

# **RNAemo**

Análisis de Requerimientos de Software

*Second Iteration*

Franco Gaspar Riberi

6 de noviembre de 2012

## Índice

<b>1. Introducción</b>	<b>3</b>
1.1. Propósito . . . . .	3
<b>2. Problema</b>	<b>3</b>
2.1. Solución propuesta . . . . .	3
<b>3. Algoritmos y Métodos</b>	<b>4</b>
<b>4. Búsqueda de datos mínimos</b>	<b>4</b>
4.1. $RNA_m$ . . . . .	4
4.2. $miRNA$ . . . . .	6
4.3. Determinación de constantes para calcular score . . . . .	7
<b>5. Control sobre las secuencias</b>	<b>7</b>
<b>6. Herramientas</b>	<b>7</b>
6.1. R . . . . .	7
6.2. Software para Humanizar . . . . .	8
6.3. BioEdit . . . . .	9
6.4. Jalview . . . . .	9
6.5. FASTA . . . . .	9
6.5.1. Representación de secuencias . . . . .	10
<b>7. Software para folding</b>	<b>12</b>
<b>8. Software para hibridar</b>	<b>13</b>
<b>9. Bibliotecas existentes a utilizar</b>	<b>15</b>
9.1. MiLi . . . . .	15
9.2. FuD . . . . .	16
9.2.1. Diseño de original de FuD . . . . .	16
9.3. BioPP . . . . .	17
9.4. Biopp-filer . . . . .	18
9.5. Odf-gen . . . . .	18
9.6. Fideo . . . . .	18
<b>10. Conclusiones</b>	<b>18</b>

# 1. Introducción

## 1.1. Propósito

El presente documento corresponde al análisis de requerimientos de la *iteración 1* de la tesis de grado referente a la carrera Licenciatura en Ciencias de la Computación de la UNRC, “*RNAemo*”, a cargo de Franco Gaspar Riberi.

A continuación se detalla dicho análisis.

# 2. Problema

Actualmente no hay evidencia respecto a si existe diferencia en cuanto a reconocimiento de  $mi$ RNA en los  $RNA_m$  virales y sus supuestos humanizados. Por lo cual es necesario determinar la capacidad de hibridación de los microRNA en la secuencia viral tal como existe en la naturaleza y en la secuencia humanizada, para poder establecer conclusiones al respecto. En otras palabras, determinar si el uso de codones divergente con respecto al huésped podría ser una consecuencia de la presión evolutiva generada por los  $mi$ RNA. Si eso es así, los  $mi$ RNA deberían tener menor capacidad de unirse al RNA viral que al genoma viral humanizado.

En principio esa respuesta es importante desde el punto de vista biológico, y daría una herramienta importante para desarrollos posteriores, por ejemplo actualmente se está estudiando el uso de virus modificados para combatir cánceres, el programa podría predecir que virus sería menos afectado por los  $mi$ RNA en células cancerosas y tenerlo en cuenta en el diseño del virus modificado.

## 2.1. Solución propuesta

Para determinar la divergencia en la hibridación de los  $mi$ RNA en la secuencia viral original y humanizada, se contabilizará por separado la cantidad de  $mi$ RNA que se hibridan a una secuencia viral y a la humanizada. Con esta información, se calcularán dos score de matching por cada genoma de las secuencias (viral y humanizada). El primero, dará un porcentaje de hibridación; el segundo, involucrará valores para las uniones A=T y G=C que permitirán calcular un score estimando aproximadamente la energía libre de estas uniones. Además se determinará si los nucleótidos del  $RNA_m$  están apareados o no, determinando así que nucleótidos están disponibles para aparearse con el  $mi$ RNA.

### 3. Algoritmos y Métodos

Para calcular el score de matching sobre las secuencias se utilizará la siguiente fórmula:

$$\frac{(\#AT \times \text{constAT} + \#GC \times \text{constGC})}{(\text{totalAT} \times \text{constAT} + \text{totalGC} \times \text{constGC})} \quad (1)$$

donde:

- **#AT**: cantidad de Adenina que hace matching con Timina, o viceversa.
- **#GC**: cantidad de Guanina que hace matching con Citosina, o viceversa.
- **constAT**: valor predeterminado para el apareo A=T.
- **constGC**: análogo al anterior, pero con apareo G=C.
- **total AT**: total de adenina y timina (apareadas o no).
- **totalGC**: total de guanina y citosina (apareadas o no).

**constAT** y **constGC** tendrán diferentes valores según el score que se calcule. Si se calcula el score de matching en porcentaje, las constantes tendrán valor 1, pero en contrapartida, si se calcula el score de matching teniendo en cuenta las uniones, se utilizarán los valores que se mencionan en la subsección 4.3.

## 4. Búsqueda de datos mínimos

Esta etapa consistió en la recolección de datos necesarios para posteriormente realizar las primeras pruebas del producto a desarrollar. Esta fase involucró la recolección de RNA mensajeros,  $m_i$ RNA, constantes para calcular el score de matching teniendo las uniones (A=T, G=C) y el software humanizador de secuencias (Section 6.2).

### 4.1. RNA<sub>m</sub>

La bases de datos biológicas son DNA céntricas, es decir que si un virus tiene su información como RNA, la secuencia archivada esta expresada como DNA (Timina en lugar de Uracilo). Por lo cual, no se encontró RNA<sub>m</sub> directamente, sino que como gen.

Los virus obtenidos para las pruebas se obtuvieron de [3], el cual brinda el número de acceso a *GenBank*<sup>1</sup> para su obtención. Dentro de los diferentes códigos que se mencionan en [3] se obtuvieron los siguientes:

- Coxsackievirus A9 (Griggs; D00627).
- Enterovirus 71 (TW/2272/98; AF119795).
- Hepatitis A virus (HAF-203; AF268396).
- Poliovirus type 3 (23127; X04468).
- Rhinovirus type 89 (HRV89; M16248).
- Hepatitis E virus (Hyderabad; AF076239).
- Norwalk virus (BS5; AF093797).
- Astro-virus type 1 (Oxford; L23513).
- Dengue-1 virus (PDK-13; AF180818).
- Dengue-2 virus (S1; NC\_001474).
- Dengue-3 virus (H87; M93130).
- Dengue-4 virus (NC\_002640).
- GB virus C (G05BD; AB003292).
- Hepatitis C virus (HCV-A; AJ000009).
- Japanese encephalitis virus (GP78; AF075723).
- Murray Valley virus (MVE-1-51; AF161266).
- West Nile virus (2741; AF206518).
- Western tick-borne encephalitis virus (Neudoerfl; U27495).
- Yellow fever virus (Trinidad 79A; AF094612).

---

<sup>1</sup>Es una base de datos de secuencias genéticas. Una colección anotada de todas las secuencias de DNA disponibles al público. La base de datos GenBank está diseñada para proporcionar y fomentar el acceso de la comunidad científica a la mayor parte de información actualizada y completa la secuencia de DNA. Por lo tanto, *NCBI* (National Center for Biotechnology Information: <http://www.ncbi.nlm.nih.gov/>) no impone restricciones sobre el uso o la distribución de los datos GenBank.

- Eastern equine encephalitis virus (U01034).
- O'nyong-nyong virus (SG650; AF079456).
- Ross River virus (NB5092; M20162).
- Rubella virus (Cendehill; AF188704).
- Sindbis virus (Edsbyn 82-5; M69205).

A partir de los códigos de acceso, de la web: <http://www.ncbi.nlm.nih.gov/genbank/> se obtuvieron los DNA de cada virus en formato FASTA. Para traducir los DNA a RNA (U por T), se utilizará la herramienta *BioEdit*. Alternativamente, se hará uso de la herramienta *Jalview*, o de la librería *Biopp*.

## 4.2. $mi$ RNA

Se obtuvo una base de datos de  $mi$ RNA en formato FASTA. Dicha base de datos fue descargada de [http://www.mirbase.org/cgi-bin/mirna\\_summary.pl?org=hsa](http://www.mirbase.org/cgi-bin/mirna_summary.pl?org=hsa). De la misma, se seleccionaron al azar aproximadamente 50  $mi$ RNA para luego realizar las primeras pruebas. Los  $mi$ RNA están formados por 20 nucleótidos aproximadamente. A continuación se exhiben algunas de las secuencias:

- AAGAUGUGGAAAAAUUGGAAUC
- CAGUGGUUUUACCCUAUGGUAG
- CAGGCAGUGACUGUUCAGACGUC
- AUCAUAGAGGAAAAUCCAUGUU
- AAGAAGAGACUGAGUCAUCGAAU
- CUCUAGAGGGAAGCGCUUUCUG
- GUUUGCACGGGUGGGCCUUGUCU
- AAAGACCGUGACUACUUUUGCA
- UGUAAACAUCCCCGACUGGAAG
- UGGGCGAGGGGUGGGCUCUCAGAG

Otros posibles lugares de descarga de base de datos de  $mi$ RNA consultados son:

- <http://micrornadatabase.com/>.
- <http://microrna.org/>.
- <http://mirdb.org/miRDB/>.
- <http://www.mirbase.org/>.

### 4.3. Determinación de constantes para calcular score

Las constantes empleadas para el cálculos de score fueron extraídas de [8]. Las misma corresponden a valores razonables, dado que depende del contexto.

Los valores son:

$$\text{Unión CG} = \Delta G(\text{CG}) = \Delta G(\text{GC}) = -3$$

$$\text{Unión AU} = \Delta G(\text{AU}) = \Delta G(\text{UA}) = -2$$

$$\text{Unión GU} = \Delta G(\text{GU}) = \Delta G(\text{UG}) = -1$$

## 5. Control sobre las secuencias

Es necesario que las secuencias se encuentren en el marco de lectura correcto, de lo contrario significa que se perdió un nucleótido y todos los tripletes están cambiados. Para comprobar esto, es necesario traducir la cadena de nucleótidos a una cadena de aminoácidos. Esta tarea se realizará utilizando la herramienta *Bioedit*, como así también se hará uso de la librería *BioPP* la cual provee esta funcionalidad.

Al realizar el cambio, no deben aparecer codones stop (representados por un asterisco “\*”). Si no aparecen codones stop significa que las secuencias están en el marco correcto de lectura (in frame).

## 6. Herramientas

A continuación se describen las herramientas y software necesarios para el desarrollo del producto.

### 6.1. R

$R^2$  es un entorno especialmente diseñado para el tratamiento de datos, cálculo y desarrollo gráfico. Permite trabajar con facilidad con vectores y matrices y ofrece diversas herramientas para el análisis de datos.

El lenguaje de programación R forma parte del proyecto GNU<sup>3</sup> y puede verse como una implementación alternativa del lenguaje S, desarrollado en AT&T Bell Laboratories. Se distribuye bajo la licencia GNU GPL y está disponible en una gran variedad de sistemas UNIX, como así también en Windows y MacOS.

---

<sup>2</sup><http://www.r-project.org>

<sup>3</sup>El proyecto GNU se inició en 1984 con el propósito de desarrollar un sistema operativo compatible con Unix que fuera de software libre. <http://www.gnu.org/>

R también es un entorno en el que se han ido implementando diversas técnicas estadísticas. Algunas de ellas se encuentran en la base de R pero muchas otras están disponibles como paquetes (packages<sup>4</sup>).

En resumen, R proporciona un entorno de trabajo especialmente preparado para el análisis estadístico de datos. Sus principales características son:

- Proporciona un lenguaje de programación propio. Basado en el lenguaje S, que a su vez tiene muchos elementos del lenguaje C.
- Objetos y funciones específicas para el tratamiento de datos.
- Es software libre. Permite la descarga de librerías con implementaciones concretas de funciones gráficas, métodos estadísticos, etcétera.

R será utilizado como software para graficar. Se utilizarán los scripts generados por SAARNA<sup>5</sup>.

## 6.2. Software para Humanizar

*GeneDesign*<sup>6</sup> [5][6][7] será el software utilizado para humanizar las cadenas de nucleótidos. El mismo, corresponde a un software libre usado por biólogos moleculares, el gobierno y los sectores farmacéuticos, químicos, agrícolas y biotecnológicos para el diseño [4], clonación y validación de secuencias genéticas.

*GeneDesign* automatiza el proceso de determinar qué pares de bases deben ser unidos entre sí en un orden determinado para un gen.

Un gen codifica para una proteína específica, y el orden de los cientos o miles de pares de bases que componen ese gen determina el orden de los bloques de construcción de aminoácidos que componen esa proteína.

Básicamente, consta de seis módulos que pueden ser utilizados individualmente o en serie para automatizar las tareas necesarias para diseñar y manipular secuencias de DNA sintético. El programa permite al usuario empezar con la secuencia del aminoácido que componen la proteína o las bases que forman el gen que codifica para esta proteína. Entonces el usuario se mueve a través de una serie de pasos que guían el diseño del gen y el vector que llevan el gen en la célula.

Se presentan tres opciones para el uso de este software:

- *Instalado*: corresponde a tener *geneDesign* instalado en la PC, y el producto a desarrollar lo invocará a medida que lo requiera.

---

<sup>4</sup>Estos paquetes están disponibles en <http://cran.au.r-project.org/>.

<sup>5</sup>[saarna.googlecode.com](http://saarna.googlecode.com)

<sup>6</sup>Descarga de: <https://github.com/GeneDesign/GeneDesign>



- *Online*: el producto a desarrollar utiliza *geneDesign* de forma online a medida que lo necesita. (link online: <http://genedesign.thruhere.net/gd>)
- *Offline*: correr de forma online *geneDesign* y generar con su salida dos archivos en formato FASTA. Uno de ellos, con el RNA<sub>m</sub> original, y el otro con el humanizado. Luego el producto a desarrollar tomará los archivos mencionados como entrada.

El producto a desarrollar empleará la opción número 1 (*Instalado*) otorgando de esta forma mayor independencia y evitando la necesidad de conexión a internet para obtener los resultados. Para esta opción es necesario descargar el código de *geneDesign* del repositorio git e instalar y configurar el servidor web apache2<sup>7</sup>.

### 6.3. BioEdit

*BioEdit*<sup>8</sup> es un editor de alineación de secuencias biológicas escrito para Windows 95/98/NT/2000/XP/7. Permite la alineación y manipulación de secuencias relativamente fácil. Puede utilizarse *wine* para su emulación en Linux.

### 6.4. Jalview

*Jalview*<sup>9</sup> es un editor de múltiples alineación escrito en Java, multiplataforma. El programa fue originalmente escrito por Michele Clamp. Se utiliza ampliamente por una variedad de servidores web (e.g., el servidor EBI ClustalW), pero está disponible como un editor de efectos de alineación general. *Jalview* tiene una amplia gama de funciones además de la alineación de secuencias múltiples, permite la visualización y edición incluyendo el cálculo de los árboles filogenéticos y la visualización de estructuras moleculares.

### 6.5. FASTA

*FASTA* [1] es un formato de archivo informático basado en texto, utilizado para representar secuencias de ácidos nucleicos, o de péptido, y en el que los pares de bases o los aminoácidos se representan usando códigos de una única letra. El formato también permite incluir nombres de secuencias y comentarios que preceden a las secuencias en sí.

---

<sup>7</sup><http://httpd.apache.org/>

<sup>8</sup><http://www.mbio.ncsu.edu/BioEdit/bioedit.html>

<sup>9</sup><http://www.jalview.org/>

Una secuencia bajo formato *FASTA* comienza con una descripción en una única línea (línea de cabecera), seguida por líneas de datos de secuencia. La línea de descripción se distingue de los datos de secuencia por un símbolo “>”. La palabra siguiente a este símbolo es el identificador de la secuencia, y el resto de la línea es la descripción (ambos son opcionales). No debería existir espacio entre el “>” y la primera letra del identificador. La secuencia termina si aparece otra línea comenzando con el símbolo “>”, lo cual indica el comienzo de otra secuencia.

A continuación se exhibe un ejemplo de una secuencia en tal formato:

```
>gi|5524211|gb|AAD44166.1| cytochrome b [Elephas maximus maximus]
LCLYTHIGRNIYYGSYLYSETWNTGIMLLITMATAFMGYVLPWGQMSFWGATVITNLFSaipYIGTNLV
EWIWGGFSVDKATLNRFFAFHFILPFTMVALAGVHLTFLHETGSNNPLGLTSDSDKIPFHPYYTIKDFLG
LLILILLILLALLSPDMLGDPDNHMPADPLNTPLHIKPEWYFLFAYAILRSVPNKLGGVLALFLSIVIL
GLMPFLHTSKHRSMMLRPLSQALFWTLTMDLLTLTWIGSQPVEYPYTIIGMASILYFSIILAFPLIAGX
IENY
```

#### 6.5.1. Representación de secuencias

Cada línea de una secuencia debería tener algo menos de 80 caracteres. Las secuencias pueden corresponder a secuencias de proteínas o de ácidos nucleicos, y pueden contener huecos (o gaps) o caracteres de alineamiento.

- Los códigos de ácidos nucleicos soportados son:

Código ácido nucleico	Significado
-	hueco (gap) de longitud indeterminada
A	Adenosina
B	G T C (no A) (B viene tras la A)
C	Citosina
D	G A T (no C) (D viene tras la C)
G	Guanina
H	A C T (no G) (H viene tras la G)
K	G T (cetona/Ketone)
M	A C (grupo aMino)
N	A G C T (cualquiera/aNy)
R	G A (puRina)
S	G C (interacción fuerte/Strong interaction)
T	Timidina
U	Uracilo
V	G C A (no T, no U) (V viene tras la U)
W	A T (interacción débil/Weak interaction)
X	máscara
Y	T C (pirimidina/pYrimidine)

- Los códigos de aminoácidos soportados son:

Código aminoácido	Significado
A	Alanina
B	Ácido aspártico o Asparagina
C	Cisteína
D	Ácido aspártico
E	Ácido glutámico
F	Fenilalanina
G	Glicina
H	Histidina
I	Isoleucina
K	Lisina
L	Leucina
M	Metionina
N	Asparagina
O	Pirrolisina
P	Prolina
Q	Glutamina
R	Arginina
S	Serina
T	Treonina
U	Selenocisteína
V	Valina
W	Triptófano
Y	Tirosina
Z	Ácido glutámico o Glutamina
X	cualquiera
*	parada de traducción
-	hueco (gap) de longitud indeterminada

## 7. Software para folding

Se emplearán las siguientes librerías:

- **RNAFold**

Básicamente lee secuencias de RNA de entrada estándar y calcula el mínimo de energía libre (MFE), la función de partición (pf) y una matriz de probabilidad en base al apareamiento. Devuelve como salida la estructura mfe en notación soporte (empleando “()”), su energía, la energía libre del conjunto termodinámico y la frecuencia de la estructura mfe. También produce archivos PostScript con parcelas de la gráfica

resultante estructura secundaria y un “gráfico de puntos” de la matriz de apareamiento de bases. Para comprender mejor su comportamiento y método consultar [12].

Está disponible para su descarga en  
<http://mfold.rna.albany.edu/?q=DINAMelt/software>.

#### ■ UNAFold

Inspirado en el famoso “mfold”, el paquete de software *UNAFold*[11] es un conjunto integrado de programas (empleando programación dinámica) que simulan el folding, la hibridación y la vías de fusión para una o dos secuencias de cadena simple de ácido nucleico. El paquete predice el folding para RNA de cadena simple o el DNA a través de la combinación de minimización de la energía libre, los cálculos de función de partición y muestreo estocástico. El nombre se deriva de “Folding Unificado de ácido nucleico”. Está disponible para su descarga en <http://www.bioinfo.rpi.edu/applications/hybrid/download.php>.

Se deberán agregar el backends *UNAFold* en *fideo*.

## 8. Software para hubridar

Se emplearán las siguientes librerías:

#### ■ RNAup

Calcula la termodinámica de las interacciones RNA-RNA, por descomposición de las uniones a través de dos etapas. La termodinámica de tales interacciones RNA-RNA puede ser entendida como la suma de dos contribuciones de energía: la energía necesaria para “abrir” el sitio de unión, y la energía obtenida de la hibridación.

En primer lugar se calcula la probabilidad de que un potencial sitio de unión siga siendo desapareado, lo que es equivalente a calcular la energía libre necesaria para romper las uniones. En segundo lugar, este cálculo se combina con la energía de interacción para obtener la energía total de unión.

*RNAup* proporciona dos modos: por defecto calcula accesibilidad (en términos de energía libre para romper uniones, de longitud 4) y muestra la región de mayor accesibilidad y la energía libre necesaria para su apertura. En modo interacción, se calcula la interacción entre dos RNA. Este modo se activa si la entrada se compone de dos secuencias concatenadas por un “&” o si se emplea el flag `-X[pf]` o `-b`.

La energía libre “dG” se divide en sus componentes “dGint” (energía de interacción) y “dGuL” (energía de apertura).

*RNAup* está incluido en el paquete de Vienna RNA[12]. La última versión corresponde a la 2.0.7 y está disponible para su descarga en <http://www.tbi.univie.ac.at/ivo/RNA/>.

#### ■ RNACofold

Funciona como *RNAfold*, pero permite especificar dos secuencias de RNA que formará una estructura de dímero<sup>10</sup>.

Las secuencias de RNA se leen por entrada estándar, cada línea de entrada corresponde a una secuencia, a excepción de las líneas que comienzan con “;”, que contiene el nombre de la siguiente secuencia. Para calcular la estructura híbrida de dos moléculas, las dos secuencias deben ser concatenadas usando el carácter “&” como separador.

Al igual que *RNAup*, *RNACofold* está incluido en el paquete de Vienna RNA[12].

#### ■ RNAduplex

Básicamente lee dos secuencias de RNA por entrada estándar o desde un archivo y calcula las estructuras secundarias óptimas y subóptimas para su hibridación. El cálculo se simplifica al permitir sólo pares de bases inter-moleculares (corresponde a un caso específico de *RNAcofold*).

La estructura calculada óptima y subóptima se retornan por salida estándar, una por línea. Cada una de ellas consiste en: la estructura de soporte en formato de punto con una “y” que separan las dos hebras, el rango de la estructura en las dos secuencias en el formato “from,to : from,to”, la energía de estructura dúplex en kcal/mol.

El formato es especialmente útil para el cálculo de la estructura híbrida entre una secuencia pequeña y una secuencia larga.

Al igual que los dos anteriores, *RNAduplex* está incluido en el paquete de Vienna RNA[12].

- **IntaRNA** *IntaRNA*[13] predice interacciones entre dos moléculas de RNA, por ejemplo, un RNA no codificante (e.g ncRNA) y un mRNA. El scoring se basa en la energía combinada de interacción que resulta de la

---

<sup>10</sup>Complejo macromolecular formado por dos macromoléculas, como proteínas y ácidos nucleicos, usualmente mediante enlaces no covalentes.

suma de la energía libre de hibridación y la energía libre necesaria para la fabricación de los sitios de interacción accesible de ambas moléculas.

La interacción tiene que contener la semilla de una interacción, es decir, una región (casi) perfecta de complementariedad para facilitar el inicio de la interacción. Las características de esta región de semillas son definidas por el usuario.

La accesibilidad se define como la energía libre necesaria para desplegar el sitio de interacción en cada molécula.

*IntaRNA* requiere del paquete de Vienna RNA (versiones 1.8.2 - 1.8.5). La versión 1.8.5 puede descargarse de <http://www.bioinf.uni-freiburg.de/Software/index.html?en#IntaRNA-download>

Se deberá crear una interface en *fideo* para la hibridización, de forma similar al folding.

## 9. Bibliotecas existentes a utilizar

### 9.1. MiLi

*MiLi*<sup>11</sup> es una colección de pequeñas y útiles librerías desarrolladas en el lenguaje de programación C++ por **FuDePAN**, compuestas únicamente por cabeceras (*headers*). No requiere instalación para su uso y ofrece soluciones simples para problemas sencillos.

Provee varias funcionalidades mediante archivos cabecera, conocidos en el ámbito de C/C++ como archivos con extensión “.h”.

Dentro de las diversas funcionalidades podemos encontrar:

- **binary-streams:** permite serializar diferentes tipos de datos dentro de un único objeto utilizando los operadores de stream ( $\ll$  y  $\gg$ ). Hay dos maneras de utilizar esta librería:
  1. Empaquetar datos dentro de un objeto de salida (bostream) utilizando el operador  $\ll$ .
  2. Extraer datos desde un objeto de entrada (bistream) utilizando el operador  $\gg$ .
- **container-utils:** esta biblioteca provee un conjunto de funciones, optimizadas para cada tipo de contenedor STL:

---

<sup>11</sup>MiLi: [mili.googlecode.com](http://mili.googlecode.com).

- `find(container, element)`.
- `find(container, element, nothrow)`.
- `contains(container, element)`.
- `insert into(container, element)`.
- `copy container(from, to)`.

Adicionalmente, esta biblioteca provee las siguientes clases:

- `ContainerAdapter<T>`.
- `ContainerAdapterImpl<T, ContainerType>`.

Estos container adapters son una herramienta para lidiar con diferentes contenedores STL de manera homogénea, sin necesidad de conocer el tipo de contenedor que es (vector, list, map, set). Los usuarios pueden invocar al método `INSERT (CONST T&)` para insertar un elemento de tipo `T` sin la necesidad de saber el tipo del contenedor.

- **generic\_exception:** ofrece una implementación de excepciones genéricas a partir de las cuales los desarrolladores pueden crear sus propias excepciones para problemas específicos de una manera muy simple.

## 9.2. FuD

*FuD*<sup>12</sup> es un framework para automatizar la implementación de aplicaciones distribuidas. Este framework es un sistema separado en las capas de *aplicación*, *administración* y *distribución* combinando los conceptos de cliente-servidor y Divide & Conquer. Tanto el cliente como el servidor se encuentran organizados en estos tres niveles separados, cada uno de ellos con una única responsabilidad bien definida. La comunicación entre los diferentes niveles se encuentra estrictamente limitada, es decir, por cada nivel existe un único punto de comunicación ya sea para comunicarse con la capa superior o con la inferior. Cuando un mensaje es creado, éste debe atravesar las diferentes capas comenzando desde la de nivel más alto hacia la capa inferior, y luego recorrer en sentido contrario las capas del lado receptor.

### 9.2.1. Diseño de original de FuD

- **APPLICATION LAYER (L3):** proporciona los componentes que contienen todos los aspectos del dominio del problema a resolver. Estos aspectos incluyen todas las definiciones de los datos usados y su manipulación

---

<sup>12</sup>FuDePAN Ubiquitous Distribution: [fud.googlecode.com](http://fud.googlecode.com).



correspondiente, como así también todos los algoritmos relevantes para la solución al problema en general.

Esta capa no es considerada como parte de *FuD*. La implementación de una aplicación que usa la librería, no es parte de la librería.

Es necesario que del lado del servidor se implemente la aplicación principal, la cual hará uso de una simple interfaz en la abstracción de un trabajo distribuible permitiendo así codificar la estrategia de distribución de trabajos. Del lado cliente, solo se necesita implementar los métodos encargados de realizar las computaciones indicadas por una unidad de trabajo.

- **JOB MANAGEMENT LAYER (L2):** permite manejar los trabajos que se desean distribuir como así también generar las unidades de trabajo que serán entregadas a los clientes para su procesamiento. Estas unidades de trabajo llegan a su cliente correspondiente gracias a la capa más baja, encargada de la distribución. Una vez finalizado el procesamiento, se informa que todo ha terminado y otorga los resultados a la capa superior.
- **DISTRIBUTING MIDDLEWARE LAYER (L1):** constituye un esquema de administración de clientes en particular. Tanto del lado cliente como del servidor, la parte fija está dada por interfaces.

Las implementaciones concretas de este nivel son variables y están determinadas por el middleware a utilizar, por ejemplo *BOOST.ASIO*<sup>13</sup>, *MPI*<sup>14</sup> o *BOINC*<sup>15</sup>.

Actualmente, *FuD* cuenta con dos capas más, conocidas como *FuD-RecAbs* y *FuD-CombEng*.

### 9.3. BioPP

*BioPP*<sup>16</sup> es una biblioteca C++ para Biología Molecular. La misma proveerá las diversas estructuras de datos y métodos para manipular las secuencias de nucleótidos con las que se trabajará.

---

<sup>13</sup>[http://www.boost.org/doc/libs/1\\_4\\_00/doc/html/boost\\_asio.html](http://www.boost.org/doc/libs/1_4_00/doc/html/boost_asio.html)

<sup>14</sup><http://www.mcs.anl.gov/research/projects/mpi/>

<sup>15</sup><http://boinc.berkeley.edu/>

<sup>16</sup>[biopp.googlecode.com](http://biopp.googlecode.com).

## 9.4. Biopp-filer

*Biopp-filer*<sup>17</sup> es una librería de persistencia para *Biopp*. Esta librería será utilizada para leer las secuencias en formato FASTA.

## 9.5. Odf-gen

*Odf-gen*<sup>18</sup> es una librería que ofrecerá funcionalidades que permiten generar archivos OpenDocument. Soporta diferentes tipos de archivos tales como:

- Hojas de cálculo. Compatible con OpenOffice Calc.
- Características compatibles: tablas, gráficos, gráficos de automóviles.
- APIs disponibles: C++, Python.

Será utilizada para generar un archivo en el cual se almacenarán las tablas comparativas. Alternativamente, las tablas respetarán el formato CSV.

## 9.6. Fideo

*Fideo*<sup>19</sup> es una librería que proveerá, entre otras, las funcionalidades necesarias para obtener la energía libre de una secuencia de nucleótidos. Casi la totalidad de su código proviene del proyecto VAC-O<sup>20</sup>.

Esta librería será utilizada para la invocación externa de los programas de cálculo de estructura secundaria (folding e hibridación). Provee la abstracción necesaria para poder utilizar indistintamente cualquiera de ellos, ya sea *mfold*, *unafold*, *vienna*, entre otros.

# 10. Conclusiones

Como resultado de esta etapa de análisis se obtuvieron los datos mínimos necesarios para realizar las futuras pruebas del producto. Además se identificó la necesidad de extender la librería *fideo*. Por un lado, la inclusión de los backends *mfold*[9][10] y *unafold*[11], y por el otro, la inclusión de interfaces de hibridación de secuencias.

Asimismo, se observó que será importante almacenar el folding de los RNA<sub>m</sub>, debido a que podrían utilizarse como input en otra iteración.

---

<sup>17</sup>[biopp-filer.googlecode.com](http://biopp-filer.googlecode.com).

<sup>18</sup>[odf-gen.googlecode.com](http://odf-gen.googlecode.com).

<sup>19</sup>[fideo.googlecode.com](http://fideo.googlecode.com).

<sup>20</sup>[vac-o.googlecode.com](http://vac-o.googlecode.com)

## Referencias

- [1] “FASTA” [http://en.wikipedia.org/wiki/FASTA\\_format](http://en.wikipedia.org/wiki/FASTA_format)
- [2] “A Framework for Small Distributed Projects and a Protein Clusterer Application” BISET, GUILLERMO, 3 de diciembre de 2009.
- [3] “The extent of codon usage bias in human RNA viruses and its evolutionary origin” GARETH M. JENKINS AND EDWARD C. HOLMES, 2003.
- [4] “Designing genes for successful protein expression.” WELCH, ET AL., Methods Enzymol 2011 498:43-66.
- [5] “GeneDesign” [http://en.wikipedia.org/wiki/Gene\\_Designer](http://en.wikipedia.org/wiki/Gene_Designer)
- [6] “GeneDesign” <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC1457031/?tool=pubmed>
- [7] “GeneDesign 3.0 is an updated synthetic biology toolkit” SARAH M. RICHARDSON, PAUL W. NUNLEY, ROBERT M. YARRINGTON, JEF D. BOEKE AND JOEL S. BADER Nucleic Acids Research Advance Access published March 8, 2010
- [8] “Computational Methods for RNA Secondary Structure” ZUKER, MICHAEL, June 8, 2006.
- [9] “The use of dynamic programming algorithms in RNA secondary structure prediction.” M. ZUKER In M. S. Waterman, editor, Mathematical Methods for DNA Sequences. Boca Raton, FL, CRC Press, 1989.
- [10] “Mfold web server for nucleic acid folding and hybridization prediction.” M. ZUKER, Nucleic Acids Res., 31(13):3406-3415, 2003.
- [11] “UNAFold: Software for Nucleic Acid Folding and Hybridization. In Data, Sequence Analysis, and Evolution” N. R. MARKHAM & M. ZUKER., Bioinformatics: Volume 2, Humana Press Inc., 2008.
- [12] “Fast Folding and Comparison of RNA Secondary Structures (The Vienna RNA Package)” I.L. HOFACKER, W. FONTANA, P.F. STADLER, S. BONHOEFFER, M. TACKER, P. SCHUSTER. 1994. Monatshefte f. Chemie 125: 167-188.
- [13] “IntaRNA: efficient prediction of bacterial sRNA targets incorporating target site accessibility and seed regions.” A. BUSCH, A. RICHTER & R. BAC-KOFEN. Bioinformatics Group, Albert-Ludwigs-University Freiburg, Georges-Koehler-Allee 106, Freiburg D-79110, Germany.