 BioSmalltalk: a pure object system and library for bioinformatics

Hernán F. Morales* and Guillermo Giovambattista
Instituto de Genética Veterinaria (IGEVET), CONICET La Plata–Facultad de Ciencias Veterinarias, Universidad Nacional de La Plata, La Plata B1900AVW, CC 296 Argentina

Associate Editor: Janet Kelso

ABSTRACT

Summary: We have developed BioSmalltalk, a new environment system for pure object-oriented bioinformatics programming. Adaptive end-user programming systems tend to become more important for discovering biological knowledge, as is demonstrated by the emergence of open-source programming toolkits for bioinformatics in the past years. Our software is intended to bridge the gap between bioscientists and rapid software prototyping while preserving the possibility of scaling to whole-system biology applications. BioSmalltalk performs better in terms of execution time and memory usage than Biopython and BioPerl for some classical situations.

Availability: BioSmalltalk is cross-platform and freely available (MIT license) through the Google Project Hosting at http://code.google.com/p/biosmalltalk

Contact: herman.morales@gmail.com

Supplementary information: Supplementary data are available at Bioinformatics online.

Received on January 12, 2013; revised on June 5, 2013; accepted on July 3, 2013

1 INTRODUCTION

We present a novel free/open source software (FOSS) platform for the development of bioinformatics software and applications. BioSmalltalk attempts to reconcile the current de facto scripting modalities of textual programming languages with the features of Smalltalk (Goldberg and Robson, 1983), which has a pure object dynamic programming environment.

BioSmalltalk provides similar functionality to other FOSS toolkits for bioinformatics, such as BioPerl (Stajich et al., 2002), Biopython (Cock et al., 2009) and BioJava (Holland et al., 2008), based in industry-leading general-purpose textual programming languages. Precedent of bioinformatics tools exists in Smalltalk, but none of them has tried to provide a bioinformatics Application Programming Interface (API), MolTalk (Diemand and Scheib, 2004) was developed in StepTalk, a scripting environment, for doing structural bioinformatics. Also, a cross-platform Graphical User Interface (GUI) for protein sequence analysis was done in Smalltalk (Wishart et al., 1997).

Object-orientation (OO) is a term first coined by one of the Smalltalk inventors, Alan Kay (1969). It was initially conceived as a programming paradigm based on the recognition of real-world communicating objects in computer simulations (Fichman and Kemerer, 1992; Kölling, 1999). OO features were integrated accordingly to platform limitations, in virtually all major programming languages, and toolkits, including the Bio* projects. The Bio* toolkits’ usage of OO is commonly hybrid or emulated through modules (Cock et al., 2009; Stajich et al., 2002), mixing objects with primitive data types and hampering the use of reflective functionalities (Maes, 1977). BioSmalltalk benefits from decreased source code verbosity, and its execution in a self-contained snapshot system that promotes run-time adaptability, critical for systems where shutdown cycles cannot be tolerated (Hirschfeld and Lämmel, 2005).

2 FEATURES

2.1 Bioinformatics

BioSmalltalk provides objects to manipulate biological sequences and data from databases like the Entrez system (Schuler et al., 1996). It also contains wrappers for command-line tools like ClustalW (Thompson, 1994) and HMMER (Finn, 2011) sequence visualization and format conversion. We based implementation on existing FOSS bioinformatics platforms, specifically BioPerl and Biopython, to prevent educational obsolescence, preserving the familiar object model interfaces for experienced bioinformaticians.

BioSmalltalk contains tokenizers, parsers and formatters for common sequence identifiers, FASTA, BLAST and Entrez XML, PHYLIP (Felsenstein, 1989), Arlequin (Excoffier, 2005) and others. Most parsers use PetitParser (Renggli et al., 2010), a dynamically reconfigurable parser library. Additional features can be found in the project documentation. We did a microbenchmark to compare the performance of our library using the script in Figure 1. We have executed the scripts five times immediately after booting without unnecessary processes (Tests were performed on GNU/Linux Debian kernel 2.6.32-358.2.1.el6.x86_64 using an Intel(R) Xeon(R) CPU E5620 at 2.40GHz, 8 GB DDR3 RAM). Results show that BioSmalltalk has a faster execution time compared with the corresponding BioPerl and Biopython versions. Our approach enabled the removal of unnecessary iterators, thus also reducing the lines of code. Comparison details and scripts are included in the Supplementary Material (Table 1).

2.2 Software engineering

We wrote a cross-platform engine subsystem for enabling multiple interchangeable implementations of packages, which may fail, become unsupported or become too slow. Interchangeable serializers, web client and servers and accessing OS functions were included in our initial release. We have applied design patterns through the library; for example, the Façade pattern...
platforms under Squeak and Pharo Smalltalk (Blacktive-type coercion casting, checking class or function scopes and This style replaces taking care of boilerplate code, static or primi-
refactorings directly through menu options (Opdyke, 1992).

BioSmalltalk project can evolve.
search is building a user-base and solid automated build process.
human-centered long-running software. Of interest for future re-
that our platform is suitable for a bioinformatics evolution to
reflective language for bioinformatics development. We believe
We delivered an interactive programming system using a fully
3R E S U L T S
(Bergel
molecular biology and bioinformatics.
structures and structure analysis. BMC Bioinformatics, 5, 39. Excoffer,L.G. et al. (2005) Arlequin ver. 3.0: an integrated software package for
design methodologies. IEEE Computer, 25, 22–29. Finn,R.D. et al. (2011) HMMER web server: interactive sequence similarity search-

Funding: This work is supported by The National Scientific Research Council in Argentina (CONICET), (PIP 2010-2012 N 379).

Conflict of Interest: none declared.

REFERENCES

3 RESULTS
We delivered an interactive programming system using a fully reflective language for bioinformatics development. We believe that our platform is suitable for a bioinformatics evolution to human-centered long-running software. Of interest for future re-
search is building a user-base and solid automated build process. We are open to collaboration in any of the areas in which BioSmalltalk project can evolve.

Note: LOC, lines of code.

Table 1. Microbenchmark results

<table>
<thead>
<tr>
<th>Environment</th>
<th>LOC</th>
<th>Average execution time (msec)</th>
<th>Peak memory usage (MBytes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BioPerl</td>
<td>23</td>
<td>11.496</td>
<td>88.035</td>
</tr>
<tr>
<td>Biopython</td>
<td>18</td>
<td>9.595</td>
<td>47.443</td>
</tr>
<tr>
<td>BioSmalltalk</td>
<td>11</td>
<td>8.073</td>
<td>45.717</td>
</tr>
</tbody>
</table>

(BioNCBI)BlastReader newFromXML: 'BS617-Alignment.xml')
selectedNodes: #('Hit_id' 'Hsp_align-len' 'Hsp_hseq')0
hitNodesDo: [ :hitNode |
  hitNode hspNodes: [ :hspNode |
    hspNode selectHspAlign = 240 ifTrue: [
      msa addSequence: (BioSequence
        newNamed: hitNode selectAccessionNumber
        sequence: hspNode selectHspHseq) ]].
  msa asFasta outputTo: outFileStream.
outFileStream close.

Fig. 1. A downloaded NCBI BLAST XML result is filtered with alignments matching 240 nucleotide bases. Alignment object is built from sequences and exported as FASTA in a file.

Bioinformatics