのに必須なトランザクション機能，データの永続機能が使えるようになった．

1998 年の会議では，Enterprise JavaBeans（EJB）が発表された．本格的に Java で企業システムを開発する方向がこれで見えてきた．この年日本ユニシスでは，前年の Java プロジェクトでの実証評価の結果を踏まえ社内の基幹システムの Java による本格的な開発に入った．このシステムは業務要件から，高い可用性と高効率の実現，トランザクション処理の実現が必要であり，トランザクション機能をもつ CORBA 製品である SYSTEM ν［nju：］と Java の組み合わせによる先進的な開発となっている．本システムは 1999 年 5 月より本番運用に入っている．

このシステム開発により Java によるプラットフォーム独立性と CORBA による分散透過性によりスケーラブルなシステムが構築できることが実証できている．Java のもつプラットフォーム独立性は，開発環境と実行環境を自由に行き来できるということに威力を発揮し開発生産性を上げている．また，Java 言語の特性からプログラミングから結合テストまでの生産性は十分に高いことが体験できた．特にメモリ管理からプログラマを解放していることは効果的である．

この社内システムの開発を報告しているのが，「勤務表システムの業務要件とシステム要件」，「勤務表システムのアーキテクチャ」，「勤務表システムにおける技術課題」

の三つの論文である．

この事例は Java と CORBA によるシステム開発であるが，Java と CORBA という新しいオブジェクト技術を使ったシステム開発を行う上での技術について一般的に論究しているのが論文「Java と CORBA によるシステム開発」である．また，クライアント/サーバシステムを Java で開発する上での適用技術について論究しているのが論文「クライアントサーバシステムにおける Java の適用」である．

先述した理由により我々の体験は Java でサーバプログラムを開発することに注力しているが，その中にあってクライアントについて論究しているのが論文「Java によるドキュメント作成ツール」である．

3. コンポーネント開発

3.1 J 2 EE

1999 年の JavaOne 会議での主題は，Java 2 Platform Enterprise Edition（J 2 EE）と JavaServer Pages，XML であった．

JavaBeans は，クライアントにダウンロードして Web ブラウザ上で実行できるコンポーネントである．それに対し，EJB はサーバ側で機能するコンポーネントのアーキテクチャであり，J 2 EE はそのサーバ・コンポーネントを本格的に使用できる開発実行環境である．そのコアになる技術は，1998 年 3 月に EJB 仕様 1.0 として発表されていたが，1999 年 6 月に仕様 1.1 が提供され，新たに永続性を保証する Entity-Beans が必須条件として定義された（図 1）．これにより本格的なエンタプライズ・コンピューティングに対応できるコンポーネントの実行環境の実現が可能となり，オブジェクト指向のプログラミングモデルを既存の企業システムの中に浸透させる準備が整ったといえる．プラットフォームに依存しないオブジェクト指向言語として誕生し，サーバ・アプリケーションに対応できる言語仕様に発展した Java は，J 2 EE によりサーバサイドのコンポーネントモデルとアプリケーションサーバモデルの提供を現実にした．

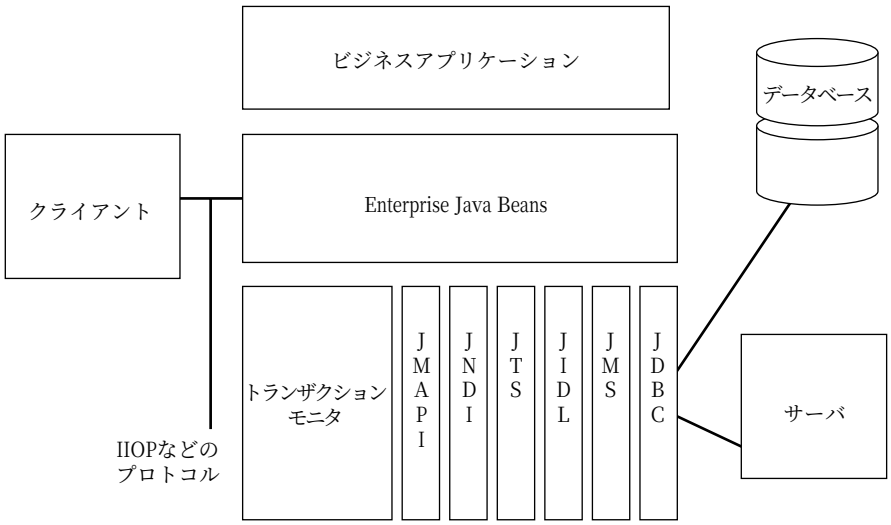

図 1　EJB アーキテクチャ

コンポーネントは，オブジェクト指向のプログラムで，他のオブジェクトとメッセージ交信により連携して機能できるよう設計されたものである．プリント基板上に電子部品を配置してコンピュータのマザーボードができあがるように，メモリ空間上に用意したコンテナに，ソフトウェアコンポーネントを配置して互いのインタフェースを関連づけて機能させる形でアプリケーションを作ることができる．コンポーネントは，ソフトウェアを再利用したり流通したりできる「部品」のことである．

これを実現するには，ビジネスロジックだけでアプリケーションを表現するようにし，ビジネスロジックをシステムレベルのサービスであるネットワークやトランザクションやセキュリティに関わるコードから完全に分離できるようにする必要がある．EJB は，ビジネスロジックだけで表現するサーバ・コンポーネントとして登場した．ビジネスロジックだけのコンポーネントであれば，モデリングツールで画面上で視覚的にエンティティを関係づけて，必要な EJB クラスを自動的に生成することができるようになってくる．

J 2 EE は，この環境を支えるための，アプリケーション・プログラミング・インタフェース（API）の集合体であり，EJB コンポーネントを異なるアプリケーションサーバで実行できるよう互換性を保証するプラットフォームとして構成されている（図 2）．Java アプリケーションサーバは，J 2 EE を実世界へ実装したソフトウェア製品として理解されるようになって来ている．プラットフォーム独立なコンポーネントソフトウェアの実行運用環境となるアプリケーションサーバにとって，EJB と J 2 EE の標準化は，データベースに対する SQL の標準化に匹敵する意義をもつ．このことはオープン環境における共通のアーキテクチャを持てることを意味し冒頭で言及した問題の解決が得られることにもなる．

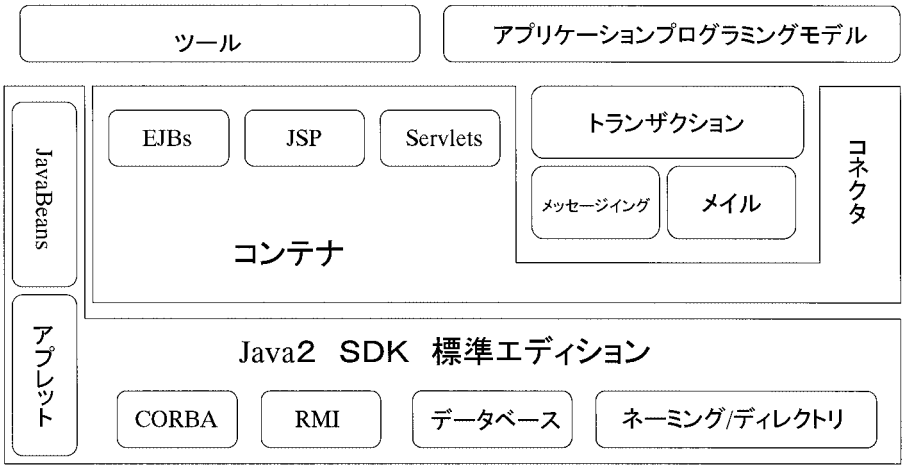

図 2　J 2 EE アーキテクチャ

EJB サーバは，EJB コンテナが EJB コンポーネントのサポートに必要な JVM と JDK のクラスを提供するサーバ側の実行環境である．EJB コンテナは，複数の EJB をカプセル化し周りの環境との仲介をする（図 3）．EJB コンテナは，EJB home クラ

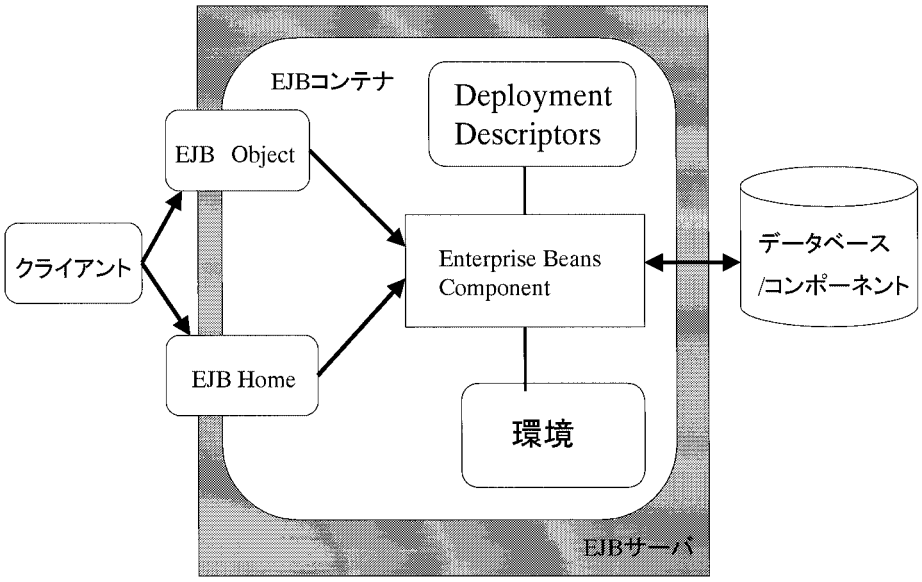

図 3　EJB コンテナ　アーキテクチャ

スと EJB object クラスをインタフェースとして，クライアントとコンテナ内の各 EJB の仲介を果たす．EJB home クラスは，実行される EJB インスタンスの親となり，新しいインスタンスを生成，既存のインスタンス間の関係を定義し，EJB のライフサイクルを管理するコンポーネントファクトリである．EJB object はクライアントから EJB へのアクセスを仲介するプロクシであり，EJB のトランザクションやセキュリティを確保しながらクライアントからのアクセスが行えるようにする．EJB object クラスは，コンテナ内の EJB インスタンスと外部の通信を仲介し，EJB のインタフェースをクライアントにルックアップサービスを提供する JNDI に自動的に登録し，EJB インスタンスが生成されるとコンテナが参照を保持できるようにする．クライアントからのメッセージは EJB object から EJB に渡されるが，これにより EJB は EJB コンテナの機能を含めて応答したり，コンテナから EJB に追加的な入力をしてクライアントの要求を処理したりする．EJB コンテナは，一つのセッションである EJB のインスタンスを複数必要とする場合，EJB home クラスは複数のインスタンスと EJB object を生成する．従って複数のクライアントが同じ EJB へ同時にアクセスすると，リクエストの数だけ EJB インスタンスを生成しスケーラブルなサーバ環境をサポートできる仕組みになっている．このために，Entity Bean と 2 種類の Session Bean という Enterprise Beans が存在している．

J 2 EE の特徴は，EJB，Servlet，JSP（Java Server Pages），XML が相互に結びついた Web アプリケーションを実現できることにある（図 4）．Servlet とはサーバ側で最初に実行される Java のプログラムである．クライアント側で機能するアプレット機能をサーバ側で実現したものといえる．Servlet は，HTML サーバアプリケーションの機能と連携できるモジュールで，HTML サーバとデータベースのリンクを扱うことができるので CGI スクリプトを代替でき，トラフィックを軽減することが

できる．また，JSP ページは XML ドキュメントにマッピングでき，XML エディタやオーサーリングツールで JSP ページを作成することができるようになる．

JNDI に関しては，技術報告をサンマイクロシステムズ社から寄稿いただいた．参考にしていただきたい．

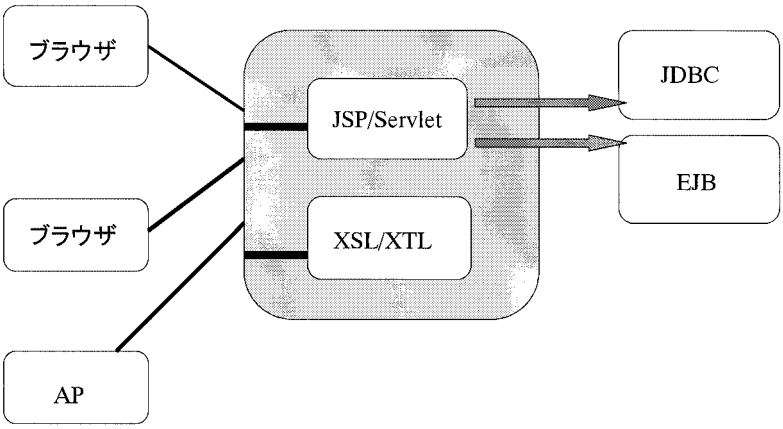

図 4　JSP と XML

3.2　部品化と再利用

業務システムを Java で開発するに当たって幾つかの解決しなければならない問題があった．オブジェクト指向開発をどう行えばよいのかという技術的な問題がある．開発を担当したプログラマ達は Java の初心者もしくはこれから学ぶものたちであった．まして CORBA 技術者はほんの 1，2 名しかいなかった．Java や CORBA といった新技術に関する部分を大多数のプログラマから切り離す策をとった．ほとんどのプログラマは Java によるビジネスロジックだけを実装することに専念すれば良いような分担を図った．このアプローチは結果として J 2 EE のアーキテクチャに沿うものになっている．J 2 EE は，コンポーネントの開発者，コンポーネントを組み立てるアプリケーションアセンブラと分業体制を明確にできるフレームワークを提供しているが，現実的にそのようなことを実施したわけである．

今日の経済界の状況では，迅速性ということが重要になってきている．その中で情報技術（IT）の果たす役割はますます大きなものになってきている．銀行や電話会社などでは新しいサービスを迅速に開発し展開するための競争を行っている．IT システムの利用なくしては業務システムは成り立たない．「ソフトウェアコンポーネントが適切に管理されることが，この競争に打ち勝つ必須条件である」と言われて久しい．プログラマの大半が他人によって書かれたコードの修復や変更に携わっているような状況においては，ソフトウェアの機能を効率的かつ時宜に応じて追加することが，差別化において重要な要素となる．「ソフトウェア部品であるコンポーネントを基本にした開発アプローチにより競争力のある製品を開発することができる」ことは理解できるがそのような部品化/再利用は成功していない．我々の開発においても部品化を意識してきたが，GUI 部品のような粒度の小さい部品の再利用は成功するにして

もビジネスロジックの部分についてはただプログラムをモジュール化すれば良いといったものではない．再利用をするための明確な筋書きと体系的な再利用を目指すソフトウェアプロセスがなければ，大規模な再利用を成功させるためにオブジェクトを採用しても効果は全くないと思われる．

ソフトウェアは，品質の低下やコストの増大を排除しながら迅速に開発されなければならない．同時に，新しい技術に対応し，メインフレームからイントラネットやインターネットを基本とした分散システムへ移行していく中で，レガシーソフトウェアの巨大な投資を置き換えなければならない．このような挑戦に対応でき顧客ニーズを満足させるためには，効果的で柔軟なシステム・アーキテクチャとソフトウェアのコンポーネント戦略に依存するところが大きい．

Java や CORBA といったオブジェクト技術を実践的に適用する中から，これらのインタフェースを重視するオブジェクト技術は，コンポーネントというオブジェクトの集合体である，粒度の大きい単位でのソフトウェア組立開発を行えることを実感した．既存のシステム開発に十分な経験と技術を持っているプログラマに対し新しい技術に対応させながら開発を進めていくためには，また再利用を大規模に実現するためには伝統的なソフトウェアアーキテクチャと開発プロセスを根本的に変更しなければならないと感じた．ソフトウェア開発を予測可能で反復可能な手続きに置き換えることが必要という信念のもとにこれまでの開発体験をふまえたものが論文「コンポーネント指向の Java アプリケーション開発技法」である．それを支えるものとして準備した開発環境が論文「Java/CORBA アプリケーション開発環境の実現」で報告されている．その後，これらの開発技法と開発環境を使用した実開発が行われその有効性が実証されている．論文「基幹系ミドルウェアのオブジェクトモデル」は特にレガシーでの基幹系システム開発の実績を踏まえオブジェクトモデルで開発できることをテーマにしている．COBOL を意識した EJB アーキテクチャという趣がある．

4. おわりに

Java の出現から 5 年が経つ．J 2 EE の発表とその実装ともいえるアプリケーションサーバ製品の出現などを見るとその成熟の早さは驚くものがある．

業務アプリケーションには，スケーラブルで，安全な標準プラットフォームが鍵を握る．J 2 EE のアーキテクチャはそれを約束するものにみえる．しかしまだまだ仕様と実装製品にはギャップが存在する．できるはずのものができないことが多々あるのが現状である．適用技術を共有しコンポーネントベースのソフトウェア開発業務を早く近代化することがオープンな時代のシステム技術者の義務だと思う．そのような視点で編集された我々の Java 適用の技術報告が広く Java によるシステム開発に関わる皆様のお役に立てば幸いである．

参考文献 ［1］ http：//www.java.sun.com
J 2 EE 関連の文献を見ることができる．

執筆者紹介　原　　潔（kiyoshi Hara）

1945 年生．1969 年京都大学理学部数学科卒業．1970 年日本ユニバック(株)(現日本ユニシス(株)) 入社．データベース管理システムの開発・保守業務を経て，オブジェクト技術関連製品(Gnosis，TIPPLER，SYSTEM ν [nju:]) の開発を担当．1997 年以後は Java を中心とした分散オブジェクト技術の適用業務に従事．日本ユニシス Authorized Java Center 代表．東京理科大学理工学部非常勤講師．現在，生産技術部情報技術室担当部長．著書：「標準 SQL プログラミング」，カットシステム，1997．共著「オブジェクト指向のおはなし」，日本規格協会，1995．「実践 SQL 教科書」，アスキー出版局，1996，共訳「JavaIDL プログラミング」，カットシステム，1998．「経営科学 OR 用語大辞典」，朝倉書店，1999．その他．