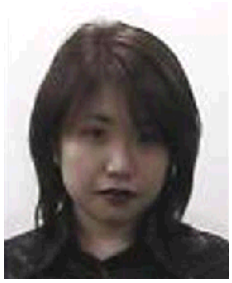
XML-RPC を利用したアプリケーション開発

富士ソフトＡＢＣ 株式会社

■ 執筆者Ｐｒｏｆｉｌｅ ■

1995 年　　富士ソフトＡＢＣ株式会社 入社
2003 年 6 月 現在
IT 事業本部オープンシステム部
第１技術グループ 課長

高　野　　啓



1997 年　　富士ソフトＡＢＣ株式会社 入社
2003 年 6 月 現在
IT 事業本部オープンシステム部
第１技術グループ 主任

入　江　内　弥

■ 論文要旨 ■

インターネットの普及に伴い，各種業務アプリケーションにおいて従来 LAN 内に閉じていた分散オブジェクトを，インターネットに対応できるよう機能拡張したいという要求が増えている．

こうした状況の中，ビルや工場の設備管理を行うシステムにおいて，従来サーバと同一 LAN 上に接続されていたクライアントを，インターネット経由で使用できるようにするため，プロトタイプの開発を行った．

この開発では，通信パフォーマンスの向上やプラットフォームの拡張性，開発コストを抑えることなどが要求された．通信プロトコルとして XML-RPC を使用することにより，これらの要求を満たすことができ，XML-RPC の有効性を実証することができた．この実績を生かして，今後さらに応用の範囲を広げることができると思われる．

今後の課題としては，プラットフォームによる違いの検証，セキュリティなどが挙げられる．

■ 論文目次 ■

１．はじめに ・・《 4》

２．システム概要 ・・《 4》

２．１　システムの概要

２．２　現行システムのソフトウェア構成

２．３　システム要件

３．実装方式の検討 ・・《 6》

３．１　方式検討

３．２　SOAP の概要

３．３　XML-RPC の概要

３．４　SOAP と XML-RPC の比較

４．XML-RPC による実装 ・・《 10》

４．１　ソフトウェア構成

４．２　クライアント側の実装

４．３　サーバ側の実装

５．むすび ・・・《 12》

■ 図表目次 ■

図１　現行システム概要 ・・《 4》

図２　現行システム ソフトウェア構成 ・・《 5》

図３　SOAP メッセージの構成 ・・・《 7》

図４　XML-RPC メッセージの構成 ・・・《 8》

図５　XML-RPC リクエストメッセージの構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・《 8》

図６　XML-RPC レスポンスメッセージの構成 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・《 9》

図７　ソフトウェア構成 ・・《 10》

図８　クライアント側ソフトウェア構成 ・・・《 11》

図９　サーバ側ソフトウェア構成 ・・・《 12》

１．はじめに

当社は 1970 年の創立以来，独立系ソフトハウスとして，特定分野に偏らないソフトウェアの受託開発を行ってきた．

分散オブジェクトを利用した通信アプリケーションの開発も数多く手がけてきており，最近はインターネットに対応した開発が増えてきた．

今回，従来 LAN 上でクライアント／サーバ型で稼動していた設備管理システムをインターネットに対応するため，プロトタイプの開発を行った．

２．システム概要

２．１　システムの概要

本システムは，工場やビルの空調などの設備の制御・監視を行う設備管理システムであり，以下の機能を持つ．

（１）サーバ上のデータベースに設備機器の状態を保持する．

（２）設備機器から，データベースの設備機器の状態をリアルタイムに更新する．

（３）クライアントからデータベースの設備機器の状態をリアルタイムに照会し，表示する．

現行システムの概要を**図１**に示す．

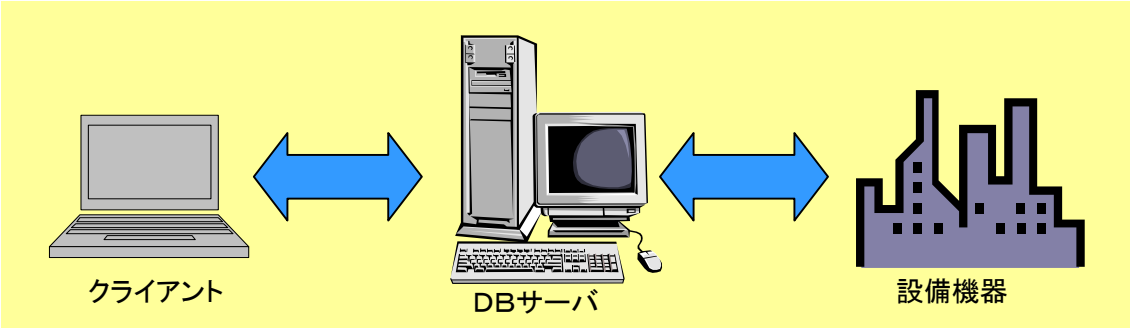


図１　現行システム概要

２．２　現行システムのソフトウェア構成

現行システムのソフトウェア構成は以下のとおりである．今回インターネットに対応するのはクライアント・DB サーバ間の通信であるため，クライアント・DB サーバを中心に記述する．

（１）データベースは Windows 2000 上の SQL Server を利用する．

（２）データベースとクライアントは，ローカルの同一 LAN 上にある．

（３）クライアントの表示部はグラフィックソフト Microsoft Visio 2000 を使用する．

（４）クライアント上の Windows アプリケーションとして作成された監視アプリケーションが DB サーバにポーリングを行って機器の状態を取得し，Visio の COM インターフェースを利用して表示を更新する．

（５）監視アプリケーションからデータベースへの接続は ADO を使用する．

現行システムのソフトウェア構成を**図２**に示す.

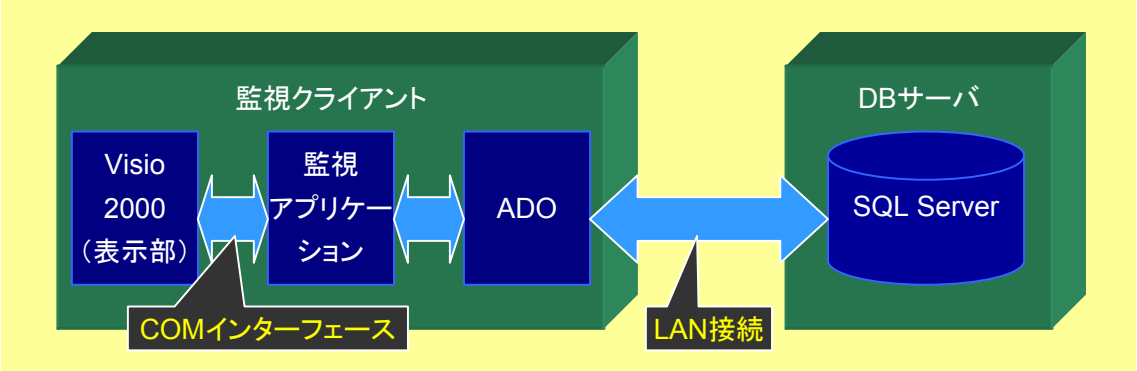


図２　現行システム ソフトウェア構成

２．３　システム要件

クライアントをインターネット対応にするための要件として，次のような項目がある．なお，DB サーバと設備機器間の通信については開発の対象外のため，本論文では言及しない．

（１）インターネット経由

クライアントからインターネット経由でサーバへの接続を可能とし，ファイアウォールやプロキシに特別な対応をしなくても通過できるようにする．

（２）通信パフォーマンス

リアルタイム性を重視するため，クライアント・サーバ間の通信を高速にし，レスポンスタイムを最小にする．

（３）プラットフォーム拡張性

将来，サーバ，クライアントとも Windows 以外のプラットフォームに容易に対応できるようにする．

（４）クライアントにインストール不要なシステム

現行システムでは，クライアントに Visio ファイルおよび監視アプリケーションをインストールする必要があった．新システムでは，クライアントに Web ブラウザと表示ソフトがインストールされていれば，本アプリケーションをインストールしなくてもよいようにする．

（５）開発コスト

通信プロトコルはなるべく単純なものを利用し，保守性，拡張性を犠牲にしないで開発にかかるコストを抑える．

本論文では，これらの要件を実現するための環境及び実装技法について，検討した内容を記述する．

３．実装方式の検討

３．１　方式検討

クライアント・サーバ間の通信を行うための実装方式として以下の候補を挙げた.

・CORBA
・DCOM
・SOAP
・XML-RPC

本システムの要件をふまえ，それぞれの方式について検討を行った.

（１）CORBA

CORBA は分散オブジェクトの通信を行うために必要な機能を備えた，特定のベンダーに依存しない規格である．多くのベンダーによってサポートされており，Java，C++ をはじめとした多くの言語から利用可能であるためプラットフォームの拡張性は高い．しかし，インターネットが普及する以前に規定された方式であるため，通信ポートの制限などによりファイアウォールやプロキシの通過が難しく，インターネットへの対応が困難である．また仕様および実装方法が複雑であるため，開発者が習得・実装するのに時間がかかり，開発期間・開発コストが増加してしまう.

（２）DCOM

DCOM も分散オブジェクトの通信を行うために必要な機能を備えているが，CORBA と同様の理由でインターネットへの対応が困難である．また Microsoft 独自の方式であるため Windows 以外のプラットフォームへの拡張性がない.

（３）SOAP

SOAP は，CORBA や DCOM と違い通信内容の記述に XML を用いる点が特徴である．SOAP の仕様はウェブ標準化団体 World Wide Web Consortium（W3C）で策定されている．言語やプラットフォームに依存しない汎用的なプロトコルであり，多くの実行環境や言語から利用可能である.

また SOAP はデータ構造のみが規定されており，HTTP や SMTP など既存の通信プロトコルの上で通信を行う．そのため，HTTP と組み合わせることにより，ファイアウォールを通過することができる.

（４）XML-RPC

XML-RPC は SOAP と同様，通信内容の記述に XML を用い，言語やプラットフォームに依存しない汎用的なプロトコルであり，多くの言語から利用可能である．HTTP と組み合わせることにより，ファイアウォールを通過することができる.

以上により，「インターネット経由」，「プラットフォーム拡張性」の観点から，実装方式として SOAP または XML-RPC が適切であると考えられる.

３．２　SOAP の概要

SOAP は，異なるシステム間で XML 文書をやり取りするために開発されたプロトコルであり，ウェブ標準化団体 World Wide Web Consortium（W3C）により宛先などの記述方法が規定されている．SOAP は元来 Simple Object Access Protocol の略であったが，現在では何かの略語ではなく，固有名詞であることが W3C により宣言されている．

SOAP は拡張マークアップ言語（XML）や Web サービス記述言語（WSDL）とともに，普及しつつある Web サービスの基盤となる技術である．

SOAP メッセージは「SOAP エンベロープ」「SOAP ヘッダ」「SOAP 本体」の 3 つの領域に分かれている．エンベロープにはエンコーディングが宣言され，ヘッダに宛先などが格納され，ボディに通信内容が記述される．SOAP メッセージの構造を**図３**に示す．

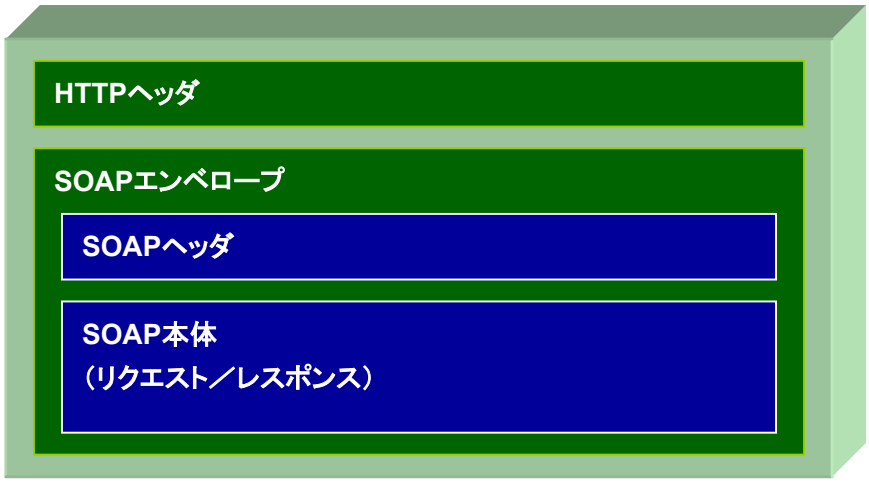


図３　SOAP メッセージの構成

３．３　XML-RPC の概要

RPC(リモート・プロシージャ・コール)は，従来からあるクライアント・サーバ型の通信方式である．XML-RPC は，Dave Winer により，XML と HTTP を使って RPC の機能が拡張されたものである．

XML-RPC の元となっている RPC では，クライアントマシンのプログラム内から呼び出されるプロシージャが，ネットワーク経由で RPC サーバに送られる．このプロシージャは RPC サーバ上で実行され，RPC サーバは実行結果をまとめ，その結果を呼び出し側に送り返す．以前は処理能力の低いクライアントが処理能力の高いマシン上のリソースを利用するのが主な利用目的であった．

XML-RPC では，クライアントが RPC 要求（リクエスト）を XML にエンコードし，それを HTTP を介してサーバに送信する．サーバは XML をデコードし，要求されたプロシージャを実行して，その結果（レスポンス）を XML にエンコードし，HTTP を介してクライアントに送り返す．クライアントは，XML をデコードし，結果を必要な使用言語のデータ型に変換して実行を続ける．

XML-RPC で送信できるのはメソッド名とパラメータのみであり，SOAP のように任意の XML 文書を送信することはできない．このため XML-RPC メッセージは SOAP メッセージに比べてシンプルで，ヘッダなどがなく本体のみである．XML-RPC の仕様自体も非常にシンプ

ルで，サンプルを含んだ仕様書全体で 10 ページに満たない．

XML-RPC メッセージの構成を**図４**に，リクエストメッセージの構成を**図５**に，レスポンスメッセージの構成を**図６**に示す．

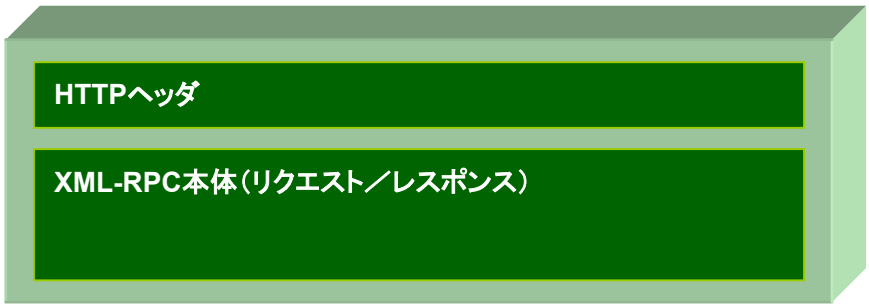


図４　XML-RPC メッセージの構成

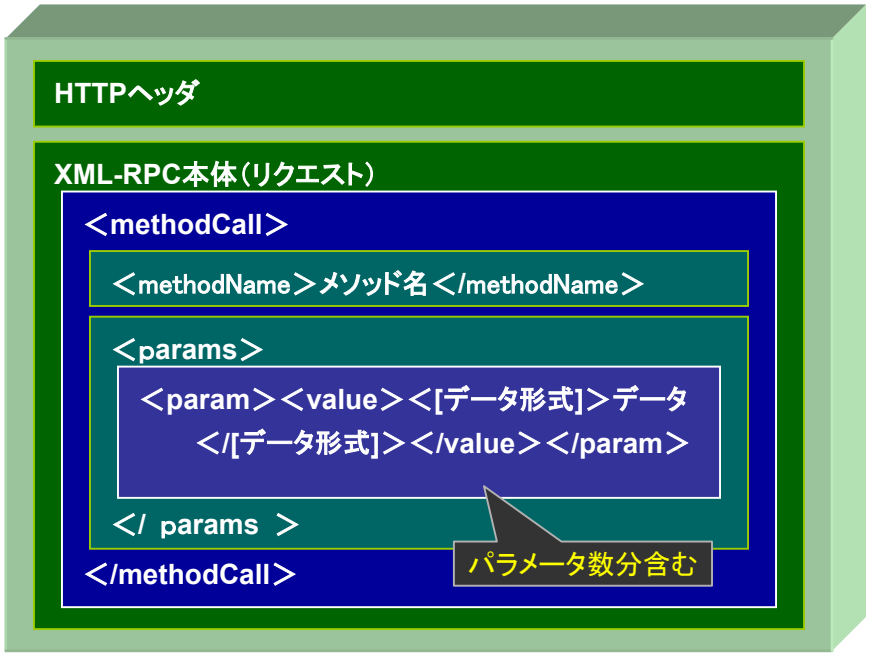


図５　XML-RPC リクエストメッセージの構成

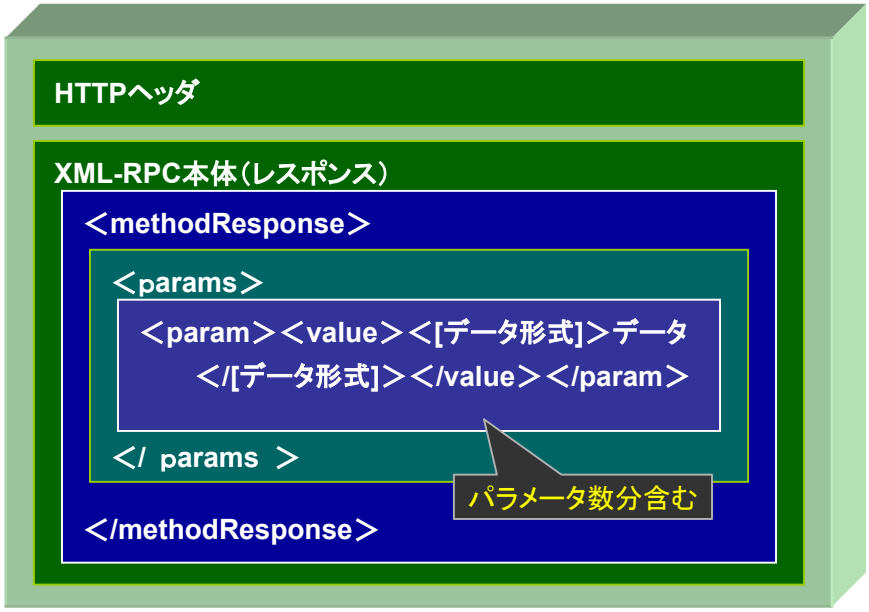


図６　XML-RPC レスポンスメッセージの構成

３．４　SOAP と XML-RPC の比較

次に，本システムの要件を考慮し，以下の観点から SOAP と XML-RPC を比較する．

・バージョンアップの可能性
・開発コスト
・通信パフォーマンス

（１）バージョンアップの可能性

SOAP の現在のバージョンである SOAP 1.1 にはいくつかの問題点があり，今後 SOAP 1.2 へのバージョンアップが予定されている．このように SOAP は今後バージョンアップの可能性がある．

一方，XML-RPC は仕様がシンプルであるため今後バージョンアップの予定はない．

（２）開発コスト

SOAP は多くの機能を含んでいるため仕様が複雑である．一方，XML-RPC は SOAP に比べて機能が少なく仕様がシンプルなため，短期間で習得でき，開発やテストにかかる工数を削減することができる．XML-RPC では複雑なデータを扱うことができないが，本システムで送受信するデータを扱うには，XML-RPC の仕様の範囲内で十分可能である．

（３）通信パフォーマンス

XML-RPC メッセージは SOAP メッセージに比べてヘッダなどがないため，通信量を少なくすることができる．このため通信パフォーマンスが向上させることができる．

以上の検討により，本システムでは，SOAP より XML-RPC のほうが適していると思われるが，当社での XML-RPC の使用実績が少なかったため，プロトタイプの開発を行い XML-RPC の有効性を実証することとした．

４．XML-RPC による実装

４．１　ソフトウェア構成

XML-RPC を利用することを前提として，クライアント側／サーバ側の実装方法について検討する．

クライアント側のソフトウェアは，以下の構成で検証した．

OS	: Windows98/2000
Web ブラウザ	: Microsoft Internet Explorer 5.5
グラフィックソフト	: Microsoft Visio 2000

サーバ側のソフトウェアは，以下の構成で検証した．

OS	: Windows 2000 Server
Web サーバ	: Apache 1.3.20
アプリケーションサーバ	: Tomcat 3.2.4
DB サーバ	: Microsoft SQL Server 2000
開発言語	: J2EE，Java サーブレット

検証したソフトウェア構成を**図７**に示す．

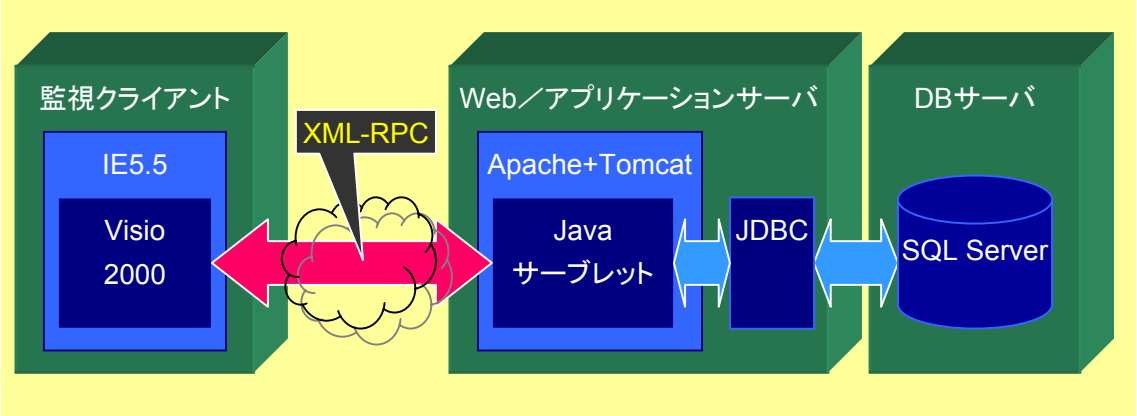


図７　ソフトウェア構成

４．２　クライアント側の実装

クライアントの実装方法について検討する．

（１）インストール不要のシステム

最初に，クライアントへのアプリケーションのインストールを不要とするための実装方式について検討する．

現行システムでは表示部をグラフィックソフト Visio 2000 で実装している．この処理を Web ブラウザで置き換えた場合，表現力や操作性の点で格段に劣り，機能要件を

満たすことができないため，Visio 2000 をそのまま使うこととした．また本システムは不特定多数のユーザが使用するものではないので，動作環境として Web ブラウザ（Internet Explorer）と Visio 2000 がインストールされていることを前提とした．

実現方式として Web サーバ上に Visio ファイルを置き，Microsoft Visio 2000 がインストールされたクライアント上の Web ブラウザ（Internet Explorer）からこの Visio ファイルを指定（リンク）することにより，Visio ファイルがダウンロードされ Visio 2000 が起動されて Internet Explorer 上で開くことができる．

これにより，アプリケーションに変更があった場合もサーバ側だけの修正で済み，クライアントへのインストールは不要となる．

この方式では，アプリケーションを Visio 上で実行するため，Visio のマクロ（VBA）を使用する必要がある．

（２）VBA での XML-RPC の実装

VBA から標準で使える XML-RPC ライブラリが用意されていないため，XML-RPC の実装は，すべて VBA で作り込むこととした．この際，XML-RPC メッセージ・VBA 変数間の変換用にパーサ／レンダラが必要となる．そこで，XML-RPC のメッセージ構造がシンプルであるうえ，今回のアプリケーションでは複雑なデータ構造を使用しないので，XML の特定のタグだけに対応した本アプリケーション専用の軽量パーサ／レンダラを開発することとした．

データの送受信は VBA から Windows のインターネットライブラリ Wininet を呼び出すこととした．

Viso の VBA プログラムに関しては，ユーザによる書き換えを防ぐため，パスワードによる保護を行うこととした．

クライアント側のソフトウェア構成を**図８**に示す．

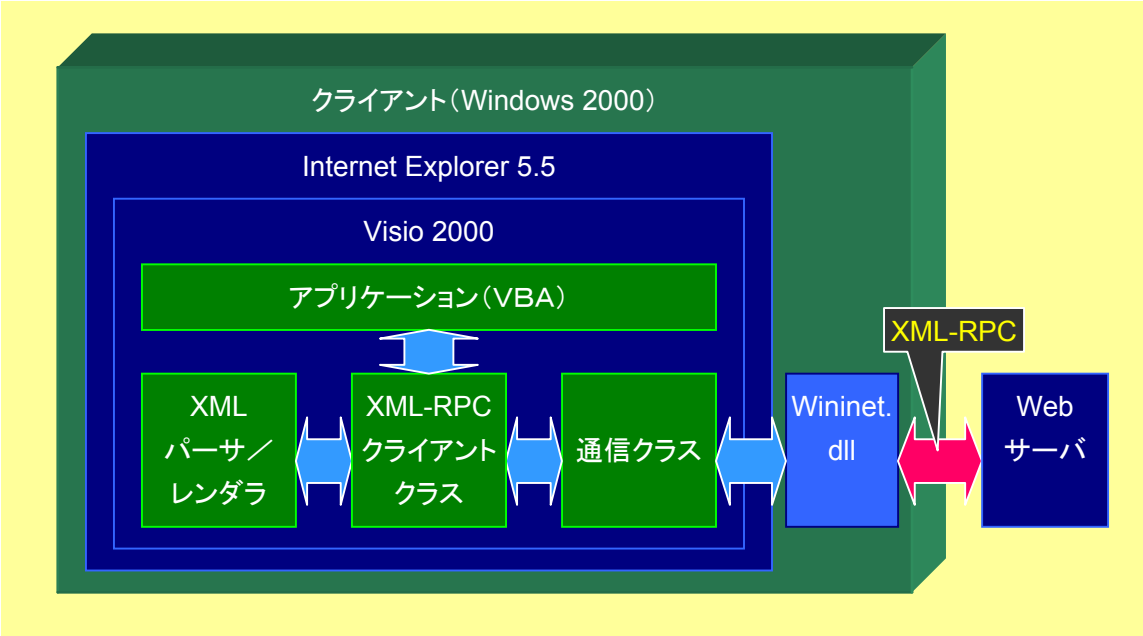


図８　クライアント側ソフトウェア構成

４．３　サーバ側の実装

サーバ側は，インターネットに対応するため Web サーバとアプリケーションサーバとして Apache と Tomcat を追加した．

開発言語は Web システムの開発で一般的に使われている Java とした．Java からの XML-RPC の実装は，Apache XML-RPC パッケージを利用した．このパッケージにより Java 変数・XML-RPC 間の変換およびメソッド呼び出しを行う．

クライアントから受信したリクエストをリクエスト受信サーブレットが受け取り，XML-RPC パッケージを経由して，該当する各処理クラス（メッソッド）が呼び出される．各メッソッドは JDBC によりデータベースアクセスを行い処理を行う．実行結果（レスポンス）は XML-RPC パッケージ，リクエスト受信サーブレットを経由してクライアントに送信される．

このうち開発対象は，リクエスト受信サーブレットと処理クラス（メッソッド）の部分である．

サーバ側のソフトウェア構成を**図９**に示す．

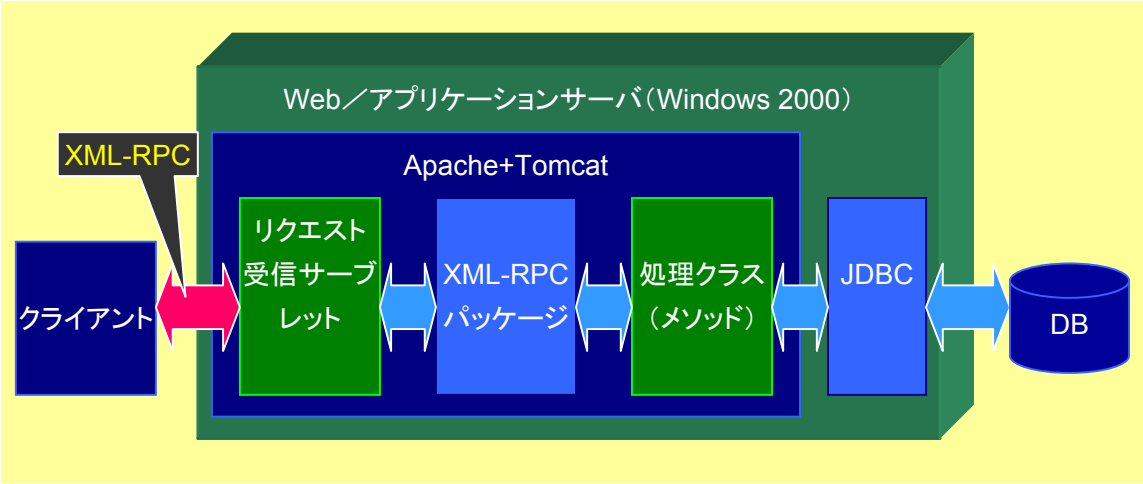


図９　サーバ側ソフトウェア構成

５．むすび

クライアント／サーバ型のシステムをインターネットに対応させるため，本システムでは XML-RPC を採用し，その有効性を実証することができた．XML-RPC を使用することによって要件を実現した方法は，以下のとおりである．

（１）インターネット経由

ファイアウォールやプロキシを通過できるプロトコルとして，http 上で動作する XML-RPC を採用した．

サーバ側は Java で開発し，XML-RPC の実装として Apache XML-RPC を利用した．

クライアント側は VBA で開発したため，XML-RPC の実装はすべて作り込みとなったが，XML-RPC の仕様がシンプルであったため，特に問題なく開発することができた．データの送受信は Windows のインターネットライブラリ Wininet を利用した．

（2）通信パフォーマンス

XML-RPC メッセージは，構造が単純でヘッダ情報などがなく，必要最低限のデータの送受信のみを行うため，通信量を少なくすることができた．この結果通信レスポンスは要件の範囲内に抑えることができた．

（3）プラットフォーム拡張性

XML-RPC は，特定のベンダーに依存しない標準仕様であるため，C や Java など多くの言語，プラットフォームでサポートされている．VBA のように標準で XML-RPC がサポートされていないクライアント環境でも，独自の軽量パーサ／レンダラを作り込むことにより，比較的簡単に実装できることが実証できた．

（4）クライアントにインストール不要なシステム

本システムでは，クライアントの動作環境として Web ブラウザ（Internet Explorer）と Visio 2000 がインストールされていることを前提とした．

Web ブラウザから，サーバ側に置いてある Visio ファイルをダウンロードし Internet Explorer 上で Visio 2000 を開く方式とした．これにより，アプリケーションに変更があった場合もサーバ側だけの修正で済み，クライアントへのインストールは不要とすることができた．

（5）開発コスト

XML-RPC は多くの開発環境や言語でサポートされている．VBA ではサポートされていなかったが，XML-RPC は仕様がシンプルなため，短期間で作り込むことができた．結果として開発コスト抑制しを予算内に収めることができた．

今後の検討課題として，以下の項目が挙げられる．

（1）プラットフォームによる違い

今回は，Visio 2000 がインストールされた Windows 上の Microsoft Internet Explorer での動作確認を行ったが，Windows のバージョンや Web ブラウザのバージョンによる違いについて細かな検証を行っていない．今後はこのような点についても検証していく必要がある．また，システムの拡張性について，Macintosh など Windows 以外のプラットフォームでの実装方式を含めた検討・実証が必要である．

（2）セキュリティの考慮

本システムでは，設備管理という安全にかかわる情報を扱う．今回はプロトタイプのため通信データのセキュリティ対応を行わなかったが，情報の漏洩および改ざんを防ぐという観点から，サーバ・クライアント間の通信データについて，SSL の実装などセキュリティを考慮する必要がある．

また，認証により，利用できるユーザを限定することも必要である．

（3）アプリケーションのバージョン整合性

クライアントにダウンロードされた Visio ファイルをクライアント側で直接実行した場合，旧バージョンのアプリケーションが実行されサーバ側のバージョンと合わない可能性がある．このための対策として，メッセージ中にバージョン情報を追加し毎回バージョンを確認する，あるいは起動時にバージョンの確認を行うなどの対応が必要である．

XML-RPC に関してインターネット上の検索エンジンで検索すると，ヒットする日本語のサイトは非常に少ない．このことから，XML-RPC は日本ではあまり普及していないことがわかる．今回の開発では，実装が簡単で軽量な XML-RPC の有効性を実証することができた．今後さらに応用の範囲を広げることができると思われる．