R からの DDBJ アクセスメモ

Ei-ji Nakama COM-ONE Ltd.

2007 年 7 月 20 日

1 環境設定

R [1] からウェブサービスを利用するパッケージは幾つか存在する[*1] が, 我々は RCurl [2],XML [3] パッケージを用いる SSOAP [4] を試した. [*2]

R の各パッケージのインストールは, CRAN(`http://cran.r-project.org`) [1], BioConductor(`http://www.bioconductor.org`) [5] から行う. BioConductor には,R の通常のパッケージの他に, annotation,experiment,omegahat,monograph があり, XML は CRAN,RCurl SSOAP は omegahat に含まれる.

1.1 管理者向け

基本的なインストールには,

最も簡単なインストール

```
> source("http://www.bioconductor.org/biocLite.R")
> biocLite()
```

を行えば良い.

[*1] RSOAP,SSOAP 等がある

[*2] RCurl は libcurl(`http://curl.haxx.se/`) を用い, XML は libxml(`http://xmlsoft.org/`) を用いる. 同等のプログラムを perl の SOAP::Lite で書かれた物と R の SSOAP を用いた物で比較したが,SSOAP の方が高速だった. perl の SOAP::Lite はバージョン 0.60a より新しい物ではうまく機能しなかった.SSOAP も不具合が見受けられたが SSOAP-0.4-1(bioconductor) では無く,SSOAP-0.4-2 を omegahat の web から直接とれば良い

より多くの機能を使うには全てインストールしてしまう事も出来る．[*3] 各種パッケージのインストールには, options()$repos の設定が必要だが.Rprofile に以下のように定義しておけば, install.packages で簡単にインストールが可能となる．[*4]

.Rprofile

```
options(repos=c(CRAN="http://prs.ism.ac.jp/",
                BIOC="http://prs.ism.ac.jp/bioc/1.9/bioc",
                data_annotation="http://prs.ism.ac.jp/bioc/1.9/data/annotation",
                data_experiment="http://prs.ism.ac.jp/bioc/1.9/data/experiment",
                omegahat="http://prs.ism.ac.jp/bioc/1.9/omegahat",
                monograph="http://prs.ism.ac.jp/bioc/1.9/monograph"))
```

全てのパッケージのインストール

```
> install.packages(new.packages())
```

1.2 ユーザ向け

ism のスーパコンピュータから DDBJ や KEGG 等の web サービスを利用するには, PROXY 経由でアクセスする必要がある．[*5] これは環境変数 http_proxy [*6] に http://prs.ism.ac.jp:3128 を設定する．バッチやフロントエンドには以下のように設定すれば良い[*7] ．

b シェル系

```
http_proxy=http://prs.ism.ac.jp:3128
export http_proxy
```

c シェル系

```
setenv http_proxy http://prs.ism.ac.jp:3128
```

[*3] 依存関係もあるので, 繰り返してインストールを行う必要がある．
[*4] ただし, 全てのパッケージのインストールには膨大な時間を要する．
[*5] 直接には外部へのポート TCP:80 のアクセスは行えない．
[*6] R の socket モジュールは http_proxy が HTTP_PROXY でも受け付けるが, RCurl パッケージが使う libcurl では小文字の http_proxy しか受け付けない．
[*7] ism のスーパーコンピュータ上では, /usr/local1/bin/env_local1.{sh,csh} で設定している．

2 使用例

DDBJ のチュートリアル (http://www.xml.nig.ac.jp/tutorial/index_jp.html) [6] に倣って, R 版を作成した.

2.1 どんなサービスを提供しているのか?

WSDL リスト (http://xml.nig.ac.jp/wsdl/index.jsp) [7] にアクセスする. WSDL ファイルの紹介があり, 実際にサービスを使用することができる.

2.2 どうやって Web service にアクセスするのか?

SOAP を使用するには, Perl や Java や R のようなプログラミング言語が必要である. Perl を使用する場合は SOAP::Lite, Java を使用する場合は Axis, R を使用する場合は SSOAP などが必要になる. これらを導入しさえすれば

1. WSDL ファイルの指定
2. メソッドの呼び出し

を行うだけで SOAP を使用できる.

例えば アクセッション No から DDBJ Entry を用いて DDBJ-XML フォーマットで結果を取得したい場合

1. GetEntry の WSDL の指定 (http://xml.nig.ac.jp/wsdl/GetEntry.wsdl)
2. getXML_DDBJEntry メソッドにアクセッション No を引数として実行

とする. 以下に R を用いた例を紹介する.

2.2.1 R を用いた場合

パラメータとメソッドの紹介ページへ http://www.xml.nig.ac.jp/doc/index.html

SOAP にアクセスする為には以下が必要.

- libxml (http://xmlsoft.org/)
- libcurl (http://curl.haxx.se/)
- XML (CRAN)
- RCurl (omegahat)
- SSOAP (omegahat)

以下で紹介するサンプルコードは, Rnw から Rtangle で取得出来る.

エントリ 1 件を XML 形式で取得する方法を例に説明する. エントリーを取得する為にはアクセッション番号を指定する.

```
> # 1. SSOAP のロード
> library(SSOAP)
> # 2. WSDL の指定
> ddbj <- processWSDL ("http://xml.nig.ac.jp/wsdl/GetEntry.wsdl")
> iface <- genSOAPClientInterface(def = ddbj)
> # 3. WEB サービスの呼び出し
> result<-iface@functions$getXML_DDBJEntry("AB000003")
> print(result)
[1] "<?xml version=\"1.0\" standalone=\"no\"?>\n<!DOCTYPE DDBJXML SYSTEM \"DDBJXM"
```

1. で SOAP にアクセスするために必要なパッケージをロード.
2. で使用したい SOAP サービスの wsdl ファイルを指定.
3. で使用したいサービスを呼び出す.

アクセッション番号 AB000002 ～ AB000005 を一度の実行で取得したい場合は

```
> acsession<-c("AB000002","AB000003","AB000004","AB000005")
> result<-sapply(acsession,iface@functions$getFASTA_DDBJEntry)
> print(result["AB000002"])
                                                                AB000002
">AB000002|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, 28S rRNA,\n"
> print(result["AB000003"])
                                                                AB000003
">AB000003|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, 28S rRNA,\n"
> print(result["AB000004"])
                                                                AB000004
">AB000004|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, 28S rRNA,\n"
> print(result["AB000005"])
                                                                AB000005
">AB000005|Rhizoctonia solani genes for 18S rRNA, 5.8S rRNA, 28S rRNA,\n"
```

のようにする.

次に複数の SOAP を用いた連携を行うワークフローのサンプルを紹介する. 例として SRS, GetEntry, Blast を連携させる.

2.2.1.1 キーワード検索システムである SRS 用いて検索をかける

キーワードとして ‘prion’, Division ‘Human’, 分子種 ‘mRNA’ を指定.

```
> # ライブラリのロード
> library(SSOAP)
> # WSDL の指定
> SRS <- processWSDL("http://xml.nig.ac.jp/wsdl/SRS.wsdl")
> SRSiface<-genSOAPClientInterface(def=SRS)
> # WEB サービスの呼び出し（条件 & は &amp; にエンコードすること）
> result<-SRSiface@functions$searchSimple(
+   "[ddbj-AllText:prion*] &amp; [ddbj-Division:hum] &amp; [ddbj-Molecule:mrna]")
> print(result)
[1] "DDBJ:AF187843\nDDBJ:AF187844\nDDBJ:AK090575\nDDBJ:AY008282\nDDBJ:AY569456\n"
```

2.2.1.2 その結果からコーディング領域のアミノ酸配列を抜き出す (GetEntry)

```
> # 検索結果を改行で分割し，配列に格納
> id <- unlist(strsplit(result,"\n"))
> getentry<-genSOAPClientInterface(def=processWSDL(
+                     "http://xml.nig.ac.jp/wsdl/GetEntry.wsdl"))
> # アクセッション No(substring で BBDJ:をサプレス)
> # sapply でアクセッション No を引数にして DAD エントリーを FASTA 形式で取得
> result<-sapply(substring(id,6),getentry@functions$getFASTA_DADEntry)
> print(result)
[1] ">AF187843-1|AAG43448.1|148|Homo sapiens doppel protein\nMRKHLSWWWLATVCM"
[2] ">AF187844-1|AAG43449.1|176|Homo sapiens prion-like protein\nMRKHLSWWWLA"
[3] "character(0)"
[4] ">AY008282-1|AAG21693.1|253|Homo sapiens prion protein\nMANLGCWMLVLFVATW"
[5] ">AY569456-1|AAS80162.1|253|Homo sapiens prion protein\nMANLGCWMLVLFVATW"
[6] ">BC001072-1|AAH01072.1|128|Homo sapiens PRNPIP protein\nMVDGQPSLQQVLERV"
[7] ">BC004456-1|AAH04456.1|128|Homo sapiens PRNPIP protein\nMVDGQPSLQQVLERV"
 [ reached getOption("max.print") -- omitted 12 entries ]]
```

2.2.1.3 相同性検索の一つである Blastp を 比較対象 Swiss-prot のデータベースに指定して実行

```
> blastiface<-genSOAPClientInterface(def=processWSDL(
+                     "http://xml.nig.ac.jp/wsdl/Blast.wsdl"))
> # blast サービスを呼び出す
> result<-blastiface@functions$searchParam("blastp",
+                                          "SWISS",
+                                          unlist(result),
+                                          "-m 8")
> print(result)
[1] "AF187843-1|AAG43448.1|148|Homo\tsp|Q9UKY0|PRND_HUMAN\t99.32\t148\t1\t0\t1\t14"
```

2.2.1.4 アノテーションを GetEntry で取得し, アクセッション番号, プロテイン ID,Swiss-Prot ID, プロテインシンボル, プロテインの定義 を表示する

```
> # 結果を改行コードで分割する
> blastline<-unlist(strsplit(result,"\n"))
> # ID を抜き出す
> swissid <- unlist(lapply(blastline,
+                          function(x){
+                              unlist(strsplit(x,"\\|"))[5]
+                          }))
> # ID から GetSWISSEntry 取得
> #swissentry<-sapply(swissid,getentry@functions$getSWISSEntry)
> swissentry<-sapply(swissid,getentry@functions$getUNIPROTEntry)
> # 空白を圧縮
> swissentry<-gsub("\\s{2,}"," ",swissentry)
> # 改行で分割
> swissentry<-strsplit(swissentry,"\n")
> # ID の編集
> ID<-unlist(lapply(swissentry,
+                   function(x)
+                       paste(substring(x[grep("^ID",x)],4),collapse=" ")))
> # DE の編集
> DE<-unlist(lapply(swissentry,
+                   function(x)
+                       paste(substring(x[grep("^DE",x)],4),collapse=" ")))
> SWISS_ENTRY<-cbind(ID,DE)
> print(SWISS_ENTRY)
       ID
Q9UKY0 "PRND_HUMAN Reviewed; 176 AA."
Q9GJY2 "PRND_SHEEP Reviewed; 178 AA."
Q9GK16 "PRND_BOVIN Reviewed; 178 AA."
       DE
Q9UKY0 "Prion-like protein doppel precursor (PrPLP) (Prion protein 2)."
Q9GJY2 "Prion-like protein doppel precursor (PrPLP)."
Q9GK16 "Prion-like protein doppel precursor (PrPLP)."
 [getOption("max.print") を越えました -- 末尾 73 行を省略します]]
```

参考文献

[1] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2007. ISBN 3-900051-07-0.

[2] Duncan Temple Lang. *RCurl: HTTP request interface*. R package version 0.8-0.

[3] Duncan Temple Lang (duncan@wald.ucdavis.edu). *XML: Tools for parsing and generating XML within R and S-Plus.*, 2007. R package version 1.9-0.

[4] Duncan Temple Lang. *SSOAP: Client-side SOAP access for S*, 2007. R package version 0.4-3.

[5] Robert C Gentleman, Vincent J. Carey, Douglas M. Bates, et al. Bioconductor: Open software development for computational biology and bioinformatics. *Genome Biology*, Vol. 5, p. R80, 2004.

[6] Center for Information Biology and DNA Data Bank of Japan. Web サービス チュ-トリアル. `http://www.xml.nig.ac.jp/tutorial/index_jp.html`.

[7] Center for Information Biology and DNA Data Bank of Japan. Ddbj wsdl リスト. `http://xml.nig.ac.jp/wsdl/index.jsp`.