

.NET C

.NET C #

小出 淳平, 渡邊 真也

Jumpei Koide, Shinya Watanabe

Abstract: This paper introduces .NET and C #. .NET is Microsoft XML Web service platform. The .NET platform is a set of development tools to build, expose, and consume XML Web services. It enables a integrated Web delivered through smart devices. C # is object-oriented language that enables programmers to quickly build a wide range of applications for the new Microsoft .NET platform.

1 はじめに

Microsoft 社は, OS とアプリケーションのライセンス販売で利益を生み出すビジネスモデルから, サービス中心へと転換させるために「Microsoft.NET」という新戦略を発表した²⁾.そして,.NET に対応するための新しい言語として C # という言語を作り出した.この言語は, Java に非常に良く似ている. SUN と Java をめぐって係争中にあるなか, このような言語を開発したのは, SUN を強く意識しているためと思われる.一方, SUN も, .NET に対抗して「SUN ONE」²⁾ という Web サービスを発表している.

本発表では, Microsoft が強く押し進めている「.NET」と「C #」についての概要と今後の可能性について述べる.

2 .Net

2.1 .NET とは

Microsoft は 2000 年 6 月, 新戦略「Microsoft.NET」を発表した²⁾. .NET の Web サービスは, 従来のように特定の OS をプラットフォームとする技術ではなく, インターネットそのものをプラットフォームとする技術である.ただし, Web サービスの開発と運用には OS として Windows を用いることが最も適したものとなると考えられる.

.NET では, SOAP (Simple Object Access Protocol) という HTTP ベースのプロトコルで Web サービスを呼び出す. SOAP とは, インターネットで接続された様々なコンピュータに配置された Web サービスを, XML メッセージの交換によって連携させるプロトコルを規定する仕様である.

XML 文書では, 情報が構造化されて記述されているため, これをソフトウェアなどで読み取り, 内容を把握することが容易である.つまり Web サーバが適切なインターフェイスさえ備えてくれれば, ソフトウェアがこ

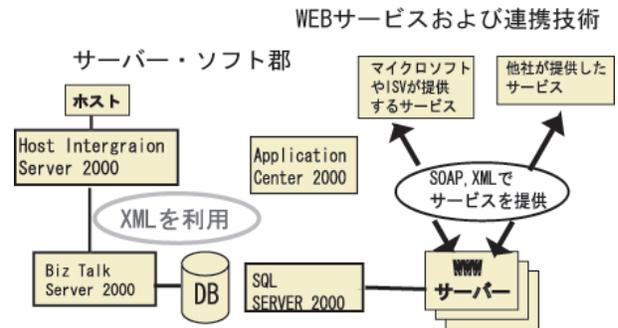


Fig. 1 .NET の仕組み

れにアクセスし, 必要な情報を自動で取り出せるようになる.

.NET プラットフォームを構成している主要な構成要素は, 4 つある. それらは (1) サーバ・ソフト郡 (2) Web サービス (3) 開発環境とツール (4) Windows 2000 とその後継 OS である. その内 (1) と (2) の 2 つについて Fig. 1 に示す. サーバ・ソフト郡では, サーバと DB 間で XML を利用して連携している.そして, Web サービスでは, SOAP, XML でサービスを連携している.

現在, 一般的な Web 情報サービスは, クライアントである Web ブラウザと Web サーバとの間で HTTP と呼ばれるリモート・ファイル転送プロトコルを使用し, この手順に従って HTML と呼ばれるフォーマットのデータを交換することで実現されている. .NET では, HTTP と HTML の部分が SOAP と XML に置き換わっている. SOAP / XML インターフェイスを Web サーバが備えてくれれば, 人間の手を煩わせることなく, 容易にソフトウェアが能動的にインターネット上にあるこれらの Web サーバを閲覧し, 必要な情報を取り出し, 分析し, アクションを起こせるようになる.

以下に .NET の利点と欠点を示す。

利点

- ・ CLR¹は、様々なプログラミング言語に対して共通のランゲージランタイムを提供する点
- ・ 様々なプラットフォームに対応している点

欠点

- ・ SOAP や XML が広まらないと、.NET が実現できない点
- ・ まだ、規格が十分定まっていない点
- ・ 開発、運用は基本的に Windows に限定

3 C # (シャープ)

3.1 C # (シャープ) とは

C # は、C および C++ を元にしたシンプルで、現代的なオブジェクト指向のプログラミング言語である。そして、C # における目標は、Visual Basic の高い生産性と、C++ の優れた能力を兼ね備えることである。

しかし、その実態は、Java を非常に意識したものとなっている。実際、C # の使用や使用環境を知れば知るほど、Java に似ていることが分かる。例えば、Java と同様ポインタの概念がないという点やガーベジコレクションがある点などが挙げられる。つまり、基本的に Java で実現できるものは、C # でも実現できるということである。さらに、Microsoft は、JUMP to .NET (Java User Migration Path to Microsoft .NET) という Java アプリケーションを .NET Platform へ移植するためのソフトウェア開発ツールとサービスを 2001 年 1 月 25 日に発表している²⁾。これは、J++ で開発した既存のアプリケーションを変換し、.NET Platform で動作可能にしたり、Java ソースコードを C # に自動変換したりするツールである。

Microsoft は、SUN と Java を巡って係争中であったが、.NET 構想の発表の二日前に和解したばかりだった。つまり、Java で作られているものをすべて C # に置き換えてしまい、現在の Java の地位を奪うつもりだと考えられる。

3.2 利点と欠点

以下に C # の利点と欠点を示す。

利点

- ・ 生産性の向上
- ・ ポインタの廃止とガーベジ・コレクションでプログラムの間違い防止

欠点

- ・ 規格が十分決まっていない点

¹様々な言語において Java のように中間コードを取り、OS やプログラミング言語に依存しない環境

4 性能比較

本発表では、Java と C # についての簡単なベンチマーク比較を行った。測定環境としては、Java では「java version (1.3.0Q2) Java HotSpot (TM) Server VM」と「java version(1.3.0) Classic VM(IBM)」の 2 種類を用いた。そして、C # として「.NET Framework SDK 1」を用いた。実行するプログラムは「sum +=i, sum -=i」を for ループで 5 億回するというものである。

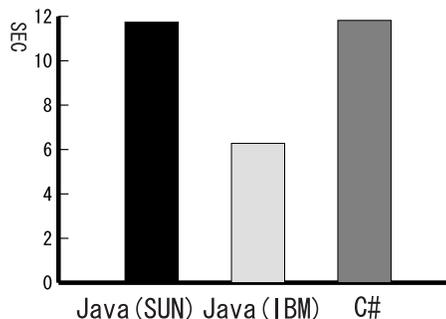


Fig. 2 Java と C # の性能比較

数値結果を Fig. 2 に示す。Fig. 2 を見ると分かるように、C # は Java (SUN) の結果とはほとんど差が見られなかったが、Java (IBM) と比較すると倍の計測時間を必要としていることが分かる。C # の性能は、Java や VB よりも速いと言われているが、今実験では Java (IBM) の方が良い結果を示していた。

5 終わりに

.NET を実現する上でコアとなる技術は、Web サービスを連携する SOAP である。しかし、SOAP は、仕様は公開されているものの Web サービスを検索するための仕組みやセキュリティーなどが決まるのはこれからであるため、ビジネスモデルもまだ見えていないのが問題である。そして、今回の数値結果では、C # の性能は Java とほぼ同様、もしくは劣っていた。よって、C # は今後、更なる性能向上や IE における完全移行などを実現しなければ、Java の勢力に阻まれ世間へ浸透しないものと思われる。しかし、.NET 自体はコモンランゲージランタイムや新しい Web サービスの実現性などを秘めており、次世代インターネットツールとして最も期待されている。ただし、今年発表されたばかりの構想であるので、人々の注目を集めている間に社会に浸透させる必要があるだろう。

参考文献

- 1) 片岡 巖 『C # で学ぶ .NET プログラミング』(株式会社技術評論社、2001)
- 2) IT PRO <http://itpro.nikkeibp.co.jp/>