IIS による Web システム開発事例報告

Report on Web Application Development Using IIS

松 木 則 夫

要 約 本稿は，1996 年 7 月から 1998 年 3 月にかけて開発した，CAD データおよびそれらに関連する属性情報の検索システムの開発報告である．このシステムは Web によるクライアント・サーバ形態を採用し，マイクロソフト社の Web サーバである IIS のアプリケーションインタフェースである ISAPI を用いて開発した．

本システムの開発において，我々はいくつかの試みを実施し，多くの困難に直面した．

今後の開発の参考のため，

1） システムの概要：システムの構成およびソフトウェアモジュール構成など，

2） Web クライアント・サーバシステムの仕組み：Visual Basic などのツールを使用して開発する通常のクライアント・サーバとの比較，

3）「オープンシステム」開発における課題，

を中心に述べる．

Abstract This paper describes the CAD data and attributes retrieval system development started in July 1996 and released on March 1998. The system, based on Web client/server technology, uses the Internet Server Application Program Interface (ISAPI) which is the application interface of the Microsoft Web server called IIS.

We encountered a number of difficulties during our many trial attempts in the course of developing this system. This paper focuses mainly on the following matters among these difficulties as guidelines for future developments.

1） System outline : System configuration and software modules.

2） Web client/server system mechanism : Comparison with a standard client/server system developed using tools such as Visual Basic.

3） Problems of " open system " development.

1. は じ め に

本稿は，Web による情報検索システム構築事例の報告である．本システムは，Oracle を属性情報のデータベースとし，WindowsNT 管理下のファイル群の定義・検索機能を実現するものである．WindowsNT を Web サーバ機とし，WindowsNT あるいは UNIX 機にある Oracle データベースを検索・更新する．システム設計のポイントは，大量のユーザに，いかに効率的な検索を提供するかであった．このため，Microsoft の ISAPI を使い，マルチスレッドのプログラム開発を行った．

しかし，本稿の目的は本システムの詳細を報告することではなく，本システムを開発する時に直面した多くの問題点とその対策，およびその時に検討した内容について報告することである．なぜなら，個々の技術の詳細は数か月単位で陳腐化するが，オープンシステム開発における問題点は，そう簡単に陳腐化しないからである．

本稿の構成は，2 章でシステムの概要を，3 章で ISAPI を使った Web システム開発の技術的な問題点とそれらに対する工夫点を報告する．4 章は議論の章とし，技術的な側面に限らず本システム開発に至るまでに検討された内容を報告する．

2. システムの概要

2.1 システムの目的

本システムの目的は，複数のシステムで作成された情報を社内外からアクセス可能にすることである．検索システムは一般ユーザのための「一般検索機能」と，管理者のための「管理機能」に大別される．

「一般検索機能」は，顧客情報，プロジェクト情報，オーダ情報，文書に関連する情報，文書本体（CAD データや文書データ）を，Web ブラウザを使って検索する機能である．検索には「あいまい条件」を指定し，複数のデータベースにあるテーブルを組み合わせて検索が行われる．検索結果の属性情報は表形式で，また文書本体はプラグインやビューイングシステム（市販，自社開発システム等）で閲覧される．

「管理機能」は，検索システムがもつデータベースの項目を新規作成，更新，削除する機能を持つ．「管理機能」を使うことができるユーザを登録する機能もあるが，本機能の使用頻度は少なく，そのユーザ数は限定される．

また本システムは，管理システムではない．管理システムとなるためには，文書本体のリビジョン管理やアクセスコントロールなど，一般的な PDM（Product Data Management）システム[1]と同等の機能を持つことが必要だが，本システムにはそこまでの管理機能はない．その理由はコストと期間の制約である．このため，Web ブラウザから，必要な情報が「もれなく」,「効率良く」，検索できることに機能を絞ったシステム開発となった．

2.2 システム構成

本システムの構成を図 1 に示す．各拠点には，拠点を代表する Web サーバを一つ設置する．Web サーバには，当該拠点の情報を集約する Oracle データベースが一つ対応している（これをプライマリ Oracle データベースと呼ぶ）．Web サーバからは，プライマリ Oracle データベース以外の Oracle データベースへもアクセス可能とする．すなわち，Web サーバから複数の Oracle データベースにアクセスする機能がある．

Web サーバはすべて WindowsNT 4.0 が稼働する PC である．プライマリ Oracle データベースは WindowsNT あるいは UNIX 機に存在する．クライアントの OS は拠点によって異なるが，Windows 95 だけでなく，Windows 3.1 も存在する．

ソフトウエア構成は図 2 に示す. Web サーバとしては, Microsoft 社の IIS[2]（Internet Information Server）を，クライアントとしては，Web ブラウザの Netscape 3.01 以上，Internet Explore 3.02 以上を想定している．文書本体のビューワについては拠点ごとに異なっている．

2.3 システム動作手順

本システムの標準動作を解説する．「一般検索機能」を使う場合は，ユーザの作業する拠点の Web サーバにまずアクセスする．そこで，情報を知りたい拠点を設定す

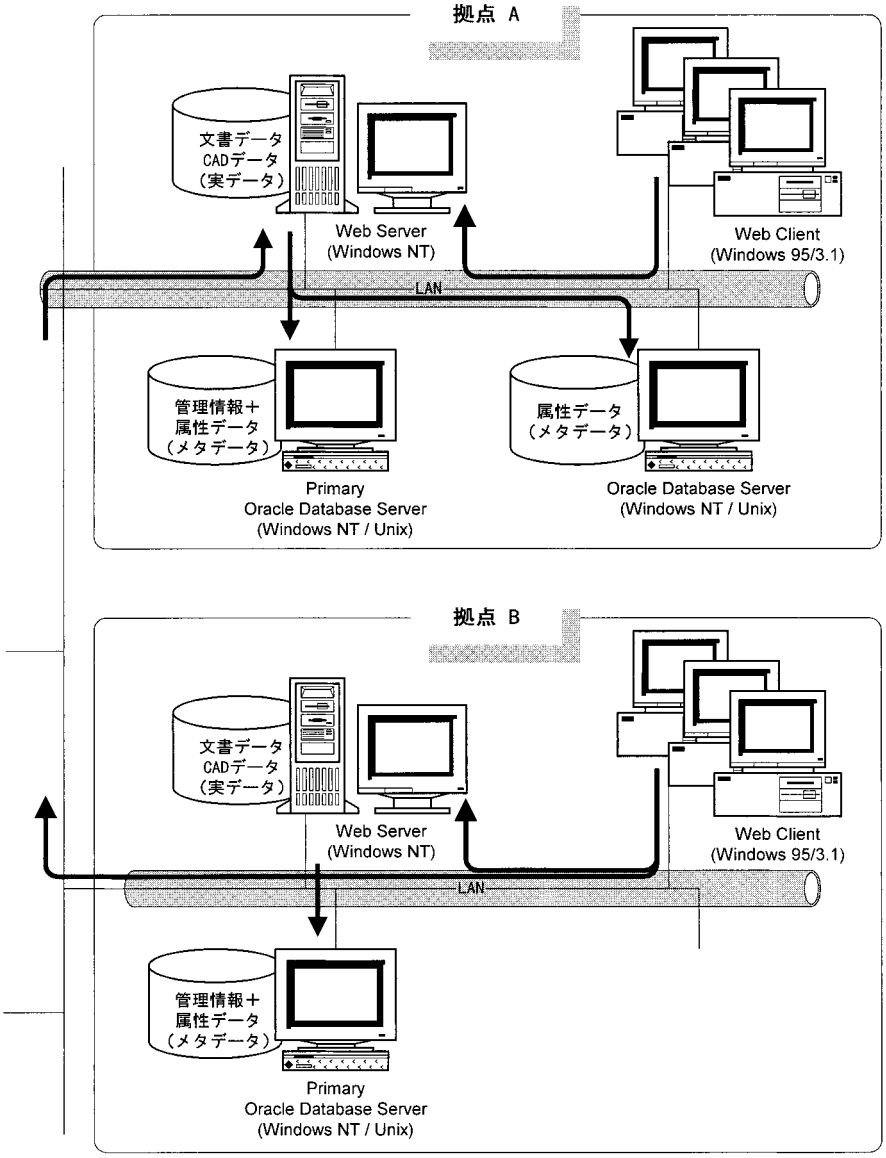

図 1　システム構成

ると，目的の場所のWebサーバに接続する．つぎに，データベース名（複数可）を指示し，知りたい情報の種類を指定する．情報の種類を特定するための条件項目に対して，あいまい条件を指定して，情報の検索を行う．一般ユーザのログインの制御は，WindowsNTのドメインとユーザ管理[3]で行っている．すなわち，本システムの実行ファイルの実行権とそれに対するユーザの権限を組み合わせて制御する．

「管理機能」についても動作は基本的に同じである．「管理機能」の場合は，一般のユーザのログイン制御だけでなく，データベース内にユーザIDとパスワードを保持し，ログインユーザのチェックを行う．

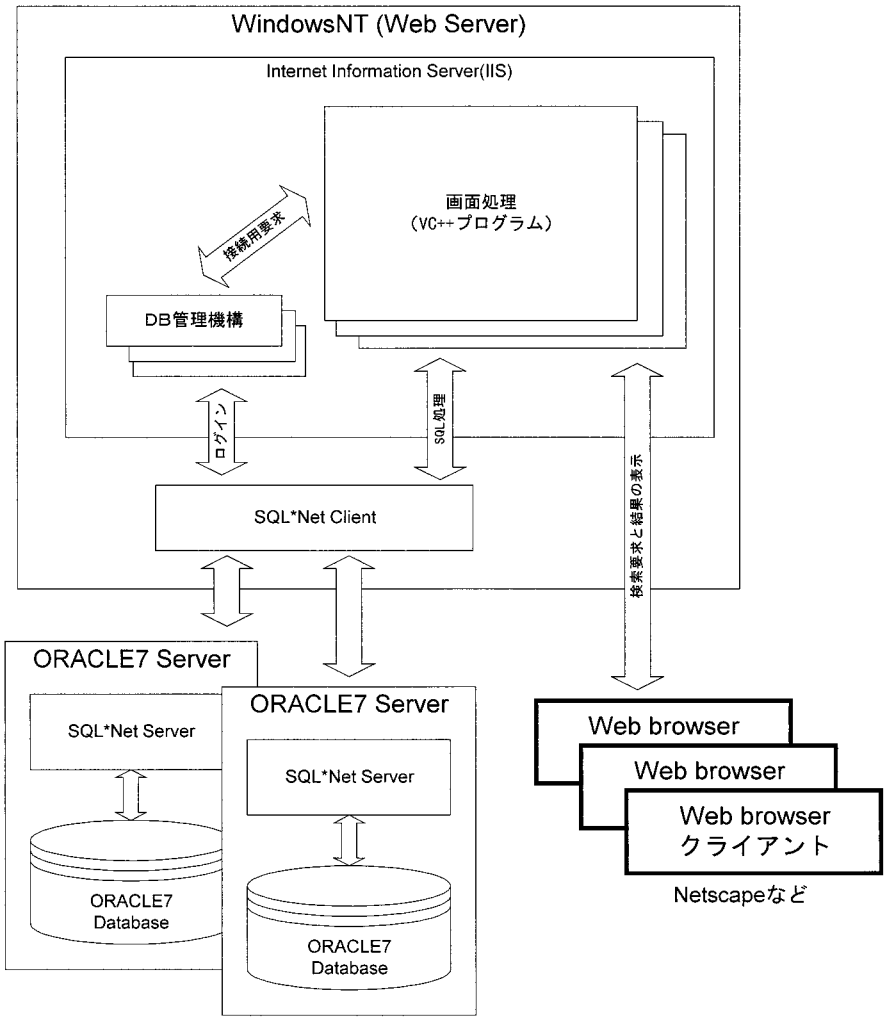

図 2　ソフトウエア構成

3.　技術的な課題と工夫点

本章では，システムを Web で開発する上で直面した問題点や課題と，それらに対してどのような工夫を考案し，対処したかを述べる．

3.1　Web によるクライアントサーバシステムの問題点

一般的に Web システムでは，ステートレス，HTML テキストの大量コード作成(開発負荷)，パフォーマンスの悪さが問題であり，これらの点について以下検討を行う．なお，以下の検討では，JavaScript を使う方法を検討対象とはしていない．これは，本システムが，JavaScripit が使用できない Web ブラウザも対象としていたためである．

3.1.1　ステートレス

Web システムのメリットの一つは，サーバがユーザを管理しないため，サーバサイドのシステムの負荷が少ないことである．これは，ここまでインターネットが普及

した最大の理由の一つであり，HTTP（Hyper Text Transfer Protocol）の仕様の特徴でもある．これは，状態（ステート）を管理できないという意味で「ステートレス」と呼ばれる．

しかし，Web システムを使ってクライアントサーバシステムを構築する場合，Web サーバが Web クライアントの状態を把握できないことは問題である．たとえば，サーバからデータがクライアントに正常に渡ったかを，サーバは知り得ない．

すなわち，ステートレスであるため，「処理の論理的な整合性の確保」，および「連続した画面（処理）に共通な情報の保持」を，どのように扱うかが問題となる．これらに対し，今回の開発で実施したことを述べる．

1)　処理の論理的な整合性の確保

処理の論理的な整合性を保つため，処理はすべてサーバ側との一回のセッションで完結するものとした．メイン処理の途中で，たとえば社員番号から氏名を取り出す処理といった中間的な処理を行ない，その結果を確認させてから，メイン処理を開始するようなことは避けた．

これは，特に目新しいことではなく，通常の Web システムでも行っていることである．ただし，一般的なクライアントサーバシステムの設計をそのまま，Web システムで実現しようとすると直面する困難である．

2)　連続した画面（処理）に共通な情報の保持

連続した画面に共通な情報の保持は Web システム開発において最も問題になる点である．ステートレスのため，何らかの形でクライアント側に状態を持つ仕組みが必要となるからである．そこで，次のような方法が考えられる．

①　クライアント側にすべての情報を持つ

これは，「隠しフィールド（Hidden）」に，ユーザ固有の情報を持つもので，通常の方法である．ただ，情報量が少ない場合問題はないが，大量のデータを隠しフィールドに保持することは，通信の負荷，クライアントの負荷を考えると現実的な方法ではない．

②　一部をクライアント側に，その他をサーバ側に持つ

キーとなる情報だけを隠しフィールドに持ち，そのキーから検索できる情報をサーバサイドに持つ．サーバに持つ方法にも二つある．

a)　ファイル：サーバ内にファイルを作成し，そのファイル名をキーとする方法．CGI（Common Gateway Interface）を使った場合に一般的な方法である．絞り込み検索時に，検索途中の情報を保存する例が報告されている[2]．ただし，ディスク容量の見積り，ファイルの削除のタイミング，パフォーマンス等の課題が残る．

b)　メモリ：サーバ内のメモリに保存する方法．Web サーバが稼働するメモリ空間がクライアント間で共有可能であれば可能であり，効率的である．メモリが十分にあれば，パフォーマンスが良い．a）の方法のようにファイル削除のタイミングを気にする必要がない．ただし，CGI では困難な方法であり，Web サーバの種類，OS の種類に依存する．

今回開発した図面情報システムでは，①と②の b)の方法を使った．このため，CGI

ではなく，Microsoft IIS（Internet Information Server）のサーバサイドアプリケーション作成の方法である，ISAPI（Internet Server Application Programming Interface）をWebアプリケーション開発の方法として採用した．ISAPIの詳細については，3.2節で解説する．なお，CGIとISAPIの主な特徴の比較を表1に示す．

表1　CGIとISAPIの主な特徴と比較

	CGI	ISAPI
作成単位	独立した実行可能モジュール.	IISの下で動作する動的なライブラリ形式(DLL).
実行形態	サーバがプロセスを起動し，プロセス内でCGIプログラムが稼動する.	初めての実行のとき，スレッド内へDLLが読み込まれ実行される．次回からはメモリ内に常駐するDLLが起動される.
実行負荷	プロセス起動の負荷が大きい.	スレッド生成をするだけで負荷は少ない.
言語	Perl, Visual Basic, C言語など自由に選択可能.	Microsoft Visual C++のみ.
汎用性	OS，Webサーバの種類に依存せずに実行可能(ロードモジュールの再リンク等の作業は必要)	Windows NT，IISのみで実行可能.
要員	独立したモジュールの開発であるため開発人口は多い.	VC++およびIIS，Windows NTのかなりの知識が必要．要員は少ない.

サーバサイドプログラムの方法は，CGIか，ベンダーが独自に設定する方法のいずれかを採用することになる．CGIは，汎用性があり，Webサーバに依存しない方法で，最も広く採用されている．ベンダーに依存する方法としては，Netscape社のNSAPIとMicrosoft社のISAPIが有名である．ベンダーに依存する方法は，汎用性に乏しいが，使用環境が限定される場合には，有力な選択肢となる．

今回の開発では，すべてのWebサーバがWindowsNTであったことにより，IISをWebサーバとして選択するのが自然であった．IISを前提にするとサーバサイドのアプリケーションをISAPIとすることも自然と思われた．さらに，ISPAIを選択することにより，オブジェクト指向に基づいた設計と，Visual C＋＋でMFC（Microsoft Foundation Class）[4]を使って開発することが可能になった．

この選択は，次節以降に説明する様々な利点をもたらした．

3.1.2　HTMLテキストの大量コード作成

Webシステム開発におけるプログラムの多くはHTMLで記述されたテキストの生成・解釈に費やされる．このため，システム開発時のコード量は大量となる．これは，制御コードによるプロトコルに比べ，Webシステムが使用するHTTPがテキストベースのプロトコルであることが原因である．

したがって，システム開発時のプログラミングにおいて，大量のテキスト文字列を操作するプログラムを開発することになり，タイプミスによるバグの発生の危険性が大きくなる．また，開発の負荷も当然多くなる．

ISAPIは，C＋＋言語で記述されるため，クラスやオブジェクトの概念が使用でき

る．このため，画面で使用するヘッダーやフッター，ボタン処理などを共通のメソッドにすることによりプログラムを再利用することができる．

また，CGI による開発と比較すると，ISAPI による開発では，実行時の変数や処理を複数の画面，複数のユーザに共通に使うことができる利点がある．ISAPI はマルチスレッドのため，IIS のメモリ空間を複数のユーザが共有できる．これは，一つのオブジェクトインスタンスを IIS サーバ内で共有できることを意味する．この機能を上手く使えば，HTML テキストコードを共有化する道も開ける．

残念ながら今回の開発では，プロトタイプから開発までの期間が短かく，仕様変更も多く，十分なプログラム設計の余裕がなかったためプログラムの共有化について特筆すべき成果はあげられなかった．今後の開発では，この点を反省し，共有化可能なクラス設計を行いたい．

3.1.3 パフォーマンス

Web システムのパフォーマンスの欠点の一つとして，HTTP がテキストであるため，クライアント・サーバ間のデータ転送量が多くなることが挙げられる．これは避けられない問題である．さらに，画面毎に起動されるタスクがサーバシステムやデータベースシステムに与える負荷によりパフォーマンスが低下するという問題もある．特に，CGI の場合，画面が変わる毎にプロセスが起動され，その都度データベースへのログイン処理が必要になる．ユーザ数が増えたとき，この処理は大幅な効率低下につながる．しかし，これは回避可能な問題である．

ISAPI では，画面毎にプロセスが起動されることはない．画面毎に行われるのは，IIS プログラム内部のオブジェクトの生成だけであり，システムに与える負荷は少ない．ユーザ数が増えたとき，CGI と ISAPI のシステムに与える負荷の差は大きい．

また，データベースログインは時間がかかる処理である．ユーザの待ち時間が増えるという意味で，これもパフォーマンスを低下させる要因である．

データベースログインについては，今回のシステムでは次のような工夫を行った．簡単に言えば，Web サーバである IIS 起動時に，システムで設定した（実際にはレジストリに記述した）数だけ，データベースへログインしておき，そのログインを複数の Web クライアントが共同利用する，というものである．ユーザ数（Oracle システムがインストールされたクライアント機器の数）は Web サーバの数，すなわち 1 であり，クライアント数は同時ログイン数である．したがって，今回のシステムはパフォーマンスと顧客の予算との関係でログイン数を決めることになる．ただし，開発時に行ったテストでは，ユーザ数とパフォーマンス向上は単純に比例しない，という結果がでている．

また，検索の結果が大量の場合，次のような工夫をしている．通常，検索結果はクライアントに転送される．サーバに仕組みを設けた場合には，サーバ側に検索結果を一時保存し，ユーザには，ユーザ画面で見ることのできるデータのみを送る方法がある．さらに，検索中にデータベースに更新がない場合には，もっと単純な方法がある．それは，画面に必要なデータのみを毎回検索する方法である．今回のシステムでは後者の方法を採用した．この結果，検索結果が大量でも，転送の時間は一定に保つことができ，パフォーマンスの向上に寄与した．

3.2 マルチスレッド環境下の ISAPI

マルチスレッドとは，プロセス中に複数のスレッド（制御の流れ）を生成し，制御する機構を言う．したがって，マルチスレッドは分散，並行処理において強力なプログラミング手法であり，様々な研究と実装が行われている．大別すると，POSIX 系のマルチスレッドと Microsoft のマルチスレッドがある．今回使用したマルチスレッドは Microsoft の WindowsNT 用のものである．

ISAPI は，Microsoft 社の WindowsNT 上の Web サーバである，IIS のアプリケーション作成の方法である．ISAPI アプリケーションは，DLL（Dynamic Linking Library）の形式で動作する．以降，ISAPI による開発の特徴を述べる．

1） オブジェクト指向

開発言語が C＋＋であるため，オブジェクト指向分析やオブジェクト指向設計に基づく開発が自然に可能である．今回の開発では，Oracle とのインタフェース部分とユーザ管理の部分にオブジェクト指向設計を生かした．また，開発環境である Microsoft VC＋＋に用意されている，オブジェクトクラスライブラリである MFC（Microsoft Foundation Class）[4]を利用できるため，ユーザの排他制御が容易に実現できる．ただし，これらを自在に使える開発者が絶対的に不足している．

2） 共通なメモリ空間

マルチスレッド環境では，ISAPI で作成されたアプリケーション群は IIS と同一のメモリ空間内で動作する．このため，アプリケーションの実行モジュールは DLL として作成する．DLL が起動されると IIS のメモリ空間内にロードされ，ロードされたモジュールは IIS が動作している間，メモリ内に存続する．このため，プログラムはリエントラントな構造にする必要がある．

言い換えると，ISAPI のアプリケーション開発とは，IIS の一つのモジュールを開発することである．このため，パフォーマンスを良くすることができる．さらに，クライアント間での情報交換や，ステートの管理，運用管理などを Web システム上で実現することができ，かなり複雑なビジネスモデルを持ったクライアントサーバシステムを実現することができる．今回のシステムでは，データベースのログイン情報をクライアント間で共有させたことにより，パフォーマンス向上とライセンス管理が可能となった．

しかし，各モジュールの不具合はサーバシステム全体に影響することになるため，運用開始までに十分なテストが必要になる．また，DLL を入れ替えるためには，IIS を一度停止する必要がある（DLL を常駐させない設定があることが後に判明したが本システムでは適用していない）．これらの点は，CGI の開発に比較して，プログラム開発側への負担となる．プログラム開発者の数が増えると，テスト作業の効率を指数関数的に低下させる危険がある．

3） Microsoft 仕様

ISAPI の最大の特徴は，良い意味でも悪い意味でも，Microsoft の仕様である，という点である．良い点は，開発動作環境がすべて Microsoft 製品であることの相性の良さである．サーバの OS が WindowsNT であり，開発環境が Visual C

＋＋である．ISAPIの開発およびデバッグのためには，Visual C＋＋とIISとの相性の良さが非常に大切であった．また，Webクライアントの認証は，Windows NTのドメイン・ユーザ管理を利用できる．

悪い点は，Microsoft以外の競合する製品との相性の悪さである．Microsoft自体にSQLServerというデータベース商品があるため，Oracleとの相性が良いとは決して言えなかった．

また，コードの再利用性にも問題がある．IIS以外のWebサーバへ展開する場合には，ソースコードはそのまま利用できない．この点はCGIによる開発と比較すると明らかに劣る．今回のシステムはターゲットがWindowsNTであったこと，効率重視の開発であったことなどの理由により，ISAPIの選択に誤りはなかったと思われるが，常に正しい選択とはなり得ないことも事実である．

3.3 クライアントサーバ構成の方法

ここでは，データベースをつかったクライアントサーバのシステム構成のいくつかの方法を比較検討する．Oracleなどのデータベースを使ったクライアントサーバの形態としては，次のような方法が考えられる（図3）．1）の方法は，クライアントとサーバの間をOracleのSQL＊Netを使うものである．クライアント側はVisual Basicなどの GUIツールを使って構築する．これをVB型と呼ぶことにする．2）の方法はWebシステムでは一般的なCGIを使った方法である．これをCGI型と呼ぶ．3）の方法が今回のシステムで使用した形態である．これをAPI（Thread）型と呼ぶ．

1） VB型

システム開発としては分かり易い形態である．データベースのスキーマを決め，クライアントの画面推移を決めれば，開発するものは主にVisual Basicなどのクライアントのプログラムであり，開発人口も多い．難点としては，ビジネスモデルがすべてクライアントシステム内部に実装されるという点（2層モデル）が一つ．もう一つが，クライアント毎にSQL＊Netのインストールとクライアントプログラムのインストールが必要になる点である．クライアントにインストールするプログラムが多量であると，システムの仕様変更時の再インストールの手間が大変である．また，クライアントOSの変更や混在などの影響を受けやすく，運用上の問題を引き起こしやすい．また，ユーザ数，データ量の増大に伴うパフォーマンス問題はOracle任せとなり，Oracleのチューニングが延々と続く可能性がある．小規模開発では問題がない方法と思われるが，大規模なシステムにこの形態を採用すると，問題が起こる可能性が大きい．

2） CGI型

事例の多い方法であり，Webでは一般的な方法である．今回の開発でも，運用管理者用のシステムに，この形態を採用しているモジュールがある．この形態では，データベースシステムとのインタフェースはすべてサーバ側に記述されている．クライアントの入力画面情報は要求の都度すべてサーバ側から転送され，結果を受けてサーバ側で処理される．この方法では，クライアントにNetscapeやInternet Explorerなどの Webブラウザがインストールされていれば良く，システム運用管理の手間はかなり減る．

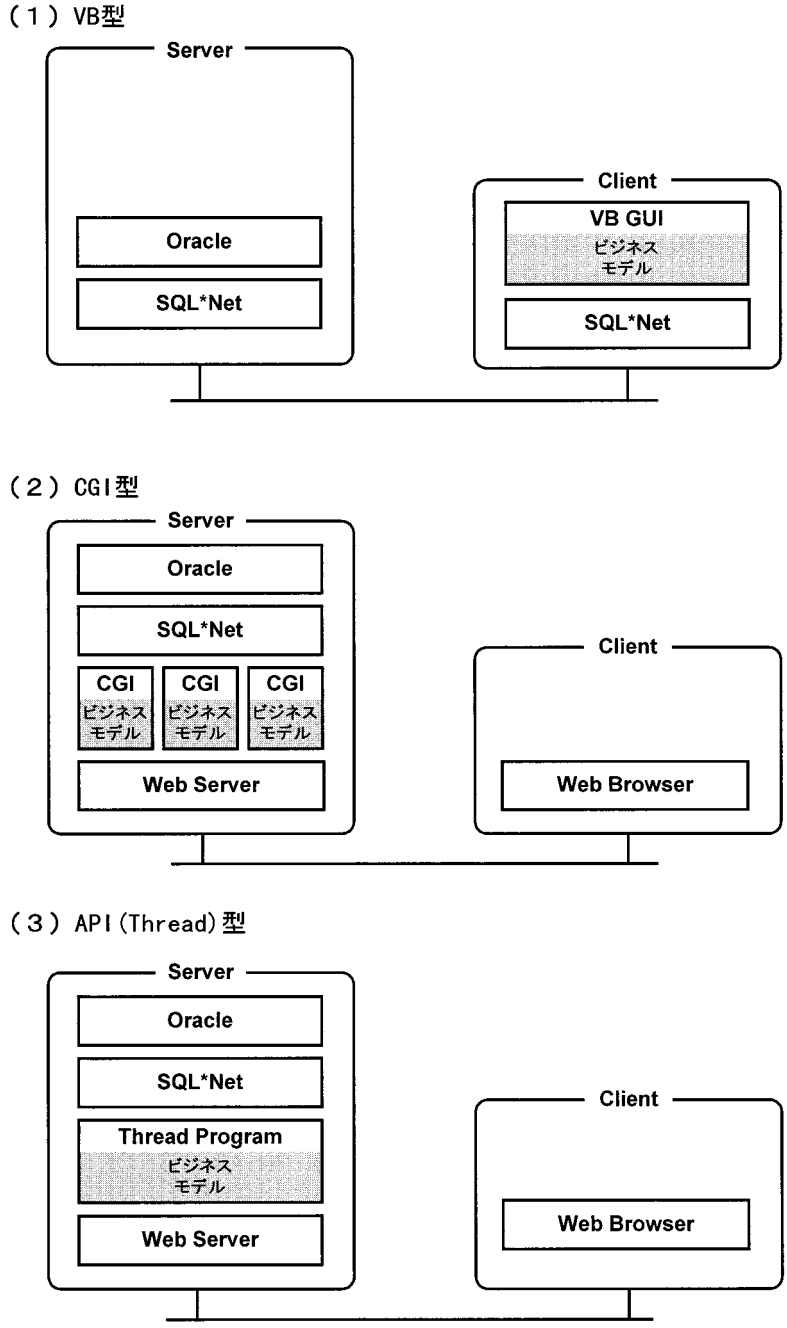

図 3　クライアントサーバの形態

ただし，今まで述べてきたように，CGI プロセスの起動の負荷や，クライアント間のデータの受け渡しがファイルになるという困難がある．また，いわゆるビジネスモデルの記述が，バラバラの CGI モジュールに散在することは避けられない．このため，例えば Web からデータベースへのログインユーザ数を制御することは簡単ではない．2 層モデルではないが 3 層モデルとも言い難い形式であ

る．

CGI型は独立性・再利用性にすぐれ，開発が比較的容易である．この方法も大規模システムではパフォーマンスの問題が発生する．

3) API（Thread）型

クライアントとサーバの役割という点では，CGI型と同じである．CGI型との違いはサーバサイドの構成方法である．サーバサイドが1つのメモリ空間を共有するプログラムとして動作するため，クライアント間や，クライアントとデータベース要求処理など，並列して行われる処理間での協調が可能になる．この方法は，効率も良く，ビジネスモデル記述の範囲が拡大する（3層モデルが可能）．

ただし，このThreadのAPIを使う方法は，サーバ機のOSに依存した設計と開発が必要であるため，汎用性に乏しい．また，開発者も不足気味である．したがって，手軽に開発したいという小規模開発より，手間がかかっても効率を重視する大規模システム開発に威力を発揮すると思われる．

3.4 WindowsNT上のIISとOracleの連携

この章では，Oracleとの連携の具体的な方法，すなわちOracleインタフェース，ライセンス管理，セッションの管理について，他の方法[5]とも比較しながら本システムで採用した方法について述べる．

3.4.1 Oracleデータベースとの接続方法

Oracleデータベースシステムとクライアントサーバシステムとの接続には，ODBC，OLE，OCIの三つの方法がある．今回使ったものは，OCIである．

1) ODBC

ODBC（Open Database Connectivity）はMicrosoftが提唱している，データベースインタフェースである．OracleもODBCインタフェースライブラリを用意している．どちらかといえばSQLServerにとって都合が良く，Oracleにとって都合が良いとは言い難いインタフェースのようである．少なくとも，検討時点（1996年夏）ではMicrosoftが提供した，Oracle用のODBCサンプルプログラムは動作に問題があった．

2) OLE

OLE（Object Linking and Embedding）機構を使う，Oracle社が用意したC＋＋インタフェースライブラリであり，Oracle Objects for OLEという資料がある．ただし，検討時点ではOLEの制限から，マルチスレッド環境でこのインタフェースを使用することは困難であるため，今回の開発では不適当であると判断した．

3) OCI

OCI（Oracle Call Interface）[6]は，Oracleインタフェースの最もプリミティブなC言語インタフェースライブラリであり，今回のシステム開発ではこのインタフェースを使用した．当部の技術資産として，OCIインタフェースライブラリがあり，流用可能であったことが利用に至った最大の理由である．ただ，Oracle社のパッケージには説明がなく，仕様書はPro＊CというC言語パッケージに付属している．このため，今後のサポートに一抹の不安がある．また，Oracle社

の提供するマルチスレッド対応の DLL の不具合のため，開発初期の数ヶ月トラブルの追求で非常に苦労した．

3.4.2 ユーザとクライアントのライセンスの考え方

Oracle のユーザとクライアントの考え方は使用許諾証の記述にある．ユーザ数とは，Oracle 社が提供するプログラムが動作する機器の数である．クライアント数とは，そのプログラムを使用して同時に Oracle データベースにアクセスする使用者の数である．TP モニターという，リアルタイムなトランザクション処理機構の場合には，SQL＊Net の介在なしで直接，クライアントがデータベースにアクセスするため，クライアントプログラムすべてが Oracle 社が提供するプログラムとみなされるようであるが，Web サーバの場合は異なった見解が Oracle 社の提供する資料に示されている．

今回のシステムの場合，Oracle 社の製品は一切クライアント機器にはインストールされない．また，サーバにおける Oracle とのアクセスには，SQL＊Net を経由している．

Web システムの登場により，ライセンスの考え方は，Oracle だけでなく，クライアントプログラムの数により利益を得ていた多くのプロダクトに影響を与え，見直しが必要になるはずである．当部で扱っている PDM システムでも，Web 化に伴い，ライセンスの考え方の見直しがされている．特に，Oracle の場合はライセンス管理をシステム機構として実現していないため，話をさらに複雑にしている．この問題は，法律的な側面もあり今後の動向に注意が必要である．

3.5 画面の構成

今回開発したシステムの画面では，入力形式の統一，あいまい検索，プラグインによる文書データの表示などに工夫を行った．

画面設計の前提として，Netscape，Internet Explorer，Notes Navigator などの複数のブラウザをサポートする必要があった．このため，フレームや Java など，画面制御についての新規機能の使用は，ブラウザのサポートに差があるため，あえて避け，できるだけ標準的な機能を用いている．

まず，入力形式は，視認性向上とレイアウトの統一感を与えるため，項目名称を右詰，入力フィールドを左詰にしている．また，あいまい条件をコンボボックス化し，ブラウザの機能に依存しない画面を作成するという要求があったため，back や home のボタンを入力形式の一部として作成した．

あいまい条件は，文字列，数字，日付について，表 2 のような条件を文章で表現し，入力は，ユーザがコンボボックスから選択する方式である．これは，システムについてほとんど知識のないエンドユーザに分かり易くするためと，入力条件の項目をコンボボックス内のものに制限させる意図があった．

テーブルの検索結果は，1 画面に一定（現在 20）行（レコード）の検索結果を表示した．このため，「パフォーマンス」の章で説明したように，検索はすべて 20 行分のみを検索結果としてサーバからクライアントに転送することで，ユーザが無駄に待つ時間を減らすことができた．画面の下に「次」,「前」の表示があり，ボタン操作が行われると，再度 20 行分の検索が行われる．

表 2　あいまい検索条件

型	条件
文字列	を含む
	と完全に一致する
	を先頭に含む
	を後方に含む
日付	以前
	以降
	の当日
数値	＝【結果】
	≧【結果】
	＞【結果】
	≦【結果】
	＜【結果】

Webシステムによる画面設計の問題として，入力データのチェックが入力された時点でできないことがある．これは，処理がすべてサーバ側で行われ，画面内容を転送されて始めてデータのチェックが行われるためである．数字フィールドや文字フィールドの設定，データベース検索の制御文字，たとえばOracleの制御文字である「%」などの入力を防ぐことが，クライアント側でできないことは設計上の制限となった．今回のシステムでもこの問題を回避することはできなかった．

4. 議　　論

システム開発の場合，技術的課題を解決する方法だけが選択されるとは限らない．防衛など特殊な例を除き，コスト，システム資源・ハードウェア資源，関連メーカとの関係，将来性など，複雑な要素を考慮して決定される．

本システム開発当初（1996年6月），3.1章で議論した問題点があること，Webによる本格的なシステム開発事例が少ないことにより，Webによるシステム開発に懐疑的な声もあった．それが，最終的に開発実施に至った理由は，

1)　Webがオープンな技術で，将来行くべき方向であると考えられたこと，

2)　PDMパッケージを使った場合と比較して開発費が1桁安いこと，

と思われる．

以下の章では，技術的な側面に限らず，本システム開発に至るまでに検討された項目について議論する．

4.1　なぜWebシステムなのか？

Webシステムの大きな特徴は，クライアントとサーバの通信制御プロトコルがHTTPという形でオープンなことである．このため，NetscapeとMicrosoftのIISのように，クライアントとサーバが別々のベンダーの製品でシステムを構築することができる．そして，良い製品が出たならば，クライアントだけあるいはサーバだけでも交換可能である．

オープンな技術は，公開され，競争され，批判され成長する．現在，Internet Explorerや Netscapeのセキュリティホールが問題になっている．このため，やはりPC

のソフトではだめで，UNIX が信頼できるという意見もある．しかし，UNIX も元々はひ弱な仕組みであったのが，公開され，競争され，批判されてここまで来たのである．これは，インターネットにより生まれた技術の特徴である．

同様にフリーソフト，シェアウェアなどに対し，信頼性やサポートの問題が言われることが多い．しかし，情報産業が拡大し様々なベンダーが参入したため，競争も激化し，ベンダーの商品でも，いつサポートを停止するかもしれないし，ベンダー自身いつまで存続するかの保証がない．

このように，Web を代表とするオープンな技術は，クライアントサーバシステム構築でのベンダーの影響を減らし，ユーザにとって選択肢が拡大するというメリットを持つシステム作りを可能にする．当然，ベンダー間の競合により，価格も下がる．この事実に気づいたユーザは，Microsoft 一色の仕組みや，ホストにしがみつく提案を拒絶し始めている．今後も，この潮流が変わるとは思えない．この環境のもとで，システム化の技術を模索せざるを得ないと思われる．

4.2 開発すべき範囲はどこまでか？

システム開発で問題になるのは，どこまで市販のパッケージを利用し，どこまでを開発するのか，ということである．一方の意見は，すべて市販のソフトウェア製品を組み合わせ，できるだけなにも開発しない方がよい，というものである．もう一方の意見は，市販のものは信頼できない，あるいはかゆい所まで手が行き届かないので，できるだけ専用に開発すべきである，というものである．

工数や利益を度外視して，なんでもかんでも作るというのは，論外であるが，利用できる技術，パッケージを無視して作りたがる傾向が開発担当のエンジニアに多いのは事実である．開発するのは，ある面で楽しいことだからである．しかし，何でも組み合わせれば良いのだ，というのも疑問が残る．情報関連の技術の進展は早い．例えば，開発したこともない技術の適用限界をどのようにして知りうるのか？ベンダーのパンフレットをたよりに，Java で開発されたモジュールを組み合わせたシステムに責任が持てるだろうか？

今回のシステム開発の前提は，クライアントは，Netscape や Internet Explorer など市販のブラウザで，Web サーバの OS は WindowsNT で，データベースは Oracle である．画面モジュールはどちらにしろ開発が必要であるから，残った部分で市販パッケージが使用可能なものは，Web サーバと Oracle のインタフェースモジュールである．

では，なぜ Oracle Web サーバを使わずにインタフェースモジュールの開発を行ったのか？これには様々な要素があったが，結論としては，Web サーバと RDB（Relational DataBase）とは，機能が違うということである．機能が違うものを一体にしたものは，システムが実現すべき仕様を制限することが多い．

オープンな環境におけるシステム開発では，特にモジュールの機能を明確にすることが大切である．そして，機能ごとに作られたモジュール間のプロトコルの仕様が国際標準，あるいは，それと同等の業界標準化団体等の推奨する規格に準拠することも同様に大切であると考える．なぜなら，このように開発されたシステムでは，仕様変更や機能追加の時に，モジュールが交換可能になるからである．

4.3 プロトタイプの必要性と危険性

今回のシステム開発は，1年間で，前半の6か月がプロトタイプという位置づけであった．期間的に厳しいなかでも，どんなシステムになるかのイメージを掴むためにプロトタイプは必要である．イメージを掴むというよりも，Webシステムのなかで何ができ，何ができないのかを正確に理解し，要求仕様を正しく決めるために必要であると思われる．

しかし，プロトタイプから本番システム用に仕様の見直しを行ったとき，プログラムは大幅な変更となった．表面上，画面のレイアウトなどはあまり変化していないように見えるが，画面の推移が変わり，一部のデータベーススキーマが変更にされた．また，要求する検索項目が複数のテーブルに渡るものもあり，複雑さも増した．

このため，Oracleのインタフェースモジュールについては再利用が可能であったが，プログラムの大半を占める，画面生成・制御のプログラムのほとんどが再利用できなかった．Webシステムの特徴として，前述したように「隠しフィールド」を使って画面間の情報の受け渡しているため，画面の推移の変更は多大な影響をプログラムに与える．さらに，HTML生成は前述したようにプログラムの量が多い．このため，Webシステムのプロトタイプは通常の画面生成の開発に比較して，捨てるプログラム量が多い．

プログラミングという観点では，Webシステムのプロトタイプ開発では，開発担当者の訓練という意味と，開発量の見積りの精度を上げることと割り切り，捨てる覚悟が必要と思われる．

5. 今後の課題

まず，WindowsNTサーバとIISの信頼性・安定性の問題が最大の課題である．1996年8月の時点では，使用バージョンはWindowsNT 3.51，IIS 1.0であったが，いくつか問題のあることが後で判明した．特に，ユーザの実行権の制御がIIS 1.0ではうまく機能していない．ユーザのモジュールの使用権利の設定は，WindowsNTのドメインとユーザ管理を使うことが前提であり，IISにおける実行権の管理はDLLプログラムファイルの所有権で制御されるはずであるが，これが正しく機能していなかった．IIS 2.0以降では解決していることがわかったが，IIS 2.0が稼動するのはWindowsNT 4.0だけである．現在，WindowsNT 4.0に変更され，この問題は解決したが，3.51から4.0へのアップグレード費用が発生した．今後，このようなことが起きたとき，責任の所在とコスト負担が問題となる．

次に，いつまで今回開発したISAPIプログラムが使用できるか，という問題がある．Microsoft社が何時まで，現在の仕様のISAPIをサポートするかがわからないからである．本来，マルチスレッド機能とISAPIを分離した設計を行うことが望ましいが，時間的な制約とコストの関係で，今回の開発では困難であった．ただし，インターネットで最もシェアのあるフリーWebサーバであるApacheがバージョン1.3でISAPIをサポートし，現在，WindowsNT 4.0で稼動しているという報告があり，今後の動向が注目される．

6. おわりに

当社においてもWebを使ったシステム開発は急増しているが，ほとんどの場合がCGIによる開発で，マルチスレッド環境下でのIISサーバプログラム開発は少なく，今回の開発でも参考となる資料がすくなかった．本稿が，今後のWebシステム開発に多少なりとも役立てば幸いである．

参考文献
［1］ 柴田晴康, PDMシステムによる業務の効率化, ユニシス技報, 通巻46, Vol. 15 No. 2, 1995年8月, pp. 38–69.
［2］ ラリー・バドニック, WindowsNT Web Server, インターナショナルトムソンパブリッシングジャパン.
［3］ Windows NT 4.0 リソースキット, アスキー出版.
［4］ David J. Kruglinski, Inside Visual C＋＋, Microsoft Press.
［5］ David Lockmann, Oracle 7 SQL 入門, アスキー出版.
［6］ Oracle コール・インタフェース・プログラマーズ・ガイド, オラクル社.

執筆者紹介 松 木 則 夫（Norio Matsuki）
1954年生．1980年早稲田大学大学院理工学研究科数学専攻修了，同年日本ユニシス(株)入社．主にCAD/CAM分野のシステム開発，PDM(Product Data Management)，オープンシステムの開発に従事．情報処理学会，精密工学会，ACM，IEEE会員．