RNAemo

Especificación de Diseño de Software

Franco Gaspar Riberi

7 de noviembre de 2012

Índice

1. Introducción

1.1. Propósito

El propósito de este documento es la especificación de diseño de software correspondiente a la primera versión del producto ***RNAemo***.

La confección de este documento se contextualiza dentro del desarrollo de la tesis de grado de la carrera Licenciatura en Ciencias de la Computación de la UNRC, ***RNAemo***, a cargo de Franco Gaspar Riberi, con la dirección de la Lic. Laura Tardivo (UNRC) y las colaboraciones de Daniel Gutson, y el Dr. Roberto Daniel Rabinovich (**FuDePAN**).

El documento esta dirigido a las personas involucradas en el desarrollo de la tesis como así también a todos los colaboradores de **FuDePAN** que eventualmente podrían participar en las etapas de desarrollo y mantenimiento del software.

1.2. Descripción general del documento

En la sección 2 se mencionan los objetivos, la metodología adoptada y las dependencias del diseño.

En la sección **??** se exhibe la arquitectura general del sistema con sus principales componentes e interacciones.

En la sección **??** se presenta el diseño de alto nivel del sistema, sus interfaces y paquetes principales.

En la sección **??** se observa el diseño de bajo nivel del sistema, esto involucra las clases concretas y sus relaciones para cada paquete.

2. Consideraciones de diseño

2.1. Objetivos

Se pretende lograr un diseño del sistema que cumpla con los principios fundamentales del diseño orientado a objetos, comúnmente conocidos por el acrónimo **"SOLID"** [1].

En particular, se pretenden respetar los principios **SRP** (*Single Responsability Principle*), **OCP** (*Open-Closed Principle*) y **DIP** (*Dependency Inversion Principle*) debido a su importancia para obtener un sistema fácilmente extensible y configurable con el fin de satisfacer las necesidades de los usuarios.

2.2. Metodología

La metodología empleada para realizar el análisis y descripción del diseño se denomina *"Diseño dirigido por responsabilidades"*[2].

Esta técnica se enfoca en *qué* acciones (responsabilidades) deben ser cubiertas por el sistema y que objetos serán los responsables de llevarlas a cabo. *Cómo* se realizara cada acción, queda en un segunda plano.

2.3. Herramientas y convenciones

Se utiliza UML[3] como lenguaje de modelado, ArgoUML[4] y http://www.websequencediagrams.com como herramientas para la confección de diagramas, y Dia[5] para la edición de diagramas de propósito general. Además se adopta la convención de nombrar a las interfaces anteponiendo una letra *"I"* al nombre de la clase concreta que la implementa.
Por ejemplo, interface: *"IPersona"* → clase concreta: *"Persona"*.

3. Diseño de alto nivel

La arquitectura del sistema y la interacción entre los diversos componentes que la conforman se exhiben en la figura 1. En las subseccones siguientes se describe brevemente cada uno de ellos.

3.1. Componetes del sistema

- **Main:** corresponde al módulo principal en términos de ejecución del sistema.
- **MasterOfPuppets (MOP):** comprende la inicialización e invocación de los demás componentes. Coordina y gestiona la mayoría de las interacciones entre módulos.
- **GeneDesign:** representa el componente encargado de generar secuencias humanizadas. Dada una secuencia original, genera la secuencia humanizada correspondiente. Dicho módulo es externo a este desarrollo, y para ello se emplea el software *GeneDesign*.
- **OutputsGenerator:** comprende la generación de archivos. Se crean tanto archivos como RNA_m se analizen.
- **TableGenerator:** este componente es el encargado de rellenar las tablas. Para ello, realiza el matching por complemento y el cálculo de score entre secuencias de RNA_m y small-RNA_s.

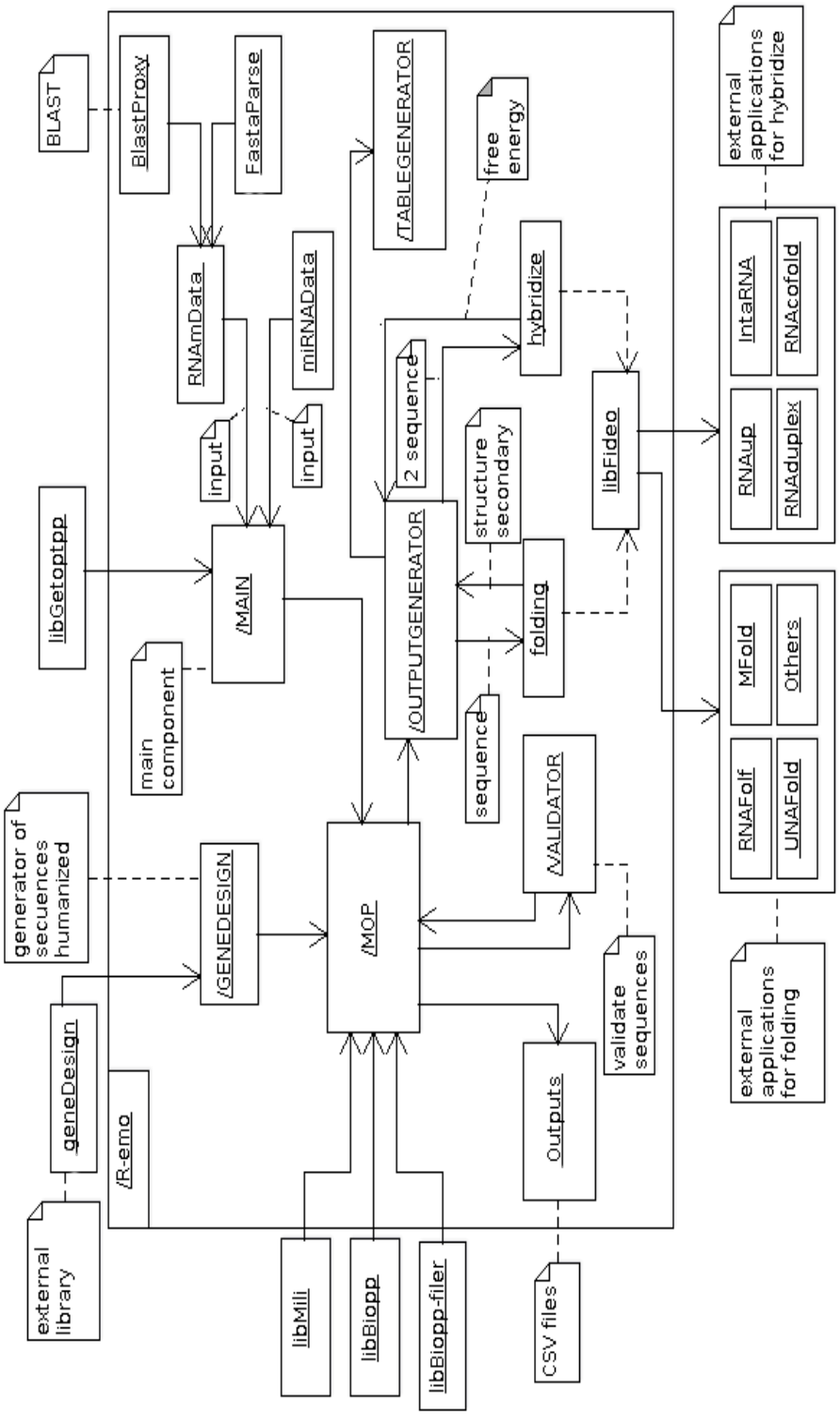

Figura 1: UML - Arquitectura del Sistema

- **Validator:** encargado de controlar la calidad de las secuencias. Es el componente encargado de decidir, si una secuencia se encuentra en el marco de lectura correcto o no.

3.2. Librerias externas

- **Mili:** corresponde a una colección de funciones de C++, para resolver detalles de implementación.
- **fideo:** corresponde a una librería parcialmente ya implementada. Provee al sistema la funcionalidad de *"folding"* e hibridación.
- **Biopp:** biblioteca C++ para Biología Molecular. Permite la manipulación de secuencias de ácidos nucleicos.
- **Biopp-filer:** para la lectura de secuencias en formato FASTA.
- **getoptpp** para facilitar el manejo de entrada estándar.

3.3. Interacción entre los componentes

En la figura 4 se presenta la interacción entre componentes a través de un diagrama de secuencias. Se empleó este tipo de diagrama ya que permite capturar en forma óptima la secuencia temporal de las interacciones entre los componente.

4. Diseño de medio nivel

4.1. Estructura de paquetes

La figura 3 muestra los diferentes paquetes que conforman el sistema.

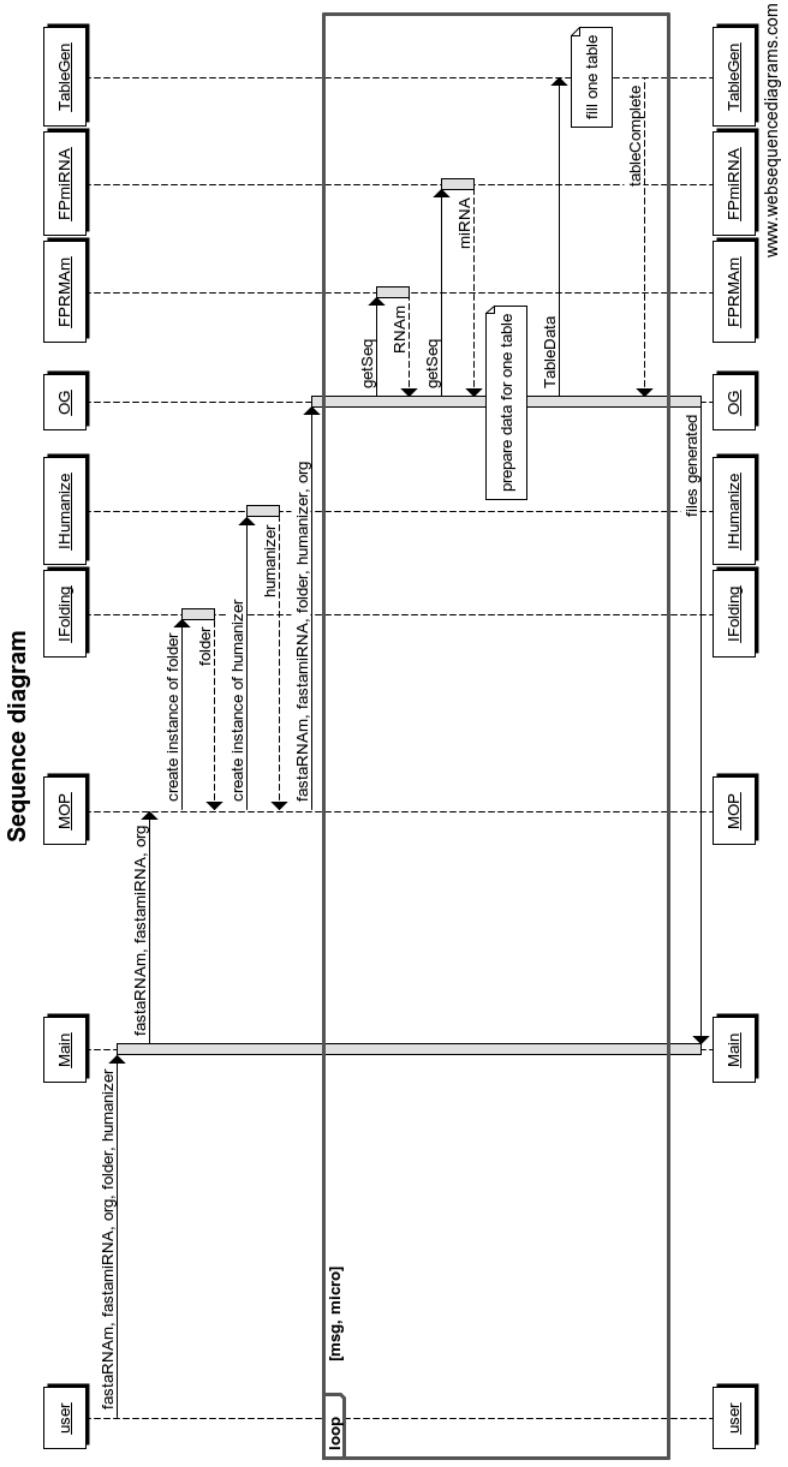

Figura 2: UML - Diagrama de secuencia

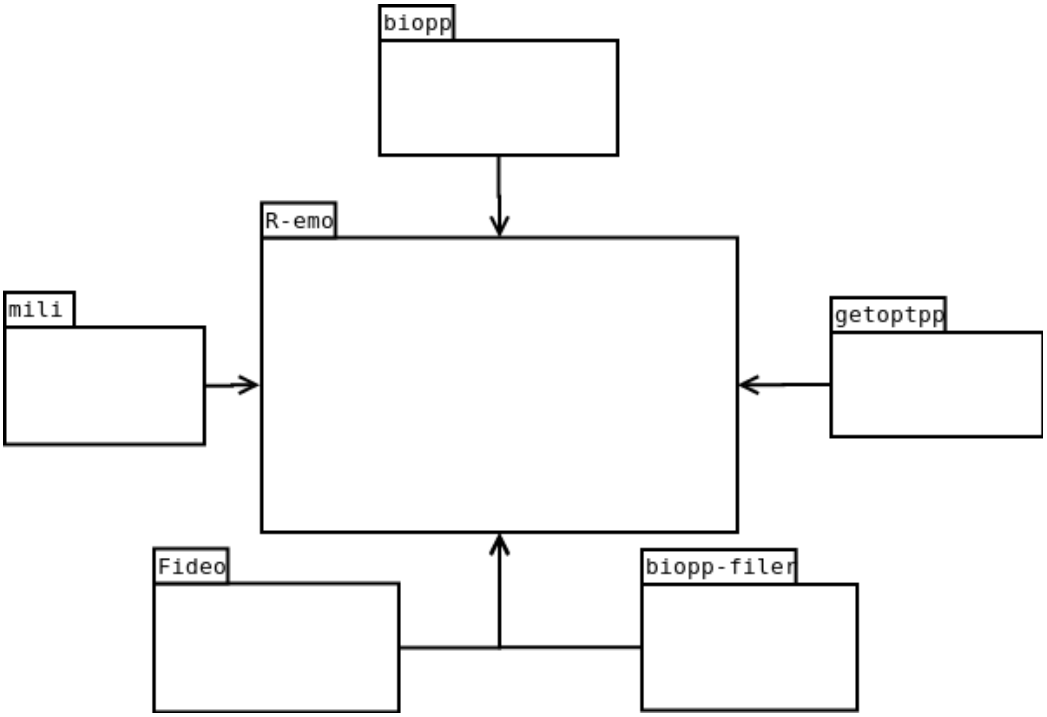

Figura 3: UML - Paquetes

4.2. Diagrama de clases

4.3. Clases

4.3.1. Interfaces

- **fideo:**
  - *IFold:* describe de manera transparente el servicios de folding.
  - *IHybridize:* describe de manera transparente el servicios de hibridación.
- **ICodonUsageModifier:** describe de manera transparente el servicios de humanización de secuencias.

4.3.2. Clases concretas

- **fideo:**
  - *RNAFold y UNAFold:* backends específicos para folding.

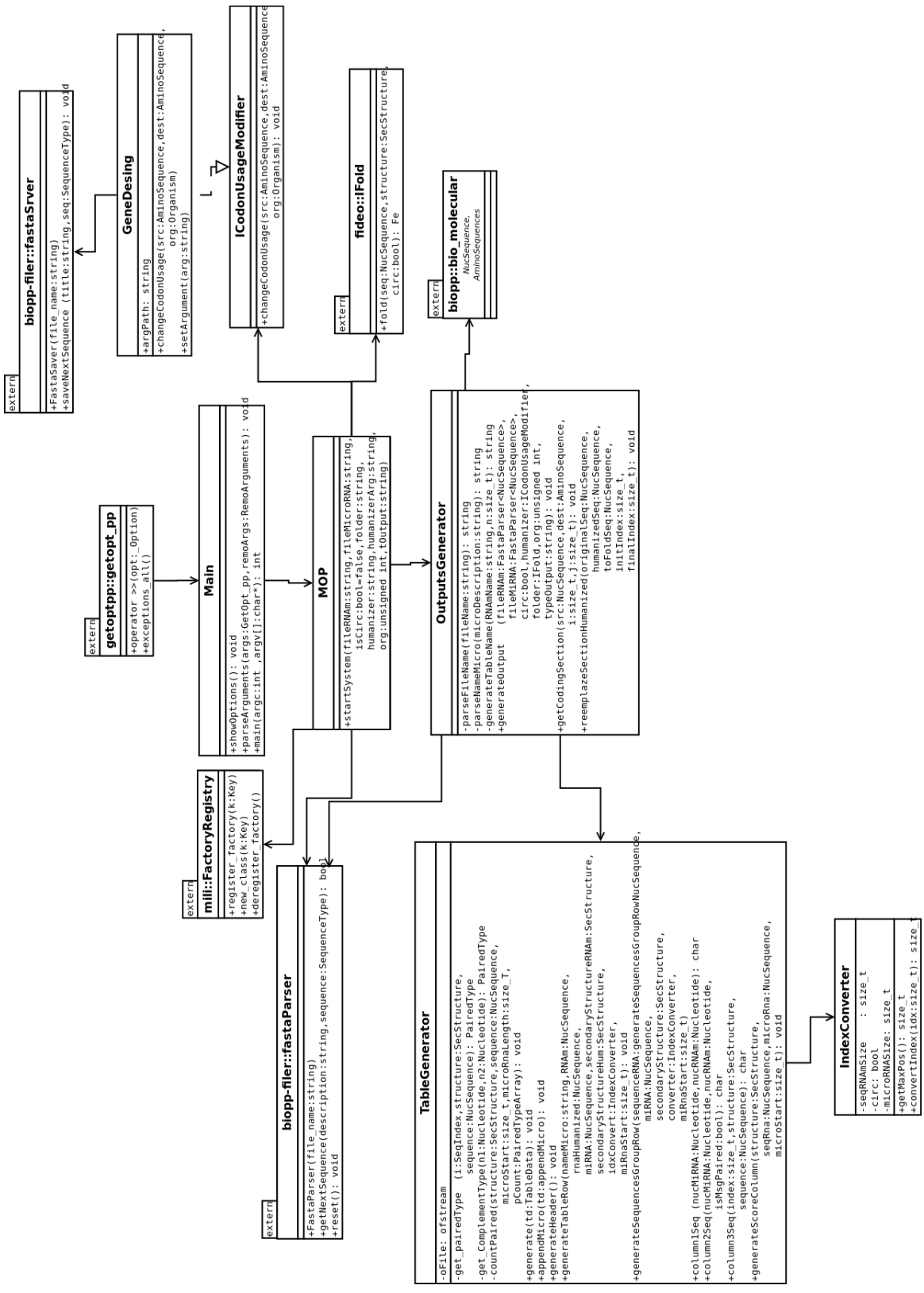

Figura 4: UML - Diagrama de clases

  - *RNAup, RNAcofold, RNAduplex e IntaRNA:* backends específicos para hibridar.

- **GeneDesign:** software específico (geneDesign) para humanizar.

4.4. Interfaces - Responsabilidades - Colaboradores

- **Module:** MOP
  Target: Gestionar la iteración de módulos.
  Responsibilities: Interconectar partes del sistema.
  Services:

  - Creación de objetos necesarios.
  - Invocación a humanización.
  - Invocación a folding secuencia original.
  - Invocación a folding secuencia humanizada.
  - Invocación al generador de archivos.

  Collaborations: GeneDesign, OutputsGenerator , TableGenerator.

- **Module:** Main.
  Target: Lanzar de aplicación.
  Responsibilities: Lanzador de aplicación.
  Services:

  - Mostrar opciones de uso.
  - Parsear los argumentos de entrada.
  - Capturar las excepciones ocurridas durante la ejecución.

  Collaborations: MOP.

- **Module:** GeneDesign.
  Target: Humanizar secuencias.
  Responsibilities: Humanizar una secuencia de RNA.
  Services:

  - Setear el path del software humanizador.
  - Humanizar una secuencia.

Collaborations: -

- **Module:** OutputsGenerator.
  Target: Generador de archivos.
  Responsibilities: Construir un archivo de salida.
  Services:
  - Generar el nombre de cada archivo.
  - Parsear la descripcion de cada Fasta.
  - Generar un archivo por cada RNAm-miRNA.

  Collaborations: TableGenerator.

- **Module:** TableGenerator.
  Target: Construir tablas.
  Responsibilities: Rellenar una tabla.
  Services:
  - Generar el encabezado de una tabla.
  - Generar fila de scores.
  - Generar fila de matching (complemento, enmascarados).
  - Escribir una linea de la tabla.
  - Calcular cada columna de la tabla.
  - Contabilización de nucleótidos apareados.
  - Escritura en un archivo.

  Collaborations: -

5. Fud-agnostic

El componente principal MasterOfPuppets será el encargado de la generación de las tablas. Para ello, se implementará un método generateTable, el cual tomará como parámetro el RNA mensajero, el RNA humanizado y un $_{mi}$RNA. El componente principal realizará una doble iteración anidad entre RNA$_m$ y $_{mi}$RNA. Es decir, por cada RNA$_m$ se recorrerá cada secuencia de $_{mi}$RNA invocando al método ya mencionado.

Esto permitirá un paralelización trivial empleando *FuD*.

generateTable (const Sequence& rna_original, const Sequence& rna_humanized, const Sequence& mi_rna)

Referencias

[1] *"Design Principles and Design Patterns."* Robert C. Martin, 2000. http://www.objectmentor.com

[2] *"Object Design: Roles, Responsibilities."* Rebecca Wirfs-Brock and Alan McKean and Collaborations, Addison-Wesley, 2003.

[3] *"Unified Modeling Language."* http://www.uml.org/

[4] *"ArgoUML."* http://argouml.tigris.org/

[5] *"Dia."* http://live.gnome.org/Dia