

# Tavernaを用いたワークフロー構築

富士通 重元 康昌

東海ソフトウェア 桑名 良和

# DDBJにおけるWEBサービス

- 2002年2月からバイオWEBサービスを公開
- DDBJをはじめとするさまざまなデータベースのキーワード検索やエントリ取得機能に加え、相同性検索、多重配列などの25カテゴリ、171メソッドを公開
- 複数のWEBサービスを組み合わせた14のワークフローの事例も公開

# DDBJで公開しているWEBサービス

サービス名	サービスの説明
getentry, SRS, ARSA	20以上のデータベースのエントリ取得やキーワード検索機能を提供
BLAST, FASTA, ClustalW, VecScreen, Mapping	相同性検索や多重配列整列などの解析機能を提供
GIB, GIB-V, GIB-IS, GIB-ENV, GTPS, GTOP	DDBJで開発した2次データベースシステム
Ensembl, NCBIゲノムアノテーション, RefSeq, GO, OMIM	他サイトで開発されたデータベースをDDBJから分かりやすく公開
TxSearch	DDBJ/EMBL/GenBankで管理されている生物分類情報のエントリ取得や検索機能を提供





# XML Central of DDBJ

「XML Central of DDBJ」は科学技術振興調整費「[新世代バイオポータルの研究開発](#)」の支援を受け、また、科学技術振興事業団(JST)[BIRD](#) 事業において開発を継続しています。

[English](#)

## What's New

### DDBJ-XML **XML**

DDBJエントリーをXMLフォーマットで出力しています。



Web services  
in the world

### Webサービス **SOAP**

当サイトは、日本で初めて、SOAPを用いたバイオWebサービスを広く一般に公開致しました。目的は、バイオインフォマティクスにおける標準化と生物情報資源の相互運用性の改良です。Webサービスは、コンピュータ

[http://www.xml.nig.ac.jp/index\\_jp.html](http://www.xml.nig.ac.jp/index_jp.html)

# DDBJのWEBサービスへのアクセス方法

```
#!/usr/bin/perl
```

```
# 1. SOAP Lite のインクルード
```

```
use SOAP::Lite;
```

```
# 2. WSDLの指定
```

```
$getentry = SOAP::Lite -> service(  
http://xml.nig.ac.jp/wsdl/GetEntry.wsdl');
```

```
# 3. WEBサービスの呼び出し
```

```
$entry = $getentry->getFASTA_DDBJEntry("AB000100");
```

```
#ワークフローを構築する場合、次のサービスへつなげる
```

```
#$blast = SOAP::Lite -> service('http://...
```

```
#$result = $blast->searchParam(..., $entry, ...)
```

# ワークフローの事例



## ワークフロー紹介ページ

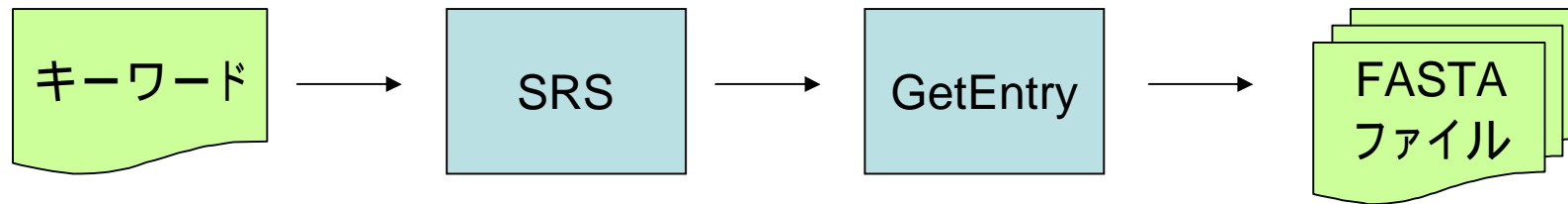
Webサービスを組み合わせたワークフローを紹介いたします。

1. [DDBJ-UniProtワークフロー](#)  
タンパク質名などのキーワードから関連するデータベースエントリを検索し、DDBJアクセッション番号、プロテインID、UniProtID の関連付けを行い表を作成します。
2. [Blast-ClustalWワークフロー](#)  
任意のDNA配列をDDBJデータベースに対して blastnを行い、その結果の上位ヒットのエントリを抽出し ClustalWで比較を行います。
3. [BLASTワークフロー](#)  
任意のアクセッション番号を入力し、DDBJ, Uniprot-Swissprot, PDBに対して BLASTを一括して実行します。

# ワークフローの構築

DDBJのWEBサービスを組み合わせ、ワークフローを構築する。

## 構築するワークフローの例



## ワークフローの手順

1. 任意のキーワードを指定する。
2. SRSを用いてそのキーワードを含むエントリのアクセッション番号一覧を取得する。
3. 各アクセッション番号の配列をGetEntryを用いてFASTAファイルで取得する。

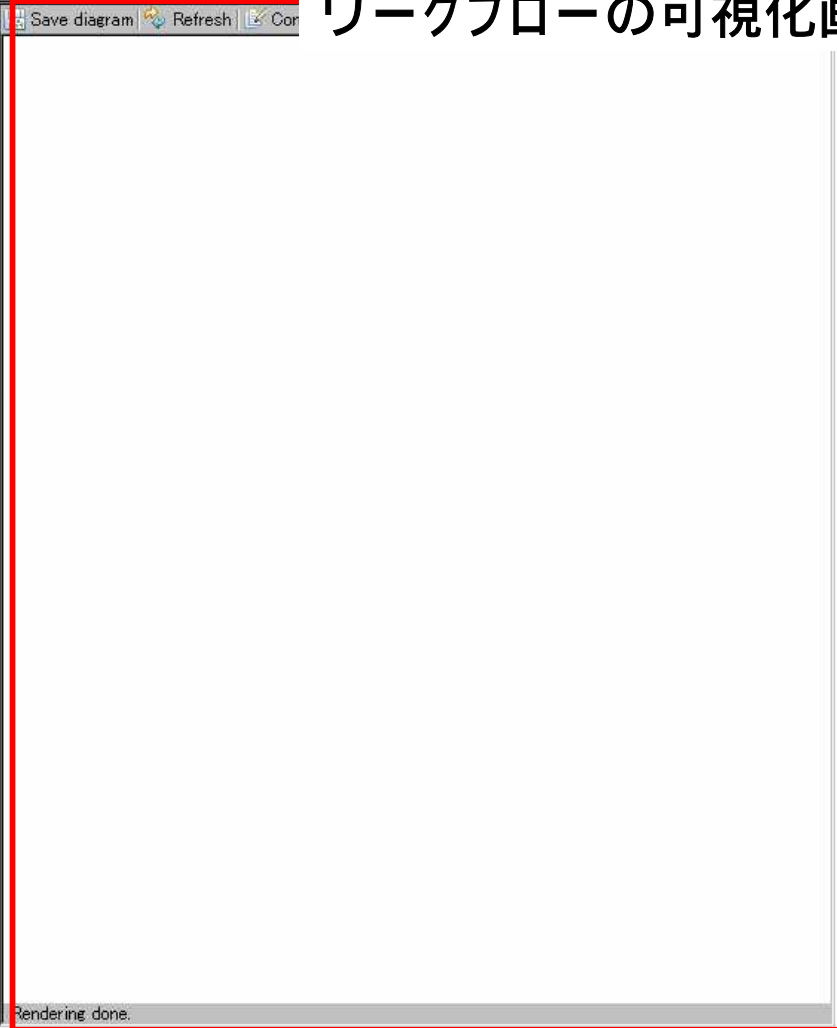
# Taverna起動画面



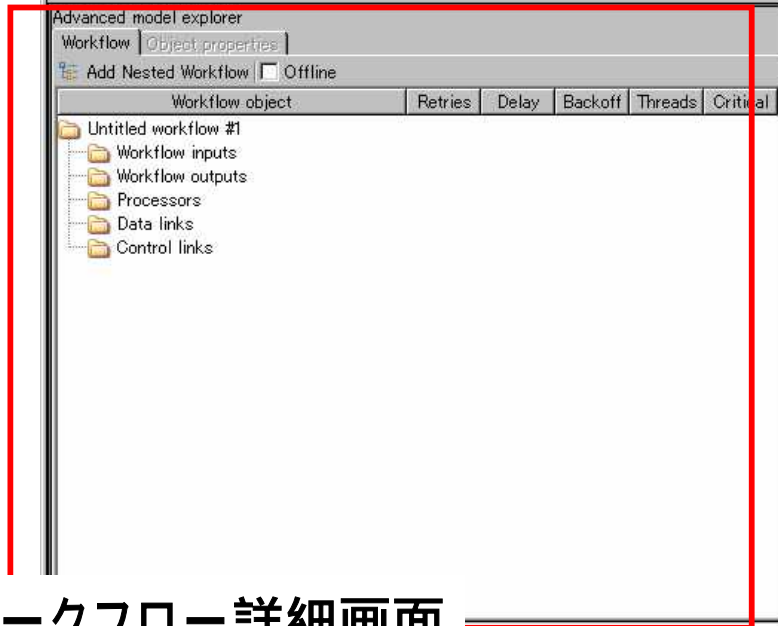
利用可能なWEBサービス



ワークフローの可視化画面



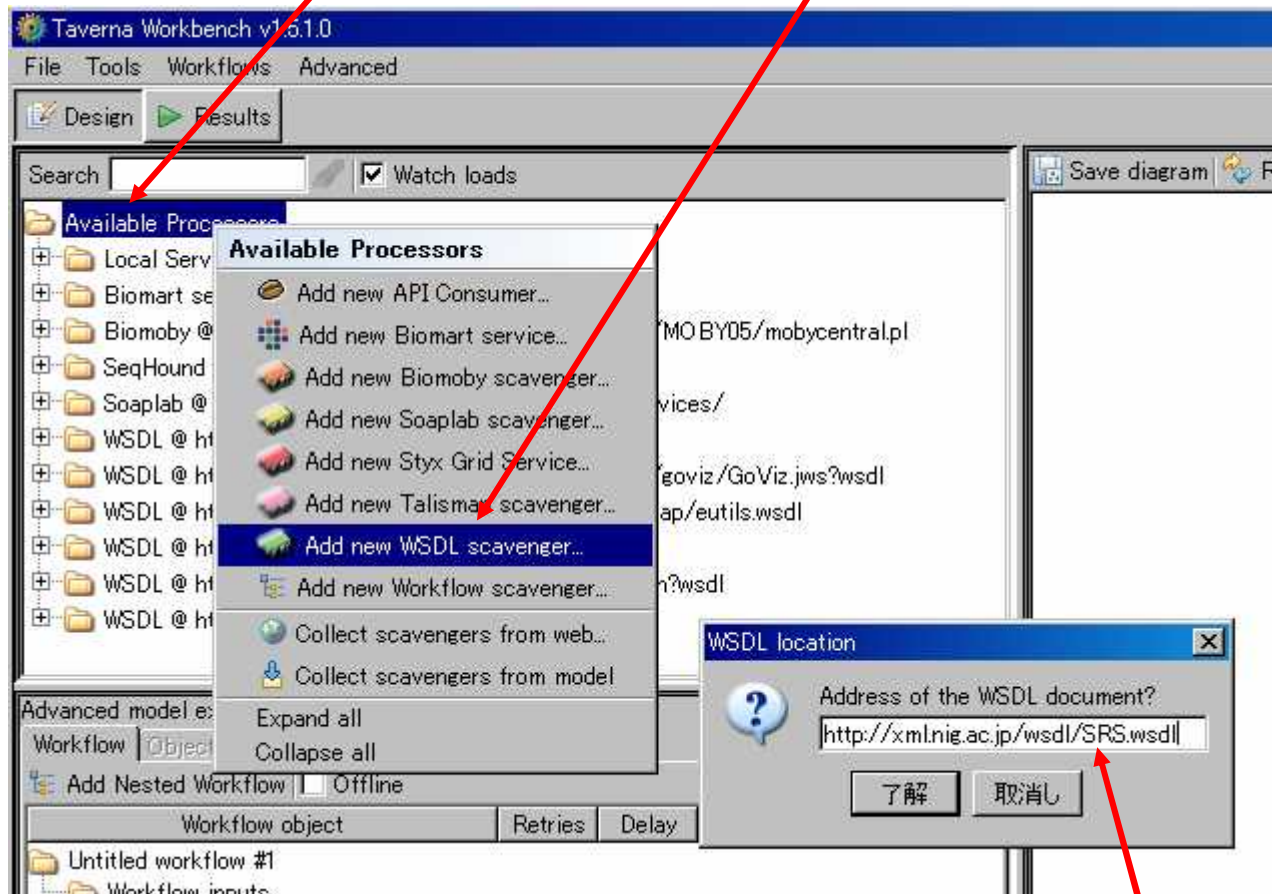
ワークフロー詳細画面



# WSDLの登録

1. 右クリック

2. クリック



3. WSDLのURL指定

# サービスをワークフローに登録

The screenshot shows the Taverna Workbench v1.5.1.0 interface. The top menu bar includes File, Tools, Workflows, Advanced, and Help. The main window is divided into several panes:

- Search:** A list of WSDL services is displayed. The service "WSDL @ http://xmlnig.ac.jp/wsdl/SRS.wsdl" is selected and highlighted in blue. Below it, a port type "porttype: SRS [RPC]" is shown with three operations: "searchSimple", "searchParam", and "searchSimpleAsync".
- Advanced model explorer:** A workflow diagram is shown. A red box labeled "1. ドラッグ" (1. Drag) points to the "SRS" service icon in the "Processors" section of the workflow.
- Workflow Diagram:** The "SRS" processor is expanded, showing its configuration: "query 'text/plain'", "param 'text/plain'", "attachmentList I()", and "Result 'text/plain'". A red box labeled "2. サービスが追加される" (2. Service is added) points to this configuration area.
- Visualization:** A large green box with the text "SRS" is shown in the visualization area. A red box labeled "3. 可視化画面に追加される" (3. Added to visualization screen) points to this box.

At the bottom right of the window, the text "Rendering done." is visible.

# 入力情報を登録

The image illustrates the steps to register input information in a workflow editor. It shows a tree view on the left with folders for 'Workflow inputs', 'Workflow outputs', 'Processors', 'Data links', and 'Control links'. A context menu is open over the 'Workflow inputs' folder, with 'Create New Input...' selected. A dialog box titled 'Name required' prompts for a name, with 'keyword' entered. A large blue arrow points from the first screenshot to the second, where the 'keyword' input is now visible in the 'Workflow inputs' folder.

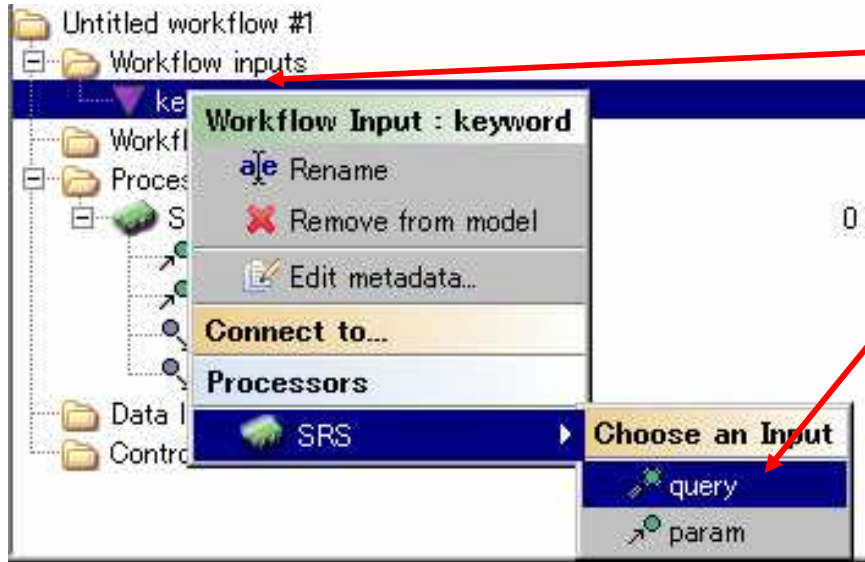
1. 右クリック

2. クリック

3. 入力情報名を指定

4. ワークフロー詳細画面に反映される。

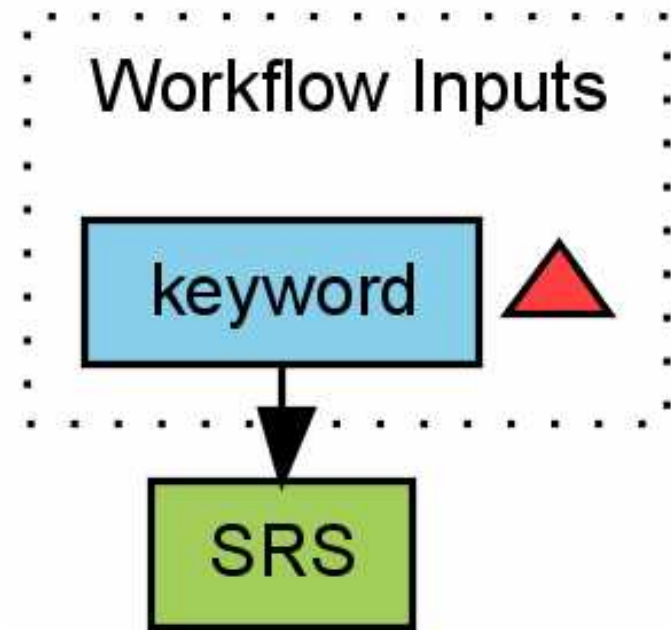
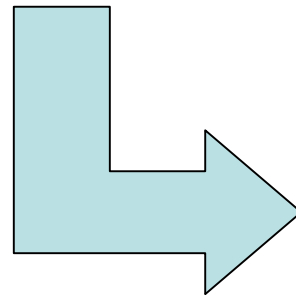
# 入力情報とSRSの接続



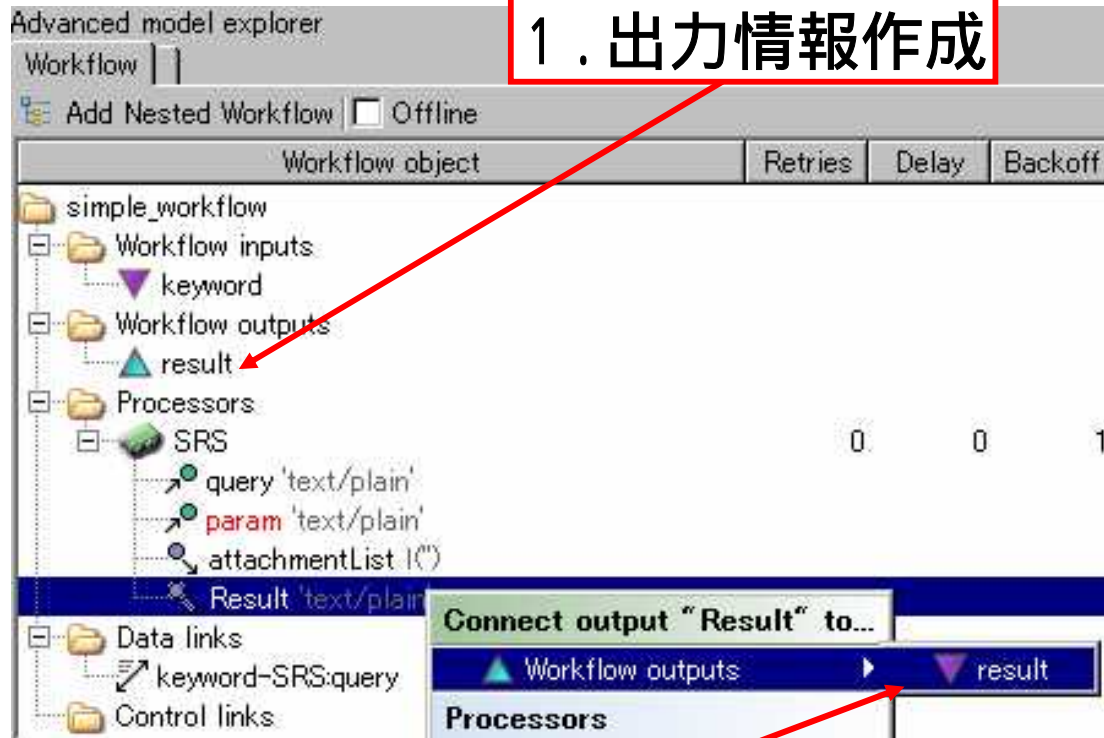
1. 右クリック

2. クリック

3. 可視化画面が更新される



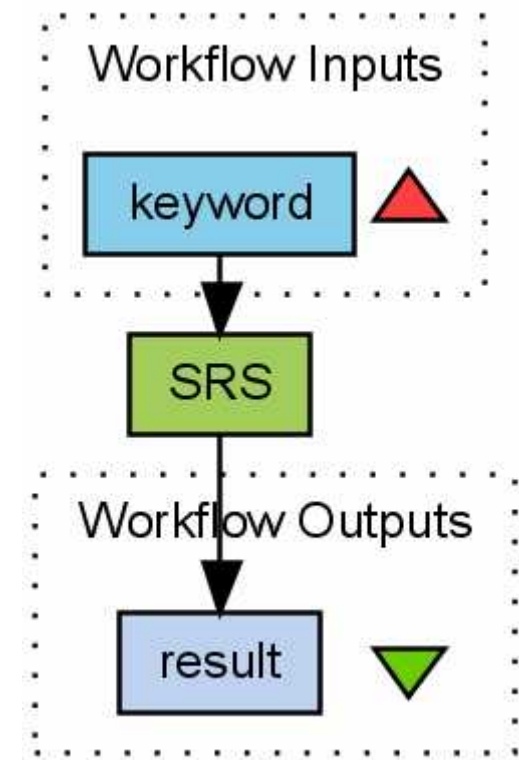
# 出力情報とSRSの接続



1. 出力情報作成

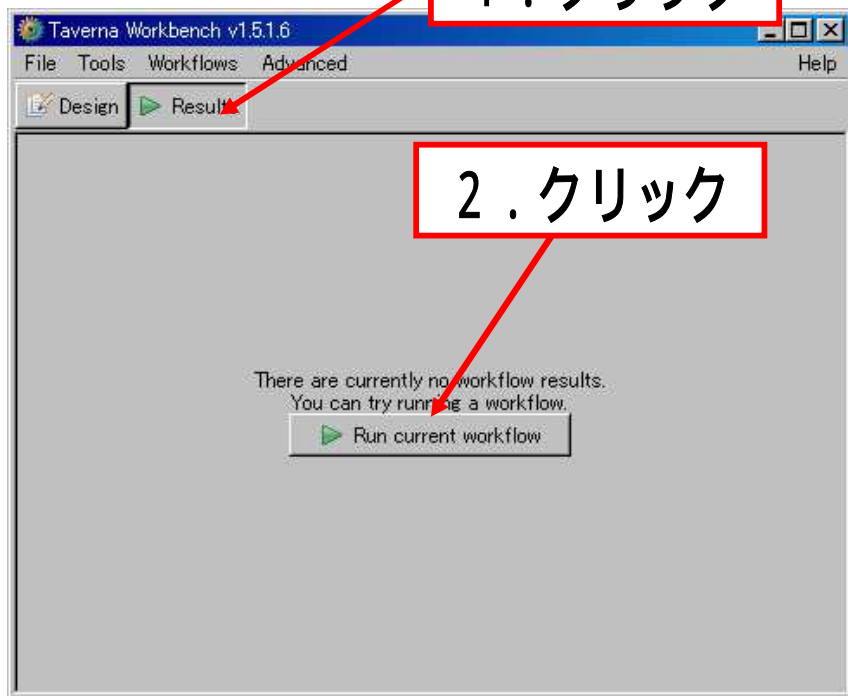
2. SRSの結果と  
出力情報の接続

3. 可視化画面が  
更新される。

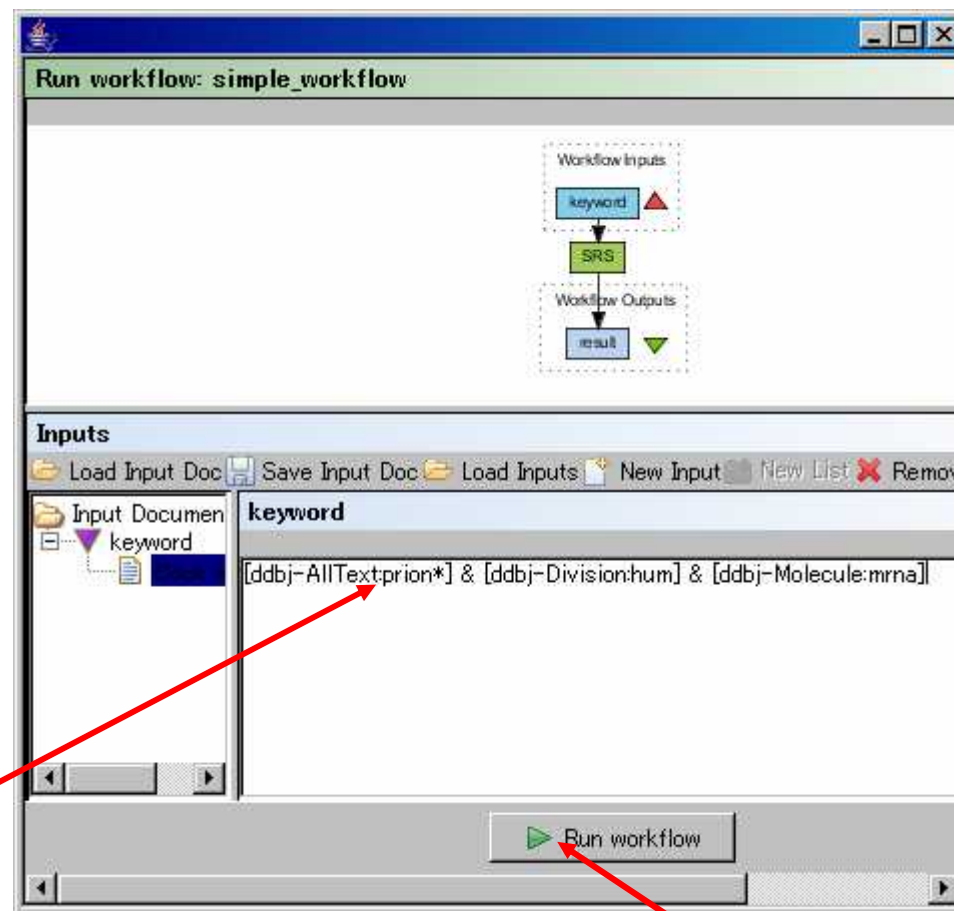


# ワークフローの実行

1. クリック



2. クリック



3. キーワードの入力

(ヒトのエントリ かつ mRNA かつ キーワード prionを含むもの)

4. クリック

# ワークフローの実行結果

Taverna Workbench v1.5.1.6

File Tools Workflows Advanced Help

Design Results

simple\_workflow 13:34

<> Save as XML Save to disk Save to disk as website Execute

Status Results Process report

result

ACCESSION	AF187843
ACCESSION	AF187844
ACCESSION	AY008282
ACCESSION	AY569456
ACCESSION	BC001072
ACCESSION	BC004456
ACCESSION	BC012844
ACCESSION	BC016809
ACCESSION	BC022532
ACCESSION	BC040198
ACCESSION	BC043644
ACCESSION	BT019496
ACCESSION	CR542039
ACCESSION	CR542072
ACCESSION	D00015 N00015
ACCESSION	M13667
ACCESSION	M13899

検索条件を満たすエントリの  
アクセッション番号一覧

# ワークフロー結果の分割(拡張)

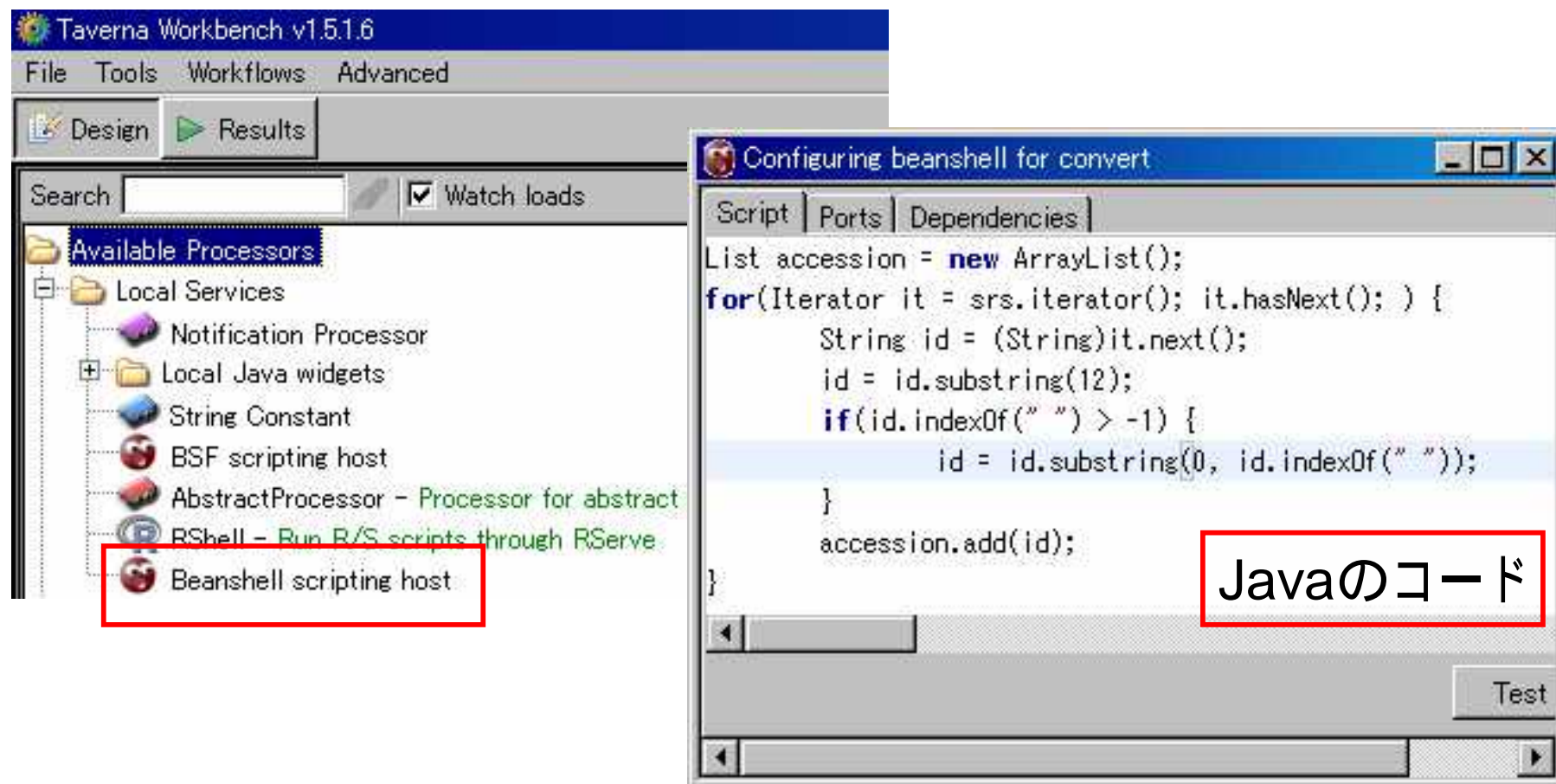
改行コードを区切り文字として分割する

Accession	Value
ACCESSION	AF187843
ACCESSION	AF187844
ACCESSION	AY008282
ACCESSION	AY569456
ACCESSION	BC001072
ACCESSION	BC004456
ACCESSION	BC012844
ACCESSION	BC016809
ACCESSION	BC022532
ACCESSION	BC040198
ACCESSION	BC043644
ACCESSION	BT019496
ACCESSION	CR542039
ACCESSION	CR542072
ACCESSION	D00015 N00015
ACCESSION	M13667
ACCESSION	M13899

- Available Processors
  - Local Services
    - Notification Processor
    - Local Java widgets
      - list
      - io
      - metadata
      - xml
      - ui
      - ncbi
      - conditional
      - net
      - text
        - Byte[] to String
        - String list union
        - Concatenate two strings
        - String list difference
        - Filter list of strings by regex
        - Split string into string list by regular expression**
        - Pad numeral with leading 0s
        - Filter list of strings extracting match to a regex

# 簡単な文字列の処理

‘ACCESSION AF187843’の最初の‘ACCESSION’が余計なので取り除く処理をBeanshellで作成する。



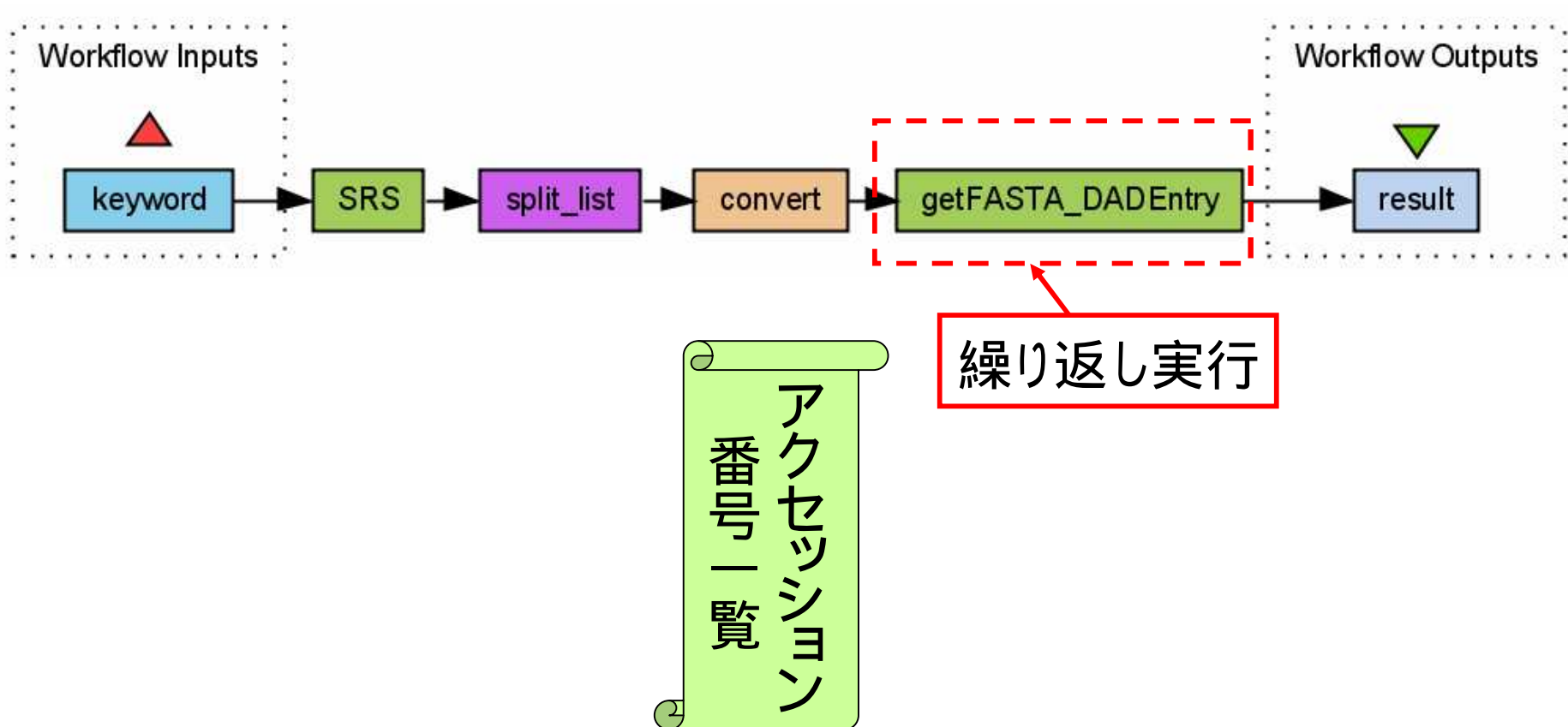
The screenshot shows the Taverna Workbench v1.5.1.6 interface. In the 'Available Processors' list, 'Beanshell scripting host' is highlighted with a red box. A configuration window titled 'Configuring beanshell for convert' is open, showing the following Java code:

```
Script | Ports | Dependencies |
List accession = new ArrayList();
for(Iterator it = srs.iterator(); it.hasNext(); ) {
    String id = (String)it.next();
    id = id.substring(12);
    if(id.indexOf(" ") > -1) {
        id = id.substring(0, id.indexOf(" "));
    }
    accession.add(id);
}
```

The text 'Javaのコード' (Java code) is written in a red box next to the code.

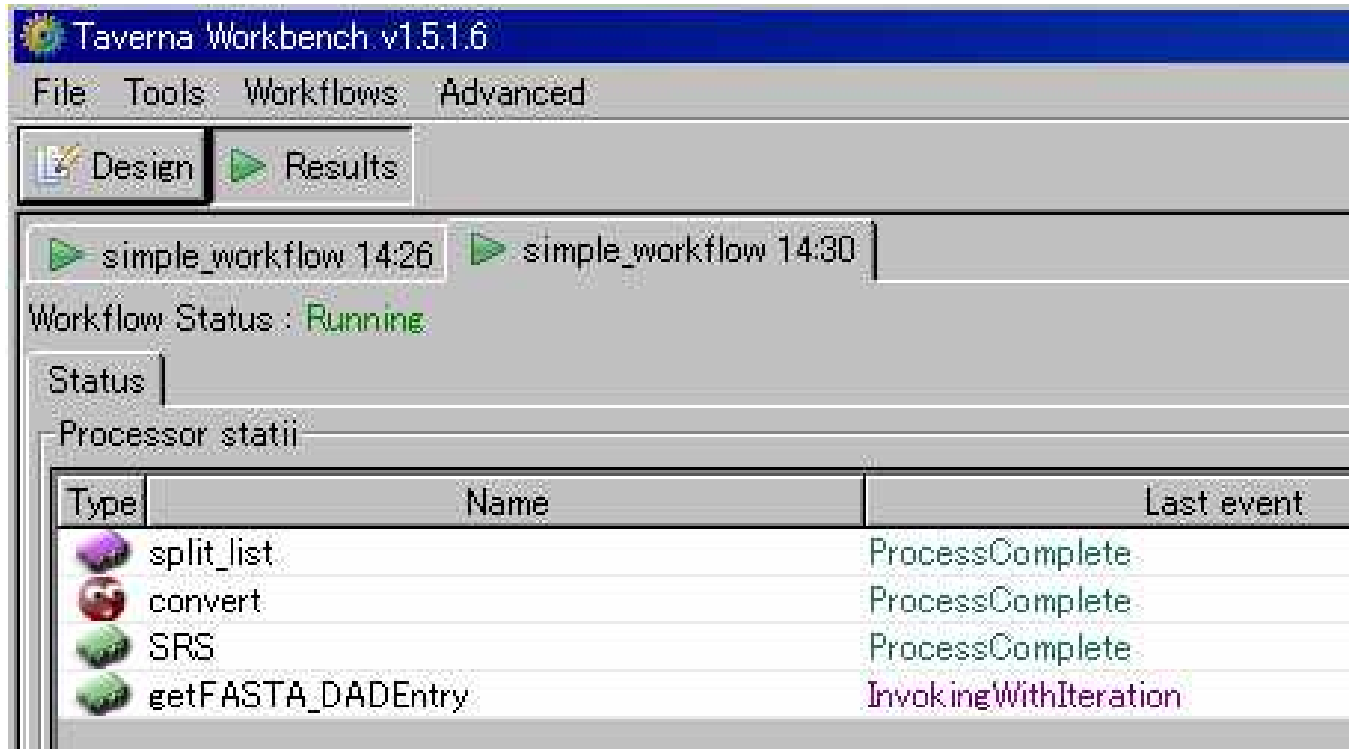
# getentryの実行

Beanshellできれいにしたアクセッション番号一覧を用いて、その一覧の分だけgetentryを繰り返し実行し、配列を取得する。







# ワークフローの実行

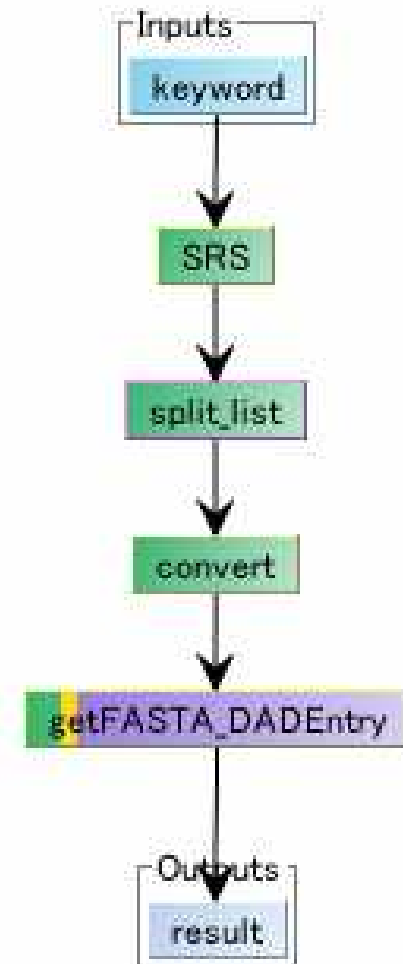
ワークフローの各ステップの様子を確認することができる。



The screenshot shows the Taverna Workbench v1.5.1.6 interface. The menu bar includes File, Tools, Workflows, and Advanced. The Design tab is active, and the Results tab is also visible. Two workflow instances are shown: 'simple\_workflow 14:26' and 'simple\_workflow 14:30'. The Workflow Status is 'Running'. The Processor status table is as follows:

Type	Name	Last event
	split_list	ProcessComplete
	convert	ProcessComplete
	SRS	ProcessComplete
	getFASTA_DADEntry	InvokingWithIteration

split\_list, convert, SRSは完了  
getFASTA\_DADEntryを繰り返し実行中



# ワークフローの結果

ワークフローの最終結果だけでなく、各ステップの結果も見る事ができる。

Taverna Workbench v1.5.1.6

File Tools Workflows Advanced

Design Results

simple\_workflow 14:26 simple\_workflow 14:30

Save as XML Save to disk Save to disk as website Excel

Status Results Process report

Processor status

Type	Name	Last event	Event
split_list		ProcessComplete	2007/03/19 14:30:38
convert		ProcessComplete	2007/03/19 14:30:38
SRS		ProcessComplete	2007/03/19 14:30:38
getFASTA_DADEntry		ProcessComplete	2007/03/19 14:30:46

Graph Intermediate inputs Intermediate outputs

sr

List  
urn:lsidnet.sf.taverna.dataCollection:ff1164b2-3b14-46ac-8739-e00  
text/plain  
ACCESSION AF187843  
text/plain

ACCESSION AF187843

Graph Intermediate inputs Intermediate outputs

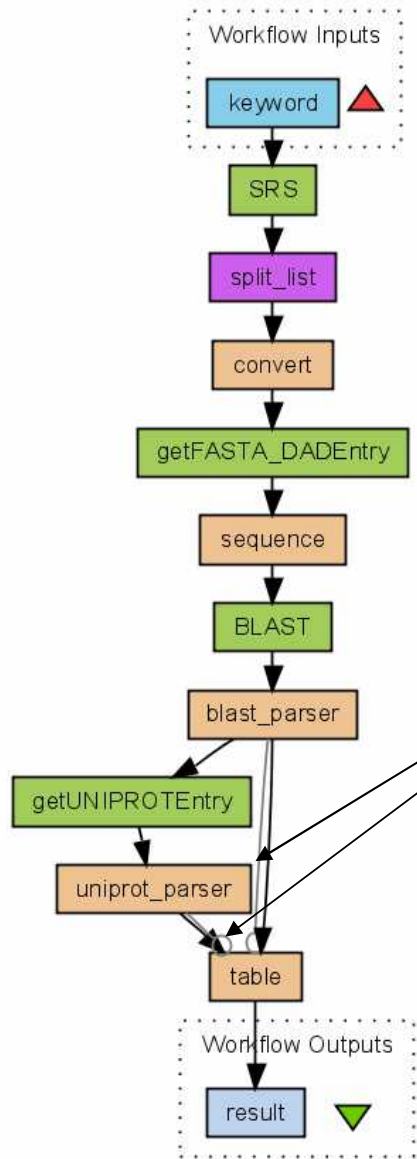
accession

List  
urn:lsidnet.sf.taverna.dataCollection:7dd8d8ef-d185-493a-9e66-0b370d9883f1  
text/plain  
AF187843  
text/plain

ACCESSION AF187843

文字列の変換処理

# ワークフローの拡張



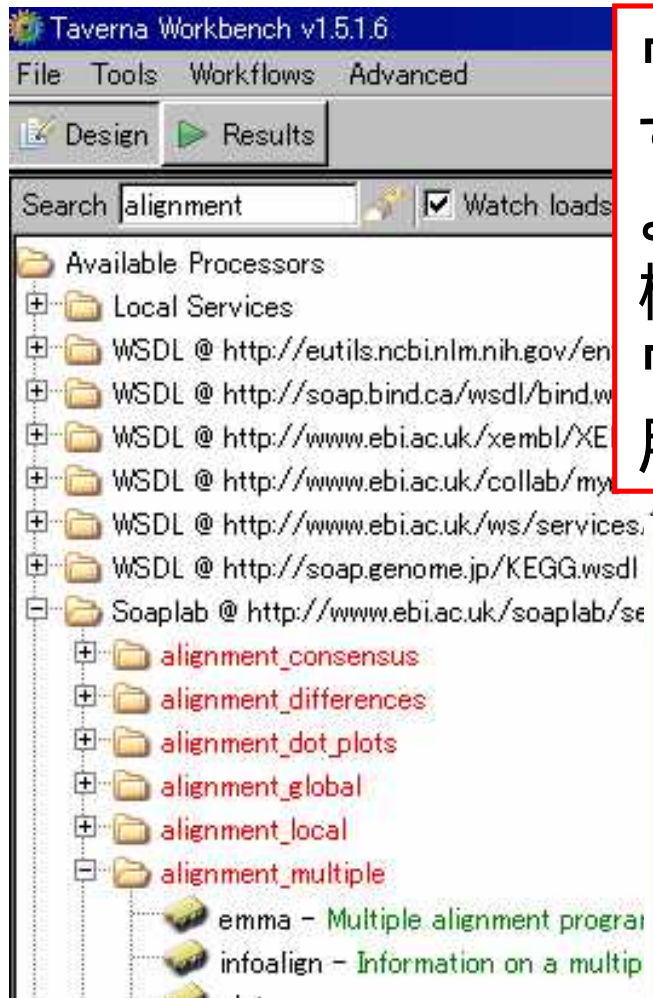
- ・キーワード検索で取得した配列をクエリとしてBLASTを実行
- ・BLASTの結果をパース
- ・UNIPROTのDEFINITION行を取得
- ・テーブル形式で解析結果のまとめ

実行制御 (table処理は、uniprot\_parser, blast\_parserの2つが完了するまで実行しない)

## 検索結果例

(DDBJ)	(protein ID)	(UNIPROT)	(DEFINITION)
AF187843-1	AAG43448.1	Q9UKY0	Prion-like . . .
AF187844-1	AAG43449.1	Q9UKY0	Prion-like . . .

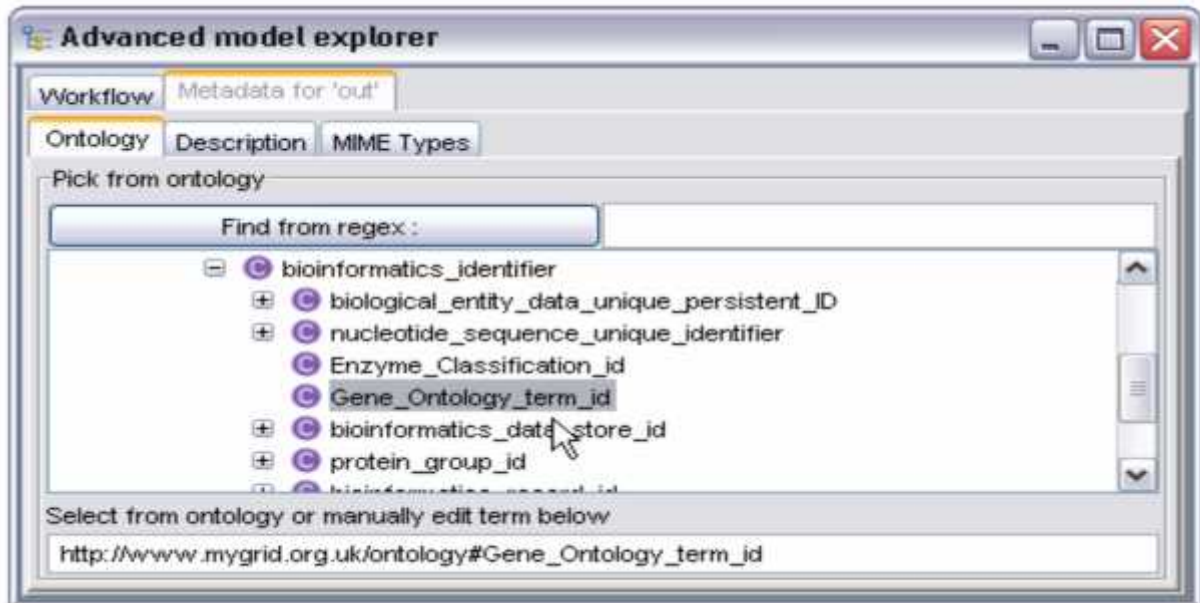
# サービスとワークフローの入出力形式



ワークフロー構築に必要なサービスを検索することができる。

より高度な入出力形式を指定したサービスの検索機能は無いようである。

ワークフローの入出力形式はオントロジーを用いて指定することが可能である。



# WEBサービスをつなぐシステム@DDBJ

あるサービスの実行結果から自動的に次に実行できるサービスをピックアップするシステムを開発中。ワークフローの設計・構築に有用であると思われる。**各サービスの入出力形式の標準化が不可欠である。**

実行結果を使って  
実行できるサービス

トップページ	LOCUS	AB000100	2992 bp	DNA	linear	BCT 05-FEB-1999
データベース	DEFINITION	Synecococcus sp. DNA for intrinsic membrane protein, malK-like protein, cyanase, complete cds.				
配列	ACCESSION	AB000100				
DDBJ	VERSION	AB000100.1				
•キーワード検索	KEYWORDS	cynS; cyanase; cynD; malK-like protein; cynB; intrinsic membrane protein.				
•エントリー取得	SOURCE	Synecococcus sp.				
•フラットファイル取得	ORGANISM	Synecococcus sp. Bacteria; Cyanobacteria; Chroococcales; Synecococcus.				
•FASTA取得	REFERENCE	1 (bases 1 to 2992)				
•系統で検索	AUTHORS	Omata, T.				
	TITLE	Direct Submission				
	JOURNAL	Submitted (26-DEC-1996) to the DDBJ/EMBL/GenBank databases. Tatsuo Omata, School of Agricultural Sciences, Nagoya University, Department of Applied Biological Sciences; Chikusa, Nagoya, Aichi 464-01, Japan (E-mail: g44512a@nucc.cc.nagoya-u.ac.jp, Tel: 052-789-4106, Fax: 052-789-4104)				
	REFERENCE	2				
	AUTHORS	Harano, Y., Suzuki, I., Maeda, S., Kaneko, T., Tabata, S. and Omata, T.				
	TITLE	Identification and nitrogen regulation of the cyanase gene from				

次に遷移可能なフロー

- [sequence](#)
- [BLASTXML出力](#)
- [BLASTTAB出力](#)
- [BLAST標準出力](#)
- [organism](#)
- [シンプル検索](#)
- [詳細検索](#)
- [DDBJ accession](#)
- [フラットファイル](#)
- [取得](#)
- [FASTA取得](#)

実行結果

# Tavernaを用いた個人的な感想

- さまざまなサービスを組み合わせてワークフローを構築することができる。
- 繰り返し処理や選択 (if文相当の機能) なども表現することが可能である。
- 途中結果を見ることができるのは便利である。
- ある研究者が構築したワークフロー (ノウハウ) を別の研究者が実行できるのは有意義である。
  
- サービスとサービスの間はパーサなどが必要である。
- パーサはbeanshellという部品を用いてJavaでコーディングした。
- コーディング作業はエラー箇所を特定するのに苦労した。  
(簡単に特定する方法があるのかもしれませんが。)

# 謝辞

ワークフローの設計およびTavernaシステムの調査を実施するにあたり多くのご助言をいただきました国立遺伝学研究所の菅原先生、阿部先生に感謝いたします。