Sequence representation: a comparison of prototypes

Fabien Campagne

Introduction

The need to store, retrieve, and manipulate sequence data has led to the development of a variety of sequence representation frameworks. These frameworks are designed to provide a uniform way to handle sequence data, allowing for efficient data access and manipulation. The choice of representation is crucial as it can affect the performance and usability of bioinformatics applications. In this section, we will compare the properties of different sequence representation frameworks.

Design patterns and differences

Some frameworks provide very rich sequence representations. These frameworks may include additional information such as annotations or alignments, which can be useful for certain applications. Other frameworks may focus on simplicity and efficiency, providing a basic representation of sequence data. The choice of framework will depend on the specific needs of the application.

Defining the Benchmark

This section defines the benchmark for comparison. It is based on a set of criteria that are common across sequence representation frameworks. These criteria include data import, export, and manipulation capabilities.

Data import facility

Most frameworks support data import in multiple formats. Some frameworks may support a larger number of sequence formats, while others may focus on specific formats like FASTA or PIR. It is important to consider the variety of formats supported when selecting a framework.

Data export facility

Support for types, number and size of sequences

Representation of annotations

Support for analysis tools

Support for persistence

Support for Knowledge Discovery Environments

Implementation status

Testing

Licensing

Remarks

Comparison

<table>
<thead>
<tr>
<th>Framework</th>
<th>Data import facility</th>
<th>Data export facility</th>
<th>Support for types, number and size of sequences</th>
<th>Representation of annotations</th>
<th>Support for analysis tools</th>
<th>Support for persistence</th>
<th>Support for Knowledge Discovery Environments</th>
<th>Implementation status</th>
<th>Testing</th>
<th>Licensing</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSA</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>In progress</td>
<td>No</td>
<td>Yes</td>
<td>Neutral</td>
</tr>
<tr>
<td>BioPerl</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Implemented</td>
<td>No</td>
<td>Yes</td>
<td>Neutral</td>
</tr>
<tr>
<td>BioJava</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Implemented</td>
<td>No</td>
<td>Yes</td>
<td>Neutral</td>
</tr>
<tr>
<td>Crover</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Implemented</td>
<td>No</td>
<td>Yes</td>
<td>Neutral</td>
</tr>
</tbody>
</table>

Conclusions

Several conclusions can be drawn from this comparison. First, the frameworks that are designed to be used with CORBA are in general more oriented towards distributed systems. Second, the frameworks that are designed to be used with Java tend to be more object-oriented in nature. The choice of framework will depend on the specific needs and requirements of the application.

References

[2] J. Of Molec. Graph. & Mo...