BioRuby: Object Oriented Open Source Library for Bioinformatics

Toshiaki Katayama[1] k@bioruby.org
Shuichi Kawashima[1] s@bioruby.org
Naohisa Goto[2] ng@bioruby.org
Mitsuteru C. Nakao[3] n@bioruby.org
Yoshinori K. Okuji o@bioruby.org
Minoru Kanehisa[1] kanehisa@kuicr.kyoto-u.ac.jp

[1] Bioinformatics Center, Institute for Chemical Research, Kyoto University, Uji, Kyoto 611-0011, Japan
[2] Genome Information Research Center, Osaka University, Yamadaoka 3-1, Suita, Osaka 565-0871, Japan
[3] Human Genome Center, Institute of Medical Science, University of Tokyo, 4-6-1 Shirokane-dai, Minato-ku, Tokyo 108-8639, Japan

Keywords: open source, ruby scripting language, database interface, object oriented

1 Introduction

BioRuby [6] is the project to bulid useful library for bioinformatics tasks with the object oriented scripting language Ruby, started in late 2000. Ruby is made in Japan and is getting popularity by its simple and powerful syntax in recent years.

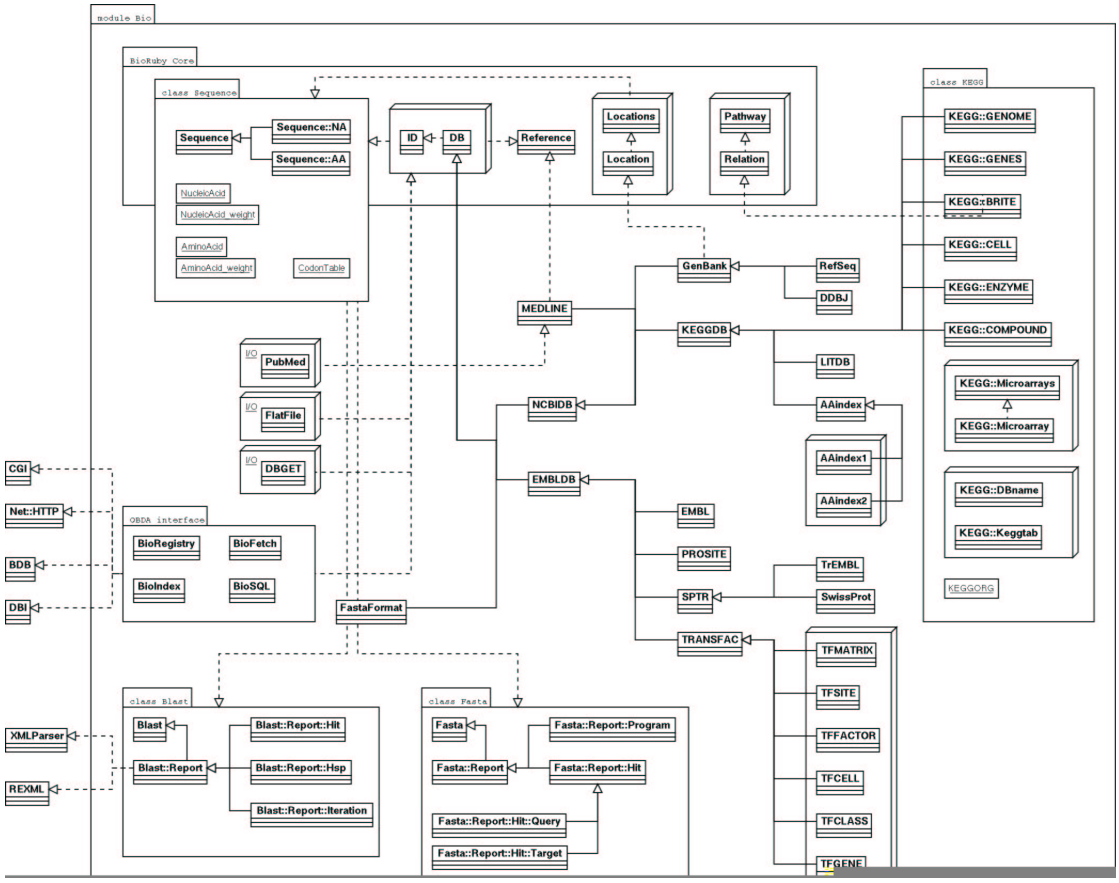
For Perl, Java and Python programmers, there are already existing BioPerl [4], BioJava [3] and BioPython [5] projects as precedent efforts, and organized by the open bio foundation [7, 2, 1]. However, they tend to be complicated unnecessarily because of the limitation of each language characteristics and historical reasons that they have gradually developed. Taking these state into consideration, we find the ruby language is suitable for constructing simple and easy to use open bio library.

In 2002, the first invited-only Open Bio* developers meeting called BioHackathon was held in Arizona and Cape Town, and we have agreed on five new ways to retrieve a record given an identifier from the biological sequence databases. These methods are named OBDA (open bio sequence database access) and include flatfile indexing (simple, Berkeley DB), BioFetch (CGI/HTTP), BioSQL (MySQL, PostgreSQL, Oracle), XEMBL and Corba described below.

2 Project Status

Currently, we have classes for biological sequences and annotations (Bio::Sequence, Bio::Location, Bio::Feature), literature management classes for retrieving and storing reference information (Bio::Reference, Bio::PubMed), parsers for over 20 major biological databases (Bio::DB, Bio::GenBank, Bio::KEGG, etc.), wrappers for sequence analysis softwares such as BLAST, FASTA and EMBOSS (Bio::Blast, Bio::Fasta etc.) and classes for pathway computation (Bio::Pathway, Bio::Relation). Additionally, we have already implemented accessing methods for OBDA via Bio::Registry, Bio::FlatFile, Bio::Fetch and Bio::SQL classes (XEMBL and Corba interfaces are planned).

As for OBDA, Bio::Registry is a mechanism to select accessing methods by using initialization files (`~/.bioinformatics/seqdatabase.ini`). Since, OBDA has five different ways to retrieve the same entry from several databases, users can determine which method to use for obtaining an entry. Flatfile indexing is a simplest method to build against flatfile databases. BioFetch is a method to retrieve an


module Bio
BioRuby Core
class Sequence
Sequence
Sequence::NA
Sequence::AA
NucleicAcid
NucleicAcid_weight
AminoAcid
AminoAcid_weight
CodonTable
ID
DB
Reference
Locations
Location
Pathway
Relation
class KEGG
KEGG::GENOME
KEGG::GENES
KEGG::BRITE
KEGG::CELL
KEGG::ENZYME
KEGG::COMPOUND
KEGG::Microarrays
KEGG::Microarray
KEGG::DBname
KEGG::Keggtab
KEGGOrG
GenBank
RefSeq
DDBJ
MEDLINE
KEGGDB
LITDB
AAindex
AAindex1
AAindex2
NCBIDB
EMBLDB
EMBL
PROSITE
SPTR
TrEMBL
SwissProt
TRANSFAC
TFMATRIX
TFSITE
TFFACTOR
TFCELL
TFCLASS
TFGENE
I/O
PubMed
I/O
FlatFile
I/O
DBGET
CGI
Net::HTTP
BDB
DBI
OBDA interface
BioRegistry
BioFetch
BioIndex
BioSQL
FastaFormat
class Blast
Blast
Blast::Report::Hit
Blast::Report
Blast::Report::Hsp
Blast::Report::Iteration
XMLParser
REXML
class Fasta
Fasta
Fasta::Report::Program
Fasta::Report
Fasta::Report::Hit
Fasta::Report::Hit::Query
Fasta::Report::Hit::Target