XML Web サービスのためのコールバック用ファイルサーバ

阿部　聡†　新城　靖††　板野　肯三††

† 筑波大学大学院理工学研究科理工学専攻　　†† 筑波大学電子・情報工学系

1　はじめに

近年、XML Web サービスと呼ばれる技術が注目を集めはじめている。XML Web サービスとは、Web サービス（Web of service）とも呼ばれ、データを交換するときに XML 形式の文書を用いるようなソフトウェア・コンポーネントである。ここでは、サービスを提供するプログラムをサーバ、利用するプログラムをクライアントと呼ぶことにする。

ローカルプログラムと XML Web サービスのサーバとの連携ができれば便利である。そのために、ローカルプログラムが作成したローカルファイルをサーバが操作できるようにする必要がある。従来の XML Web サービスではローカルファイルの入出力をクライアントで行っている。サーバは、クライアントからデータをメソッドの引数で受け取っている。しかし、この方法ではローカルファイルのデータをサーバに送るコードをクライアントごとに毎回作らなければならない。データのごく一部が必要な場合やランダムアクセスが必要な場合でもファイルの全データを送ることも多い。また、別のサーバに処理を任せる場合も、そのサーバにファイルの全データを送らなければならないこともある。

このような問題を解決するために、本研究では XML Web サービスのためのコールバック用ファイルサーバを実現する。コールバック用ファイルサーバとは、サーバがクライアント側にあるファイルにアクセスするためのプログラムである。これにより、ローカルプログラムとの連携が可能なサーバを作ることが容易になる。

2　コールバック用ファイルサーバ

図１に、サーバが本研究で実現するコールバック用ファイルサーバを経由し、ローカルファイルをアクセスしている様子を示す。コールバック用ファイルサーバは、クライアントと同じホストで動作する。

コールバック用ファイルサーバはサーバ側とクライアント側の２つのインタフェースを提供している。サーバ側のインタフェースは普通のファイルサーバと同じように `read` や `write` などのメソッドを提供する。このサーバからアクセス可能なファイル、および、ディレクトリをそれぞれ**グローバルファイル**、および、**グローバルディレクトリ**と呼ぶことにする。一方、クライアント側のインタフェースはローカルファイルとグローバルファイルの対応関係を設定するためのものである。例えば、アクセス権の設定や名前変換の設定を行うことができる。

ローカルファイルを遠隔のサーバにアクセスさせたい場合、クライアントがクライアント側のインタフェースを通じてローカルファイルの名前を渡し、WSDL（XML Web サービスのインタフェースの記述）を手に入れる。そして、その WSDL をサーバに渡す。サーバはその WSDL をもとにサーバ側のインタフェースを通じて、コールバック用ファイルサーバに要求を送る。

2．1　サーバ側インタフェース

本研究では、Java の既存のクラスおよび NFS(Network File System)[1]を参考にしてコールバック用ファイルサーバのサーバ側インタフェースを定義した（表１）。Java を参考にした理由として、Java のオブジェクトが比較的

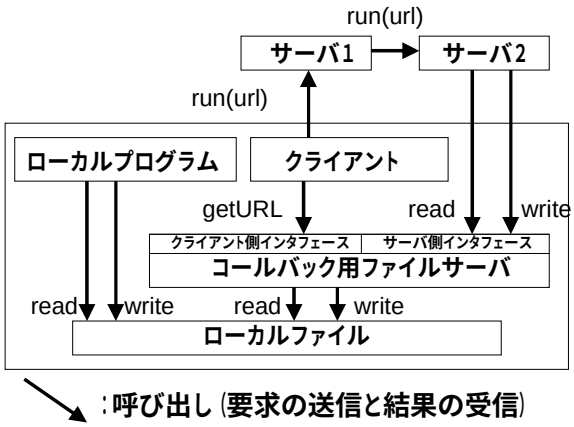

図１：コールバック用ファイルサーバによるファイルへのアクセス

近代的である点と、Java が XML Web サービスを開発するための言語としてよく使われているので開発者が XML Web サービスにおけるサーバプログラムを作りやすいという点があげられる。

サーバ側のインタフェースを定義するにあたり、Java にある既存のメソッドから残したものと削ったものがある。さらに、残したメソッドの中で、インタフェースを変更したものとしてないものがある。残したメソッドの中でインタフェースを変更していないメソッドとして `write`、および `close` などがある。`read` メソッドは以下のようにインタフェースを変更した。

Java：`int read(byte[] s)`
本研究：`ReadResult read(long size)`

`ReadResult` は構造体を表すクラスで、`long` 型と `byte[]`型で構成されている。インタフェースを変えた理由は、従来のままではクライアントからサーバにデータを送るという意味になるからである。

`IGlobalDirectory` のインタフェースを定義するために参考にしたシステムとして NFS がある。NFS では、ファイルやディレクトリへの識別子として 32 バイトのファイルハンドラと呼ばれるデータを返しているが、本研究では WSDL（厳密には WSDL を HTTP 経由で入手するための URL）を返している。

2．2　クライアント側インタフェース

クライアント側インタフェースの主なメソッドを以下に示す。

- `GlobalDirectory.add()`
- `Global{File,Directory}.copyToLocal()`

`add` メソッドはグローバルファイルの名前とオブジェクト（`GlobalFile` または `GlobalDirectory`）との束縛を作成する。これにより、サーバ側が見えるファイル

表１：サーバ側インタフェース

インタフェース名	定義の参考にしたもの
`IGlobalFile`	`File` クラス
`IGlobalRandomFile`	`RandomAccessFile` クラス
`IGlobalInputStream`	`FileInputStream` クラス
`IGlobalOutputStream`	`FileOutputStream` クラス
`IGlobalDirectory`	NFS

の名前とローカルファイルの名前を別のものにして、ローカルファイルの名前をサーバに対して隠すことができるようになる。

copyToLocal メソッドはサーバが作成したグローバルファイルをローカルファイルにコピーするためのメソッドである。サーバが作成したファイルは、ローカルファイルに直接書くのではなく、まず一時ファイルに書かれる。その理由は、悪意を持ったサーバにより攻撃されることを防ぐためである。

3 コールバック用ファイルサーバの利用

以下に、ファイルを読み込んで分割を行い、ファイルに出力するプログラムを例として本コールバック用ファイルサーバの利用法を示す。

```
public boolean split(int num,
    String inputname,String dirurl){
    IGlobalDirectory dir = (
       IGlobalDirectory)Registry.bind(
          dirurl, IGlobalDirectory.class);
    String url = dir.lookup(inputname);
    InputStream fis = new StInputStream(url);
    BufferedReader r = new BufferedReader(new
       InputStreamReader(fis));
    int f=0;boolean moredata=true;
    while(moredata){
      file = inputname + f; f++;
      url=dir.makefile(file);
      OutputStream fos = new StOutputStream(url);
      BufferedWriter w = new BufferedWriter(new
          OutputStreamWriter(fos));
      moredata = filecopy(r,w,num)
    }
    return true;
}
```

このプログラムは dirurl で指定されたグローバルディレクトリにある inputname で指定された名前を持つグローバルファイルを num で指定された行で分割し、それを数字のサフィックスを持つグローバルファイルに書き込むプログラムである。

まず、Registry.bind()により、グローバルディレクトリをアクセスするためのスタブを生成している。Registry.bind()は Glue[3]に含まれているメソッドである。Glue とは、XML Web サービスのためのプログラムを開発するためのフレームワークである。

次に、グローバルディレクトリの lookup メソッドを使って入力となるグローバルファイルの WSDL（URL）を得ている。StInputStream は、IGlobalInputStream を使いやすくするための Java のスタブである。コンストラクタの引数は WSDL（URL）である。そのグローバルファイルを行ごとに入力するために Java 標準クラスライブラリに含まれる BufferedReader クラスのオブジェクトを作っている。

ループの中では、まずグローバルディレクトリの makefile メソッドを呼び出している。引数はグローバルファイルの名前である。このメソッドを呼び出すことにより、空のグローバルファイルを作成し、そのときに得た WSDL(URL)を引数として StOutputStream クラスのオブジェクトを作成している。StOutputStream は、IGlobalOutputStream を使いやすくするためのスタブである。そして、行単位でグローバルファイルに書き込むために BufferedWriter クラス（Java 標準）のオブジェクトを作る。次に、filecopy()を呼び出している。これは、num 行分だけグローバルファイルに書き込む。ファイルの読み込みが終わったら false を返す。

次に、クライアントの概略を以下に示す。

```
  public static void main( String[] args ) {
    GlobalFileServer gfs =
       new GlobalFileServer("40000");
    FileAttribute fa = new FileAttribute();
    ISplit so = (ISplit)Registry.bind(
       splitserverurl, ISplit.class );
    GlobalDirectory gd =
       new GlobalDirectory(fa,gfs);
    GlobalFile gf=
       new GlobalFile(fa,args[1],gfs);
    gd.add("file-",gf.getURL(),"file",gf);
    so.split(args[0],"file-",gd.getURL());
    gd.copyToLocal(args[2],"file-",args[1]);
    System.exit(1);
  }
```

このプログラムの引数は分割する行数、入力ローカルファイルの名前、および、出力すべきディレクトリである。

まず、GlobalFileServer クラスのオブジェクトを作成することで、本コールバックファイルサーバを起動している。引数はポート番号である。次に、Glue の持つ Registry.bind()を使って上で述べたサーバをアクセスするためのスタブを生成している。そして、GlobalFile,GlobalDirectory クラスのオブジェクトを作成している。これらのクラスのコンストラクタの引数としてファイルの属性をあらわすオブジェクト fa と GlobalFileServer クラスのオブジェクトがある。fa はパーミッションや最大ファイルサイズなどが格納されている。加えて、GlobalFile の引数にはローカルファイルの名前が入る。

その後、add メソッドを使うことによってそのグローバルファイルをグローバルディレクトリに登録している。WSDL(URL)を得るために getURL()といったメソッドを使っている。

サーバプログラムのメソッド split を呼び出してから、最後に copyToLocal メソッドにより作成されたグローバルファイルをローカルファイルにコピーしている。

4 関連研究

XML Web サービスを使ったファイルサーバとして、XmethodsFilesystem[2]がある。XmethodsFileSystem は、ファイルの書き込みや読み込みなどを遠隔で行える XML Web サービスである。しかしながら、ローカルファイルをリモートの XML Web サービスのサーバに利用させることはできない。

5 まとめと課題

本研究では、XML Web サービスのためのコールバック用ファイルサーバを実現した。これによってローカルプログラムと連携が可能なサービスのプログラムを作ることが可能になる。現在までに表 1 に示したインタフェースを持つファイルサーバを実現した。今後の課題は、それを評価することである。

参考文献

[1] ”Network File System Protocol specification”, RFC1094(1990)

[2] XmethodsFileSystem,Xmethods(2003) http://www.xmethods.com/

[3] Graham Glass:Web Services Building Blocks for Distributed Systems, Prentice Hall(2001)