UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

ESCOLA TÉCNICA SUPERIOR DE ENXEÑARÍA

FlinkBWA

Uso de tecnologías *Big Data* para el alineamiento de secuencias genéticas

Autora:
Silvia Rodríguez Alcaraz

Tutor:
Juan Carlos Pichel Campos
Cotutor:
José Manuel Abuín Mosquera

Grado en Ingeniería Informática

Junio 2019

Trabajo de Fin de Grado presentado en la Escuela Técnica Superior de Ingeniería de la Universidad de Santiago de Compostela para la obtención del Grado en Ingeniería Informática

D. Juan Carlos Pichel Campos, Profesor del Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela, y **D. José Manuel Abuín Mosquera**, Profesor del Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela,

INFORMAN:

Que la presente memoria, titulada *FlinkBWA: uso de tecnologías Big Data para el alineamiento de secuencias genéticas*, presentada por **D.ª Silvia Rodríguez Alcaraz** para superar los créditos correspondientes al Trabajo de Fin de Grado de la titulación del Grado en Ingeniería Informática, se realizó bajo nuestra dirección en el Departamento de Electrónica y Computación de la Universidad de Santiago de Compostela.

Y para que así conste a los efectos oportunos, expiden el presente informe en Santiago de Compostela, a (Data)

O director, O codirector, O alumno,

Juan C. Pichel Campos José M. Abuín Mosquera Silvia Rodríguez Alcaraz

Agradecimientos

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introdución

1.1. Contextualización

Los continuos avances dentro del sector tecnológico han permitido que, en la actualidad, cualquier persona con acceso a los medios informáticos adecuados sea capaz de generar una importante cantidad de datos diariamente. Concretamente, el estudio anual del IDC (*International Data Corporation*) del año 2012 [1] revelaba que en 2020 cada persona generaría aproximadamente más de 5200 *gigabytes* de información única.

Consecuencia de este notorio progreso, hoy por hoy, los medios digitales son los prioritarios a la hora de salvaguardar información, pero este hecho trae consigo un reto importante: ¿cómo gestionar, almacenar y analizar de manera eficiente tal cantidad de información? En este punto es donde entran en juego las herramientas *Big Data* que, simplificando, se encargan de facilitar el tratamiento de este tipo de datos.

Por su parte, las tecnologías de secuenciación de próxima generación (NGS) también han dado lugar a una gran cantidad de datos genómicos que precisan ser analizados e interpretados. Una de las fases más costosas del análisis es la alineación de secuencias de ADN, que supone el mapeo de miles de millones de pequeñas secuencias sobre un genoma de referencia. Es por esto que los alineadores de vanguardia emplean estrategias de paralelización con el fin de disipar este problema, pero las soluciones suelen mostrar una implementación compleja además de una baja escalabilidad.

1.2. Motivación

El *Burrows-Wheeler Aligner* (BWA) [2] es un *software* empleado para mapear secuencias de baja divergencia contra un genoma de referencia grande, como el

humano. Se trata de uno de los alineadores de secuencias genéticas más ampliamente utilizados, pero sufre el mismo problema que el resto de alineadores al trabajar con grandes volúmenes de datos.

El CiTiUS (Centro Singular de Investigación en Tecnoloxías da Información) cuenta con un proyecto que aplica la plataforma *Big Data Apache Spark* para tratar de mejorar el rendimiento del BWA, *SparkBWA* [4]. Pero *Spark* únicamente está orientado al procesamiento de datos por lotes, siendo más común en las aplicaciones actuales tratar con flujos de datos.

El presente trabajo consiste en usar la tecnología *Apache Flink* [3], orientada al procesamiento de datos tanto por lotes como en flujos, para la paralelización del BWA, con el objetivo de mejorar el rendimiento del mismo en términos de eficiencia.

1.3. Objetivos

El objetivo principal de dicho trabajo es usar la plataforma *Big Data Apache Flink* para tratar de mejorar el rendimiento del alineador *Burrows-Wheeler Aligner* (BWA). Por tanto, para lograr este fin último, el trabajo englobaría el siguiente conjunto de objetivos específicos:

- Estudio previo del estado del arte que justifique el uso de tecnologías *Big Data* para tratar de mejorar los resultados temporales de BWA.
- Formación en tecnologías *Big Data* y en aplicaciones de alineamiento genético.
- Diseño e implementación del nuevo alineador BWA paralelo que haga uso del *framework Apache Flink*.
- Análisis del rendimiento de la aplicación en un *cluster*.

1.4. Organización de la memoria

La estructura que sigue la memoria es la siguiente:

- **Capítulo 1. Introducción:** presente capítulo donde se contextualiza el proyecto, se presenta la motivación del mismo y se muestra la estructura de la memoria.
- **Capítulo 2. Gestión del proyecto:** capítulo dedicado a la gestión del proyecto. Incluye la gestión de: alcance, configuración, tiempo, riesgos, costes, comunicaciones y justifica la metodología de desarrollo escogida.

- **Capítulo 3. Especificación de requisitos:** especifica los requisitos que debe cumplir el proyecto.
- **Capítulo 4. Análisis:** incluye una introducción a las aplicaciones de alineamiento genético, una descripción del BWA, un breve estudio del arte de las tecnologías *Big Data* y una justificación de *Flink* como plataforma escogida para el desarrollo.
- **Capítulo 5. Diseño e implementación:** explicación del diseño de la aplicación, arquitectura, implementación y funcionamiento de la misma.
- **Capítulo 6. Pruebas de rendimiento:** comprende la descripción de las pruebas de rendimiento y de la infraestructura utilizada para su realización además de sus resultados y su pertinente interpretación.
- **Capítulo 7. Conclusiones:** menciona y desarrolla las conclusiones obtenidas tras la realización del proyecto.

Capítulo 2

Gestión del proyecto

Este capítulo incluye todos los contenidos vinculados a la gestión del proyecto. Gestionar un proyecto implica estudiar y elaborar una planificación, organización y control de los recursos disponibles para definir y alcanzar una serie de objetivos concretos.

Tomando como base los principios de la 5ª ed. del PMBOK [5] se abordan: la gestión del alcance, la gestión de la configuración, la gestión del tiempo, la gestión de las comunicaciones, la gestión de riesgos y la gestión de costes del proyecto.

2.1. Gestión del alcance

La Gestión del Alcance comprende el conjunto de procesos necesarios para asegurar que el proyecto incluye todos los contenidos necesarios para su finalización exitosa. Básicamente, el objetivo principal es especificar qué debe incluir el proyecto y qué no de forma detallada.

2.1.1. Descripción del alcance del proyecto

Este proyecto pretende mejorar la eficiencia temporal del alineador de secuencias genéticas BWA mediante la incorporación de la plataforma *Big Data Apache Flink*.

Para lograr el fin descrito anteriormente es necesario: comprender el funcionamiento del BWA y de *Flink*, además de sus requerimientos; diseñar la nueva aplicación paralela; implementar la aplicación planteada; y, finalmente, estudiar el rendimiento de la misma en un *cluster*.

2.1.2. Criterios de aceptación

Los criterios de aceptación de este proyecto serían la obtención de: una aplicación paralela del BWA que haga uso de *Apache Flink*, un estudio del rendimiento de la misma que permita evaluar la diferencia en términos de eficiencia con la aplicación original y una memoria que detalle todo el desarrollo del proyecto de manera adecuada.

2.1.3. Entregables del proyecto

Una vez finalizado el proyecto, los contenidos a entregar son los siguientes:

- **Soporte digital con el *software FlinkBWA*:** soporte digital que contiene la aplicación desarrollada junto con la documentación adicional necesaria y la memoria.
- **Memoria del proyecto:** se trata del presente documento, donde se describe detalladamente el proceso llevado a cabo para la finalización exitosa del proyecto. Se deben entregar 3 copias de dicho documento.

2.1.4. Restricciones del proyecto

La única restricción a tener en cuenta para este proyecto es temporal e implica lograr la finalización del mismo en aproximadamente 402 horas de trabajo autónomo. Además, todos los contenidos indicados en 2.1.3 deben ser entregados en la Administración de la ETSE antes de las 11:00h del 1 de julio de 2019, que es el limite oficial establecido por la escuela [6] para realizar la lectura del trabajo en la convocatoria de julio de 2019, concretamente entre los días 17 y 18 de julio.

2.1.5. Metodología de desarrollo

En el marco de un proyecto de *software* es fundamental decidir la metodología que se va a seguir, ya que determina cómo estructurar, planificar y controlar el proceso de desarrollo. La elección de una u otra metodología varía según el caso y depende de las características del proyecto que se vaya a realizar.

Concretamente, este proyecto presenta dos factores importantes a la hora de escoger la metodología a seguir para su desarrollo:

- La aplicación a desarrollar presenta un caracter innovador ya que utilizará la plataforma *Big Data Apache Flink* y, además, está orientada a un ámbito muy específico: el alineamiento de secuencias genéticas.
- El desconocimiento de la tecnología a utilizar y la falta de formación sobre aplicaciones de alineamiento de secuencias genéticas de la desarrolladora.

Consecuencia de dos puntos descritos anteriormente probablemente la aplicación experimente cambios bastante continuos durante el desarrollo, conforme va aumentando la formación de la desarrolladora y en función de las sugerencias y pautas que marquen los expertos de validar el trabajo, en este caso el tutor y cotutor del proyecto.

Por este motivo, para el presente trabajo, se ha decidido seguir una metodología ágil. Resumiendo los 12 principios del *Manifiesto Ágil* [7], este tipo de metodologías se centran en:

- Frecuente realización de entregas útiles.
- Continua comunicación y colaboración entre responsables y desarrolladores.
- Auto-gestión del equipo de trabajo, reflexionando a intervalos regulares sobre su propia efectividad y perfeccionando su comportamiento como consecuencia.
- Atención continua a la excelencia técnica y al buen diseño.
- Capacidad para mantener un ritmo constante de forma indefinida.

La metodología ágil escogida en este caso es *Scrum*, ya que presenta un enfoque iterativo e incremental de desarrollo. De esta forma, aporta control y flexibilidad para realizar cambios en situaciones de incertidumbre o falta de conocimiento así como para predecir y/o evitar posibles riesgos.

Scrum

A continuación, se describirán los aspectos más relevantes de la metodología escogida, haciendo uso de los conocimientos expuestos en la Guía de *Scrum* [8].

Para comenzar, *Scrum* se basa en el empirismo y, por tanto, sostiene que las decisiones han de basarse en el conocimiento que proviene de la experiencia previa. Además, como ya se mencionó, aplica un enfoque iterativo e incremental, con el que se pretende optimizar la predictibilidad, así como el control de los riesgos. Podría decirse que sus tres pilares fundamentales son: transparencia, inspección y adaptación. Resumiendo: debe haber una comunicación fluida dentro del equipo de trabajo, revisión continua de los avances y aplicación de cambios en caso de encontrar un aspecto que se desvíe de los limites aceptables.

Los equipos de trabajo que presenta esta metodología son auto-organizados y multifunción. La entregan de productos es iterativa e incremental, aumentando las oportunidades de recibir retroalimentación. Los principales roles del *Scrum team* son:

- **Dueño del producto (*Product Owner*):** responsable de maximizar el valor del producto y del trabajo del Equipo de Desarrollo.
- **Equipo de desarrollo *(Development Team)*:** se encarga de realizar el trabajo correspondiente a cada incremento.
- ***Scrum Master*:** líder del proyecto y encargado de asegurar que se está siguiendo correctamente la metodología.

En cuanto al ciclo de trabajo propuesto por *Scrum*, existen una serie de eventos predefinidos que buscan crear cierta regularidad y evitar reuniones no previstas. El principal evento se denomina *sprint* y, básicamente, actúa como contenedor del resto de eventos. Los *sprints* suelen tener una duración de entre 2-4 semanas, su ciclo de vida puede observarse en la figura 2.1. Dentro de cada *sprint* se identifican los siguientes eventos:

- **Reunión de planificación del sprint:** se planifica el trabajo a realizar durante el *sprint* concretando: qué se va a entregar y cómo se va a conseguir. Con esta información y la del *product backlog* (lista completa de características que debe tener el producto a desarrollar) se genera el *sprint backlog* que, básicamente, es la lista de objetivos que se deben cumplir al finalizar el *sprint*.
- ***Scrum* diario:** breve reunión entre los miembros del equipo de desarrollo para determinar un plan para las siguientes 24 horas y sincronizar sus tareas. Para ello, se debe inspeccionar el trabajo realizado desde el último *Scrum* diario y considerando qué trabajo podría avanzarse hasta el siguiente. Se trata de una reunión clave de inspección y adaptación.
- **Trabajo de desarrollo:** es la parte del *sprint* dedicada a trabajar para avanzar en los objetivos propuestos.
- **Revisión del *sprint*:** revisión donde se inspecciona el incremento realizado y se adapta el *Product Backlog* en caso de ser necesario. Es una reunión de caracter informal entre *stakeholders* y Equipo *Scrum* para fomentar la colaboración y comunicación. En conjunto, se trata de determinar qué es lo siguiente que se debe hacer.
- **Retrospectiva del *sprint*:** el Equipo *Scrum* reflexiona sobre su propia actividad y busca mejoras que incorporar en el siguiente *sprint*.

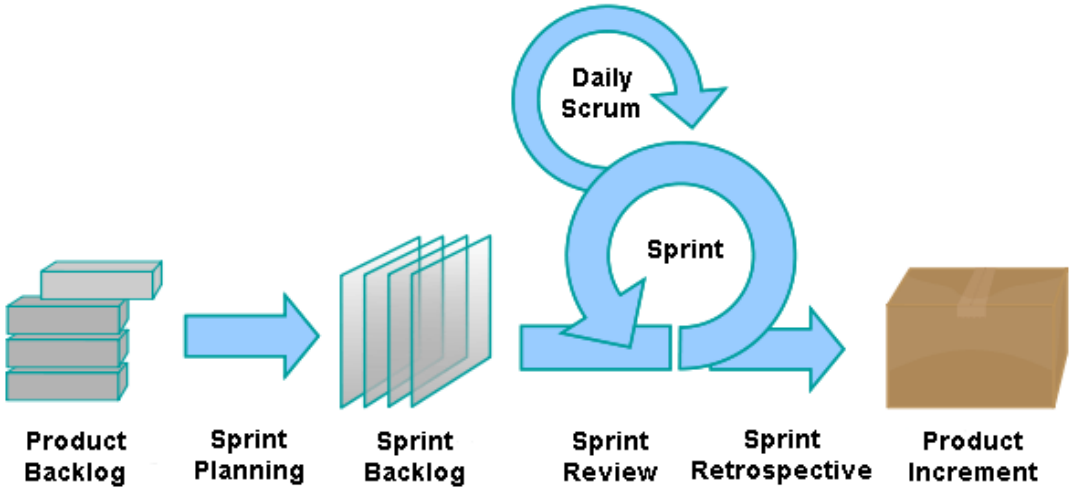

Figura 2.1: Ciclo de vida de un *sprint*

Aplicación de *Scrum*

Dado que el marco de trabajo de este proyecto no es exactamente el idóneo para el que fue ideado *Scrum*, ya que se centra en optimizar la productividad y comunicación de un equipo de trabajo, se han realizado una serie de adaptaciones para adecuar la metodología al mismo.

En primer lugar, se entenderá por reuniones diarias el tiempo empleado por la desarrolladora para organizar su actividad del día en función de los avances realizados previamente, dado que en este caso el equipo de desarrollo se reduce a una única persona.

En cuanto a los *sprints*, se fijó una duración de 3 semanas para los mismos, revisando al final de este tiempo: los avances realizados y sus posibles mejoras, el trabajo por realizar y qué hacer en el siguiente *sprint*.

2.2. Gestión de la configuración

El proceso de Gestión de la Configuración permite tener control de los elementos claves del proyecto en todo momento, los cuales experimentan gran cantidad de cambios durante su desarrollo. Estos cambios son a veces imprevistos, y pueden llegar a generar un impacto potencialmente negativo sobre el proyecto. Por esta razón, es crucial impedir que se produzcan pérdidas de información que puedan afectar al desarrollo normal del proyecto.

Se denominan elementos de configuración todos aquellos expuestos a cambios durante el desarrollo y que, por tanto, es necesario controlarlos. En el caso de este proyecto, dichos elementos son:

- Código fuente de la aplicación *FlinkBWA*
- Conjunto de información sobre el proyecto expuesta en la presente memoria.
- Ficheros varios en relación con el proyecto (actas de reunión, gráficos, diseños, etc.).

Teniendo en cuenta los elementos de configuración identificados, a continuación se plantea una gestión de la configuración para el código fuente y otra para la documentación que, englobaría el contenido de la memoria y los ficheros auxiliares al desarrollo del proyecto que pueden encontrarse volcados en la memoria o no (p.e.: las imágenes correspondientes al diseño de la aplicación se incluirán en la memoria pero no documentos como actas de reunión o similares).

2.2.1. Gestión del código

Para una gestión eficiente del código fuente de la aplicación se debe garantizar que la desarrolladora pueda acceder al código actualizado en cualquier momento y, en caso de requerirse, ser capaz de regresar a versiones anteriores del mismo.

Teniendo en cuenta los requisitos mencionados, se decidió alojar el código en un repositorio de la plataforma *Github*, empleando *git* como *software* de control de versiones. Además, se empleará *Google Drive* como *backup* adicional del código, puesto que también permite mantener un historial de versiones anteriores.

2.2.2. Gestión de la documentación

La memoria del proyecto será realizada en *LaTeX* sobre la plataforma *online Overleaf*, ya que permite la edición en línea de este tipo de documentos. Por tanto, además de servir como editor, la plataforma también proporciona un alojamiento seguro para el documento. De todas formas, al igual que con el código, cada vez que se finalice un incremento de contenido significativo, se realizará un *backup* en *Google Drive*.

2.3. Gestión del tiempo

La Gestión del Tiempo incluye procesos enfocados a administrar el proyecto para lograr su finalización a tiempo.

En esta sección se describirá la planificación inicial del proyecto mediante un EDT y la planificación final del mismo, mediante la descripción de los diferentes *sprints* que se realizaron para completar el proyecto.

2.3.1. Estructura de Descomposición del Trabajo (EDT)

El PMBOK describe a la Estructura de Descomposición del Trabajo (EDT) como una descomposición jerárquica orientada al trabajo que ejecutado por el equipo del proyecto para lograr los objetivos del mismo y crear los entregables requeridos. En otras palabras, una EDT proporciona una representación sencilla del trabajo que debe ser realizado para completar el proyecto. Realizar una EDT antes de iniciar la ejecución del proyecto proporciona grandes ventajas como: mayor determinación del alcance, mayor control sobre los objetivos a cumplir, facilidad para medir el desempeño, etc.

Las unidades de trabajo que identifica la EDT de este proyecto (figura 2.2) son:

- **Gestión del proyecto:** es la fase del proyecto dedicada a la gestión del mismo: determinar su alcance, planificar tareas, identificar y controlar riesgos, determinar el presupuesto, etc. El mayor esfuerzo en cuanto a planificación es al inicio del proyecto, pero este proceso deberá continuar durante todo el desarrollo, completándolo con detalles que sólo pueden saberse una vez finalice (p.e.: información sobre qué riesgos llegaron a materializarse).
- **Análisis y formación:** durante esta etapa la desarrolladora recopilará información acerca del contexto que rodea a la aplicación a realizar y de las tecnologías a usar. También se formará tanto en aplicaciones de alineamiento de secuencias genéticas como en tecnologías *Big Data*.
- **Diseño:** es la etapa dedicada a estudiar y realizar el mejor diseño posible para la aplicación. Se tratará de buscar un diseño modular que permita separar el código nativo del *BWA* del código correspondiente a *Flink*.
- **Implementación:** la fase de implementación engloba tanto la configuración del entorno de trabajo como la codificación de la propia plataforma *FlinkBWA*.
- **Pruebas de rendimiento:** en esta etapa se diseñarán y realizarán una serie de pruebas de rendimiento para observar los resultados relativos a eficiencia temporal de la aplicación.
- **Documentación:** fase en la que se completarán los contenidos de la memoria del proyecto y se realizará la presentación a utilizar durante la defensa del trabajo.

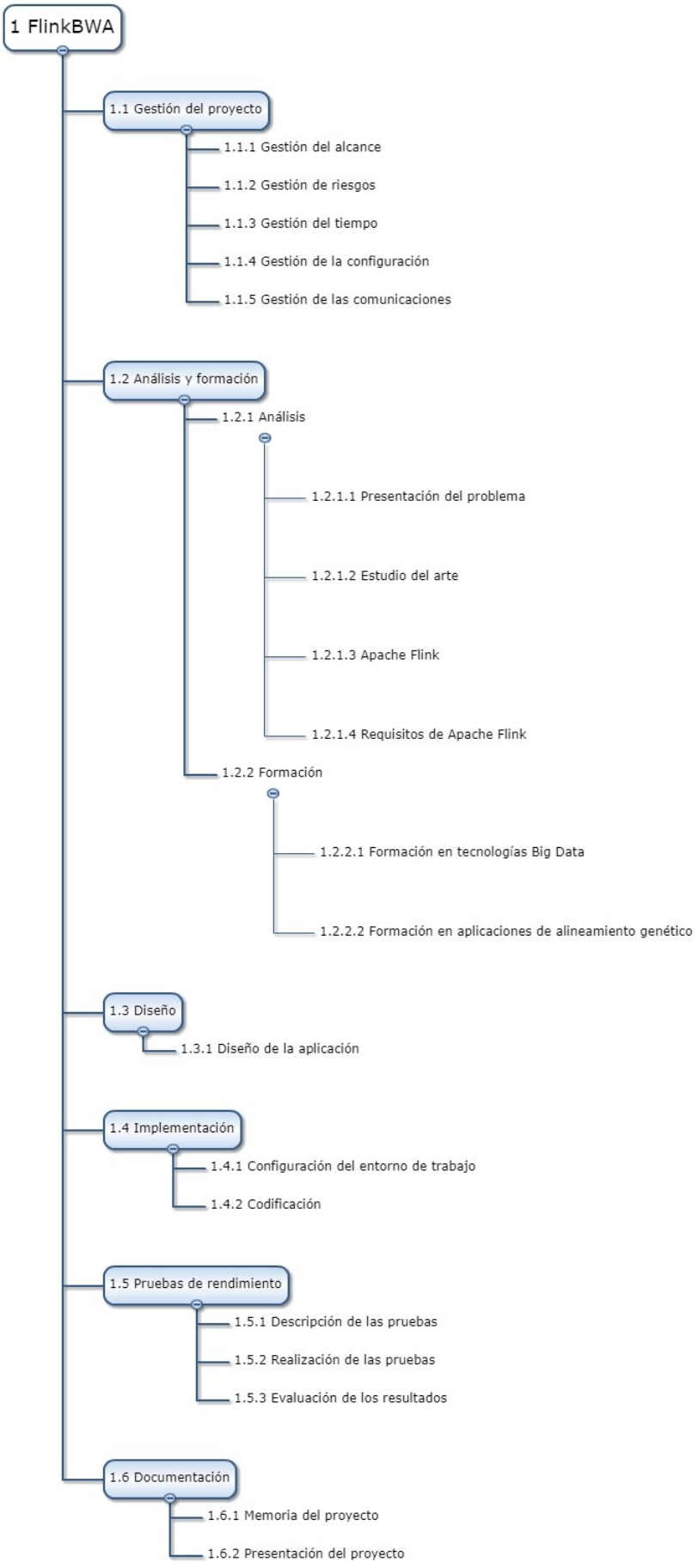

Figura 2.2: Estructura de Descomposición del Trabajo (EDT)

2.3.2. Planificación final: *sprints*

- ***Sprint* 1:** el *sprint* inicial se dedicó, en parte, a realizar toda la planificación del proyecto y también a realizar el previo análisis del mismo: presentación del problema, introducción y justificación de la tecnología a emplear: *Apache Flink*.

- ***Sprint* 2:** el segundo *sprint* incluyó la realización del diseño de la aplicación y la fase de formación necesaria para comenzar la implementación de la aplicación.

- ***Sprint* 3:**

- ***Sprint* 4:**

- ***Sprint* 5:**

2.4. Gestión de las comunicaciones

Gestionar apropiadamente la comunicación dentro de un proyecto es fundamental para tratar de lograr un entendimiento común entre todos los interesados o *stakeholders* del mismo, conectando sus niveles de experiencia, perspectivas e intereses. Esta actividad incluye todos los procesos de vinculados a la correcta generación, recopilación, distribución, almacenamiento, recuperación y disposición final de la información del proyecto.

2.4.1. Identificación de interesados

Un interesado o *stakeholder* es toda aquella persona u organización involucrada de manera activa con el proyecto y, por tanto, sus intereses pueden verse afectados -positiva o negativamente- por el desarrollo o por la finalización del mismo. Es crucial identificar los *stakeholders* en la fase inicial del proyecto, dado que en función de su importancia, sus demandas pueden tener una gran influencia sobre el proyecto.

Dado que este proyecto se corresponde con un Trabajo de Fin de Grado, se asumen como principales interesados al tutor, cotutor y a la desarrolladora, sin tener en cuenta ningún interesado adicional en el futuro. Los siguientes cuadros que se encuentran en esta subsección contienen la información completa de todos los interesados del proyecto. Concretamente, los datos que se mencionan son los siguientes:

- **Nombre:** incluye el nombre completo del interesado.

- **Empresa o grupo:** señala la empresa o grupo al que pertenece el interesado. En este caso, el grupo siempre será la Universidad de Santiago de Compostela.

- **Localización:** lugar en el que se encuentra habitualmente el interesado.

- **Rol en el proyecto:** señala el papel que tiene el interesado dentro del proyecto.

- **Información de contacto:** medio por el cual se puede contactar con el interesado. En este caso, se empleará siempre el correo electrónico como contacto.

- **Requerimientos primordiales:** principales responsabilidades del interesado en relación con el proyecto.

- **Expectativas principales:** conjunto de resultados que el interesado espera del proyecto una vez concluya.

- **Influencia potencial:** impacto que genera el interesado en el proyecto.

- **Fase de mayor interés:** determina en qué fase o fases el interesado cobra mayor importancia.

- **Interno/Externo:** indica si el interesado pertenece o no al grupo que llevará a cabo el proyecto. En este caso, todos los interesados son internos.

- **Apoyo/Neutral/Opositor:** posición del interesado en relación con el proyecto. Si el interesado sirve de apoyo, significa que ayuda al desarrollo del proyecto; un interesado neutral muestra una posición de indiferencia en cuanto a que el proyecto se complete o no; finalmente, un interesado opositor no desea que el proyecto finalice correctamente.

Nombre	Juan C. Pichel Campos
Empresa o grupo	Universidad de Santiago de Compostela (USC)
Localización	Despacho 111, CiTIUS, Campus Vida, Santiago de Compostela
Rol en el proyecto	Tutor
Información de contacto	juancarlos.pichel@usc.es
Requerimientos primordiales	Coordinar, servir de apoyo y resolver las dudas de la desarrolladora.
Expectativas principales	Finalización exitosa del proyecto, cumpliendo los requisitos señalados para el mismo
Influencia potencial	Elevada
Fase de mayor interés	Todo el proyecto
Interno/Externo	Interno
Apoyo/Neutral/Opositor	Apoyo

Cuadro 2.1: Datos del interesado Juan C. Pichel Campos

Nombre	José M. Abuín Mosquera
Empresa o grupo	Universidad de Santiago de Compostela (USC)
Localización	Laboratorio P1, CiTIUS, Campus Vida, Santiago de Compostela
Rol en el proyecto	Cotutor
Información de contacto	josemanuel.abuin@usc.es
Requerimientos primordiales	Servir de apoyo a la implementación y resolver las dudas de la desarrolladora.
Expectativas principales	Finalización exitosa del proyecto, cumpliendo los requisitos señalados para el mismo
Influencia potencial	Alta
Fase de mayor interés	Fase de implementación
Interno/Externo	Interno
Apoyo/Neutral/Opositor	Apoyo

Cuadro 2.2: Datos del interesado José M. Abuín Mosquera

Nombre	Silvia Rodríguez Alcaraz
Empresa o grupo	Universidad de Santiago de Compostela (USC)
Localización	Escuela Técnica Superior de Ingeniería (ETSE), Campus Vida, Santiago de Compostela
Rol en el proyecto	Desarrolladora
Información de contacto	silvia.rodriguez.alcaraz@rai.usc.es
Requerimientos primordiales	Diseñar, desarrollar, estudiar el rendimiento de *FlinkBWA* y realizar la documentación del proyecto.
Expectativas principales	Finalización exitosa del proyecto, cumpliendo los requisitos señalados para el mismo y pudiendo exponerlo en julio de 2019.
Influencia potencial	Alta
Fase de mayor interés	Todo el proyecto
Interno/Externo	Interno
Apoyo/Neutral/Opositor	Apoyo

Cuadro 2.3: Datos de la interesada Silvia Rodríguez Alcaraz

2.4.2. Plan de gestión de los interesados

Es preciso definir un plan de comunicación y seguirlo a lo largo del proyecto para asegurar que se intercambia información de manera apropiada entre todos los interesados. Este plan debe mostrar la información que se intercambia entre los interesados y ciertos detalles vinculados, como el medio que se usa para la comunicación o el idioma. Se han elaborado dos cuadros que pretenden mostrar los dos flujos principales de comunicación que existen y las principales características de las mismas:

Emisora	Silvia Rodríguez Alcaraz
Receptores	Tutor y/o cotutor
Propósito	Exponer los avances del proyecto, preguntar dudas y/o solicitar reuniones extra.
Nivel de detalle	Alto
Importancia	Muy importante
Periodicidad	3-4 semanas
Formato	Alta
Idioma	Castellano o gallego

Cuadro 2.4: Matriz de comunicación: Desarrolladora → Tutor y/o cotutor

Emisor/es	Tutor y/o cotutor
Receptora	Desarrolladora
Propósito	Guiar el desarrollo, resolver dudas, indicar pautas a seguir, matizar requisitos y preguntar sobre el avance del proyecto.
Nivel de detalle	Alto
Importancia	Muy importante
Periodicidad	3-4 semanas
Formato	Alta
Idioma	Castellano o gallego

Cuadro 2.5: Matriz de comunicación: Tutor y/o cotutor → Desarrolladora

2.5. Gestión de riesgos

Gestionar los riesgos de un proyecto es uno de los aspectos más relevantes de la planificación, ya que multitud de proyectos bien planteados en un inicio fracasan por la aparición de determinadas situaciones no contempladas a priori. De hecho, la gestión de riesgos consiste en planificar, organizar, dirigir y controlar los recursos disponibles con el fin de evitar, minimizar o prever posibles riesgos.

2.5.1. Identificación de riesgos

Los principales riesgos identificados en este proyecto son los que se citan a continuación:

- **RSK-001:** dificultades en el proceso de formación.
- **RSK-002:** falta de comprensión del código nativo del BWA.
- **RSK-003:** desajustes en la planificación.
- **RSK-004:** pérdida de información vinculada al proyecto.
- **RSK-005:** errores en la elección de las tecnologías de desarrollo.
- **RSK-006:** insuficiente disponibilidad de la desarrolladora.
- **RSK-007:** insuficiente disponibilidad de los expertos.
- **RSK-008:** avería en el ordenador de desarrollo.
- **RSK-009:** no disponibilidad de las plataformas *online* empleadas.
- **RSK-010:** no disponibilidad del *clúster*.

2.5.2. Análisis de riesgos

A continuación, se analizarán los riesgos identificados previamente tomando como referencia el formato de la tabla 2.6:

Identificador del riesgo	
Nombre	Nombre del identificador.
Descripción	Pequeña descripción del riesgo.
Probabilidad	Alta, media o baja.
Impacto	Tolerable, serio o catastrófico.
Indicador	Hechos que ponen de manifiesto la materialización de un riesgo.

Cuadro 2.6: Plantilla para análisis de riesgos

Los aspectos más relevantes a considerar en el análisis son la probabilidad, el impacto y la exposición al riesgo:

- **Probabilidad:** determina las posibilidades que tiene dicho riesgo de ocurrir.
- **Impacto:** señala cuán significativo para el desarrollo normal del proyecto es que dicho riesgo se materialice.
- **Exposición:** muestra la frecuencia de aparición que puede tener el riesgo.

RSK-001	
Nombre	Dificultades en el proceso de formación.
Descripción	Debido a la inexperiencia de la desarrolladora con las tecnologías escogidas para la implementación y con la naturaleza de la propia aplicación, es posible que aparezcan dificultades o dudas a nivel conceptual durante la fase de formación.
Probabilidad	Media
Impacto	Tolerable
Indicador	Aparición de dudas de la desarrolladora y dificultad para avanzar en el proyecto.

Cuadro 2.7: Análisis del RSK-001

RSK-002	
Nombre	Falta de comprensión del código nativo BWA.
Descripción	Debido a la inexperiencia de la desarrolladora en cuanto a aplicaciones de alineamiento de secuencias genéticas es posible que se encuentren dificultades de comprensión a la hora de interpretar y trabajar con el código nativo del BWA.
Probabilidad	Media
Impacto	Tolerable
Indicador	Aparición de dudas de la desarrolladora y dificultad para avanzar en el proyecto.

Cuadro 2.8: Análisis del RSK-002

RSK-003	
Nombre	Desajustes en la planificación.
Descripción	A causa de multitud de factores pueden darse retrasos con respecto a la planificación realizada en un primer momento. Es decir, algunas fases pueden atrasarse o, al contrario, pueden finalizarse tareas antes de lo previsto, desajustándose la planificación inicial.
Probabilidad	Alta
Impacto	Serio
Indicador	Finalización de un *sprint* sin todos los contenidos previstos finalizados. O bien, finalización de todos los contenidos previstos antes del fin del *sprint*.

Cuadro 2.9: Análisis del RSK-003

RSK-004	
Nombre	Pérdida de información vinculada al proyecto.
Descripción	Puede ser que por problemas con los medios físicos o con las plataformas *online* empleadas, se pierda información vinculada al proyecto en un momento dado.
Probabilidad	Media
Impacto	Catastrófico
Indicador	Pérdida de información valiosa y y crucial para la entrega a tiempo del proyecto.

Cuadro 2.10: Análisis del RSK-004

RSK-005	
Nombre	Errores en la elección de las tecnologías de desarrollo.
Descripción	Puede ser que exista una incompatibilidad no observada a priori entre la plataforma *Big Data* a utilizar y la aplicación del alineador de secuencias genéticas u con otros aspectos del entorno de desarrollo.
Probabilidad	Baja
Impacto	Catastrófico
Indicador	Imposibilidad de continuar con la implementación.

Cuadro 2.11: Análisis del RSK-005

RSK-006	
Nombre	Insuficiente disponibilidad de la desarrolladora.
Descripción	Por problemas de carácter personal o la aparición de compromisos inesperados podría suceder que la desarrolladora no contase con todo el tiempo previsto inicialmente para dedicar al proyecto.
Probabilidad	Media
Impacto	Serio
Indicador	Ritmo de trabajo más lento por parte de la desarrolladora.

Cuadro 2.12: Análisis del RSK-006

RSK-007	
Nombre	Insuficiente disponibilidad de los expertos.
Descripción	Por demasiada carga a nivel laboral, problemas de caracter personal o la asistencia a congresos en el extranjero puede ser que la disponibilidad de los expertos se reduzca.
Probabilidad	Media
Impacto	Serio
Exposición	Media
Indicador	Problemas a la hora de comunicarse o fijar reuniones con los expertos .

Cuadro 2.13: Análisis del RSK-007

RSK-008	
Nombre	Avería en el ordenador de desarrollo.
Descripción	Cabe la posibilidad de que el ordenador de desarrollo se averíe por un accidente o simplemente por el uso, ya que no es un equipo demasiado nuevo.
Probabilidad	Media
Impacto	Catastrófico
Indicador	Imposibilidad de avanzar con el proyecto hasta encontrar o conseguir un nuevo equipo, además de la posible pérdida de información local que podría suponer.

Cuadro 2.14: Análisis del RSK-008

RSK-009	
Nombre	No disponibilidad de las plataformas *online* empleadas.
Descripción	Una caída duradera de las plataformas *overleaf* y//o *Github* podrían frenar considerablemente el avance normal del proyecto si no se han hecho copias en local.
Probabilidad	Baja
Impacto	Catastrófico
Indicador	Detención del avance del proyecto hasta que dichas plataformas vuelvan a tener disponibilidad.

Cuadro 2.15: Análisis del RSK-009

RSK-010	
Nombre	No disponibilidad del *cluster*.
Descripción	El *cluster* empleado tanto para las pruebas como para dar soporte a la plataforma *Hadoop* utilizada en la implementación puede sufrir una caída o problemas técnicos.
Probabilidad	Media
Impacto	Serio
Indicador	No se puede acceder a los datos genómicos ni al resto de recursos del *cluster*.

Cuadro 2.16: Análisis del RSK-010

2.5.3. Planificación de respuesta a los riesgos

Tomando como referencia la probabilidad y el impacto de los riesgos analizados anteriormente, se ha generado una matriz de exposición con el fin de facilitar el control de los mismos. Dicha matriz muestra la probabilidad que existe de que los riesgos se lleguen a materializar y, por tanto, la elección de técnicas de control sobre los mismos.

		Impacto		
		Catastrófico	Serio	Tolerable
Probabilidad	Alta		RSK-003	RSK-001
		RSK-004	RSK-006	RSK-002
		RSK-008	RSK-007	
	Media		RSK-010	
		RSK-005		
	Baja	RSK-009		

Cuadro 2.17: Matriz de exposición ante riesgos

En función de los resultados mostrados en la tabla 2.17, se ha decidido controlar todos los riesgos con un nivel de exposición medio (área con fondo amarillo en la matriz), ya que no se detectó ningún riesgo con un nivel de exposición alto. Por tanto, se entiende que ante los riesgos con baja exposición (área de fondo verde) la estrategia de control es la aceptación: no se realizará ninguna acción con respecto a dichos riesgos hasta su activación, en caso de darse.

En cuanto a los riesgos que van a ser controlados (RSK-003, RSK-004, RSK-006, RSK-007, RSK-008 Y RSK-010), las estrategias a seguir son las siguientes:

- **Prevención:** conjunto de acciones a realizar antes de que se materialice un riesgo con el fin de que este no llegue a suceder.
- **Contingencia:** conjunto de acciones a realizar una vez se materializa un riesgo con el fin de reducir o minimizar sus efectos.

Identificador del riesgo	
Nombre del riesgo	Nombre completo del riesgo.
Acciones de prevención	Acciones a aplicar antes de que ocurra el riesgo.
Acciones de contingencia	Acciones a aplicar una vez ocurra el riesgo.

Cuadro 2.18: Plantilla para control de riesgos

A continuación, se presenta el plan de control de riesgos propuesto:

RSK-003	
Nombre del riesgo	Desajustes en la planificación.
Acciones de prevención	Para tratar de evitar posibles desajustes se realizará una planificación basada en sprints que cuente con un margen relativo para posibles imprevistos.
Acciones de contingencia	Si el riesgo se materializa, el margen dedicado a imprevistos de los sprints posteriores deberá reducirse con el fin de ganar algo de tiempo. En el peor de los casos, será necesario recortar el alcance o alguno de los requisitos del proyecto.

Cuadro 2.19: Plan de control para RSK-003

RSK-004	
Nombre del riesgo	Pérdida de información vinculada al proyecto.
Acciones de prevención	Se almacerá todo el proyecto tanto en la plataforma *Github* como en *Google Drive*, con el fin de tener respaldados de forma online los contenidos del proyecto, además de en un disco duro externo.
Acciones de contingencia	En caso de que el riesgo llegue a materializarse se accederá a las copias *online* del mismo o a la copia del disco externo. Teniendo en cuenta que son 3 copias (un par en 2 plataformas *online* diferentes y la restante en un dispositivo físico) es altamente improbable que ninguna de esas copias sea accesible.

Cuadro 2.20: Plan de control para RSK-004

RSK-006	
Nombre del riesgo	Insuficiente disponibilidad de la desarrolladora.
Acciones de prevención	Tratar de avanzar lo máximo posible de forma diaria para que dicho trabajo compense la pérdida de tiempo que puede generar un imprevisto.
Acciones de contingencia	Dependiendo del punto del proyecto en el que se materialice el riesgo y la duración del mismo en el tiempo se podría: bien, tratar de cubrir el tiempo perdido con el ahorrado en tareas que finalizaron antes; bien, replantear el alcance del proyecto con para tratar de finalizarlo exitosamente en el tiempo establecido.

Cuadro 2.21: Plan de control para RSK-006

RSK-007	
Nombre del riesgo	Insuficiente disponibilidad de los expertos.
Acciones de prevención	Fijar reuniones periódicas, determinadas con suficiente antelación. Acompañar esta medida con contacto frecuente vía correo electrónico.
Acciones de contingencia	Solicitar la atención de otro experto en caso de que la disponibilidad sea alarmantemente reducida.

Cuadro 2.22: Plan de control para RSK-007

RSK-008	
Nombre del riesgo	Avería en el ordenador de desarrollo.
Acciones de prevención	Tratar con el mayor cuidado posible el equipo.
Acciones de contingencia	Tratar de repararlo o buscar un equipo con el que finalizar el proyecto.

Cuadro 2.23: Plan de control para RSK-008

RSK-010	
Nombre del riesgo	No disponibilidad del *cluster*.
Acciones de prevención	Estudiar la posibilidad de trabajar en local sin el *cluster*.
Acciones de contingencia	Configurar el entorno *hadoop* en local y guardar los datos genómicos también en local.

Cuadro 2.24: Plan de control para RSK-010

2.5.4. Materialización de riesgos

2.6. Gestión de costes

La Gestión de Costes incluye el conjunto de actividades que tienen por objetivo estimar, presupuestar y controlar los costos para que el proyecto pueda finalizarse exitosamente sin exceder el presupuesto fijado.

2.6.1. Consideraciones previas

A continuación se aclaran una serie de aspectos sobre la información que figura en esta sección:

- **Unidad de medida:** la unidad de medida empleada para los costes será el euro (), dado que el proyecto se desarrolla en España, país perteneciente a la Unión Económica y Monetaria europea (UEM).
- **Nivel de exactitud:** esta sección muestra en su mayoría datos estimados, dado que no se posee información exacta del coste de los recursos humanos y materiales. Por tanto, aunque se han tratado de hacer aproximaciones lo más cercanas a la realidad posible, pueden existir ciertas diferencias con el coste total del proyecto, calculado con todos los datos exactos de los recursos utilizados.
- **Nivel de precisión:** las cifras mostradas sólo llegarán a los céntimos de euro para evitar el uso de fracciones, ya que más bien complican los cálculos en lugar de aportar información relevante. En caso de aparecer una cifra de este tipo se realizará un redondeo de la misma.

2.6.2. Costos directos

Los costos directos de un proyecto son todos aquellos relacionados estrechamente con el producto a realizar. Es decir, todos aquellos materiales tangibles o intangibles empleados para la realización del proyecto. En este caso, se desglosará

el cálculo de los costos directos en: costos de los recursos *materiales* y costos de los Recursos Humanos.

Costos en recursos *materiales*

Los recursos materiales empleados en este proyecto son los siguientes:

- **Equipo de desarrollo:** compuesto por un ordenador portátil, cargador del mismo, ratón USB y en ocasiones una pantalla auxiliar, teclado y adaptador HDMI. El valor del equipo portátil se estima en unos 600 que, al incluir todos los accesorios adicionales, serían finalmente unos 700. La vida media del equipo sería aproximadamente de 4 años (48 meses), utilizándolo para el proyecto un total de 4 meses. Por tanto, el coste en el proyecto de este equipo sería de 58.33 €:

$$\frac{700 euros}{48 meses} = \frac{X euros}{4 meses} \rightarrow X = 58.33 €$$

- ***Software:*** todo el *software* a utilizar durante el proyecto es libre, o bien, se cuenta con licencia de estudiante para utilizarlo y, por tanto, no supondrá un gasto (como es el caso del IDE de desarrollo: *IntelliJ* de *Jetbrains*).

- **Material fungible:** el costo de folios, carpetas, bolígrafos, fotocopias y demás utilizado para el proyecto se estima en aproximadamente 130 € (incluyendo las 3 copias impresas de la memoria).

Costos en *Recursos Humanos*

Los costos relativos a los Recursos Humanos son todos aquellos asociados al equipo del proyecto, en este caso: alumna, tutor y cotutor.

Como no se conocen con exactitud los sueldos de los componentes del equipo, por una parte, para determinar los sueldos del tutor y cotutor se consultaron las Tablas de Retribuciones del Personal Docente e Investigador (PDI) de la USC [9]. Así, suponiendo para el tutor el sueldo mensual de un profesor contratado doctor (2506,81 €) y para el cotutor el de profesor ayudante doctor ((EUR1957,56), solo faltaría calcular el costo por horas:

$$\text{Tutor} \rightarrow \frac{2506{,}81\ euros}{22\ días/mes * 8\ h} = 14{,}24 €/\text{h} \qquad \text{Cotutor} \rightarrow \frac{1957{,}56\ euros}{22\ días/mes * 8\ h} = 11{,}12 €/\text{h}$$

Por otra parte, para determinar el sueldo de la alumna se consultaron las Bases y tipos de cotización de la Seguridad Social [10], observando que el sueldo mínimo para ayudantes no titulados es de 1050 €/mes. Calculando el costo por horas para este caso:

$$\text{Alumna} \rightarrow \frac{1050\,euros}{22\,días/mes * 8\,h} = 5{,}97\,€/\text{h}$$

A continuación, se presenta la tabla 2.25 que contiene el cálculo total de los costos que suponen los Recursos Humanos del proyecto:

Costos Recursos Humanos					
Recurso	**Rol**	**Sueldo (€ / mes)**	**Costo (€/hora)**	**Horas**	**Costo total (€)**
Tutor	Profesor contratado doctor	2506,81	14,24	11,25	160,2
Cotutor	Profesor ayudante doctor	1957,56	11,12	11,25	126
Alumna	Desarrolladora analista	1050	5,97	412,99	2465,55
Costo Total RRHH (€)					**2751,75**

Cuadro 2.25: Costos de los Recursos Humanos

2.6.3. Costos indirectos

Los costos indirectos son aquellos que no tienen un impacto directo sobre el proyecto, pero que son indispensables para la realización del mismo. Por tanto, se considerarían costos indirectos el gasto en luz, labores administrativos, etc.

Para determinar los costos indirectos de este proyecto se ha tomado como referencia el valor que establece de forma oficial la USC para los proyectos TIC [11], que es un 21 % adicional sobre el valor de los costos directos.

Teniendo en cuenta que el valor de los costos directos es de 2751,75 €, los costos indirectos de este proyecto supondrían un total de 617,42 €.

2.6.4. Financiamiento

Dada la naturaleza del presente proyecto, un Trabajo de Fin de Grado (TFG), no habrá ningún tipo de pago por parte de ningún cliente.

Por tanto, en este contexto concreto, se ha omitido la realización de cálculos en cuanto al desembolso inicial, la realización de gráficos de flujo de caja (gráficos que muestran cobros y pagos en un determinado período) y demás técnicas empleadas para controlar la financiación de un proyecto.

2.6.5. Costo total

Los costes totales del proyecto sería la suma de los costes directos y los indirectos que, como ya se ha mencionado, suponen el 21 % de los mismos. La tabla 2.26, que se presenta a continuación, incluye el recuento de todos los costos del proyecto:

Costos del proyecto		
Nombre	**Tipo de costo**	**Costo (€)**
Equipo de desarrollo	Directo	58,33
Material fungible	Directo	130
RRHH	Directo	2751,75
Total costos directos (€)		2940,08
Costos indirectos (€)		617,42
Costo Total (€)		**3557,50**

Cuadro 2.26: Costos del proyecto

Capítulo 3

Especificación de requisitos

Según el PMBOK, se conoce como requisitos al conjunto de condiciones o capacidades que debe tener un sistema, producto, servicio o componente para satisfacer un contrato, estándar, especificación, u otros documentos formalmente establecidos. Es, por tanto, fundamental y necesario cumplir con los requisitos del proyecto para alcanzar el éxito del mismo.

Este capítulo está dedicado a la identificación y descripción de los requisitos que debe cumplir el proyecto para considerarse finalizado con éxito.

3.1. Definición del sistema

Esta memoria se corresponde con un proyecto que no es exactamente un desarrollo de *software* enfocado a un usuario final, sino que trata de incluir una tecnología concreta en una aplicación ya existente y funcional con el fin de mejorar su rendimiento. En base a esto, cabe hacer una serie de aclaraciones:

- **Requisitos funcionales:** los requisitos funcionales son los que identifican funcionalidades que definen al *software* a realizar. Puesto que no se va a realizar ningún tipo de ampliación sobre la funcionalidad de la aplicación *Burrows-Wheeler Aligner* (BWA), no hay ningún tipo de requisito funcional en este trabajo.

- **Matriz de trazabilidad:** la matriz de trazabilidad sirve para relacionar los casos de uso con los requisitos funcionales, entendiendo los casos de uso como "partes" útiles del *software*, proporcionaría una traza entre los entregables del proyecto y los requisitos del mismo. Puesto que en este trabajo no se cuenta con dichos requisitos funcionales, no tiene sentido la realización de una matriz de trazabilidad.

- **Actores:** en el Lenguaje Unificado de Modelado (UML) [12], un actor se emplea para modelar un rol jugado por un usuario, entidad, *hardware* u otro que interactúa con el sistema. En este caso concreto, en principio, sólo la desarrolladora y los expertos interactuarán con el sistema, pues lo que se busca dentro de este proyecto es estudiar su rendimiento.

- **Casos de uso:** describen acciones o actividades. Un diagrama de casos de uso representa un sistema o subsistema como un conjunto de interacciones entre casos de uso y entre dichos casos y sus pertinentes actores en respuesta a un evento iniciado por un actor principal. Los casos de uso de esta aplicación serían los mismos que el alineador no paralelo BWA del que se parte, pero no tiene sentido comentarlos en este contexto, pues el proyecto sólo está enfocado al estudio del rendimiento de la aplicación final.

3.2. Catálogo de requisitos

En esta sección se identificarán los principales requisitos no funcionales detectados en las reuniones iniciales del proyecto y durante la fase de análisis. Además, se describirán dichos requisitos en base a su importancia y a sus criterios de aceptación. Cabe destacar que se distinguen los siguientes tres niveles de importancia:

Nivel de importancia	Descripción
Vital	Su cumplimiento es fundamental para la finalización exitosa del proyecto.
Importante	Su cumplimiento afecta únicamente de cara a incrementar el valor del proyecto, pero su no cumplimiento no supondría una pérdida sustancial.
Deseable	Su cumplimiento aumentaría la calidad del proyecto, pero no es obligatorio cumplirlo.

Cuadro 3.1: Niveles de importancia

La plantilla que se seguirá para la especificación de los requisitos es la que se muestra en la tabla 3.2:

ID	Nombre
Descripción	Breve explicación sobre qué implica y en qué consiste el requisito.
Importancia	Determina la importancia del requisito.
Criterios de aceptación	Menciona cuáles serán los criterios que validan el cumplimiento del requisito.

Cuadro 3.2: Plantilla de especificación de requisitos

3.2.1. Identificación de requisitos no funcionales

- **RNF-001:** configuración del *cluster* para trabajar con *Apache Flink*.
- **RNF-002:** paralelización del BWA con *Apache Flink*.
- **RNF-003:** realización de pruebas de rendimiento sobre *FlinkBWA*.
- **RNF-004:** determinación de la influencia de *Flink* en el rendimiento de la aplicación.
- **RNF-005:** utilización de *software* libre o gratuíto para uso estudiantil.
- **RNF-006:** utilización de Java como lenguaje de desarrollo.

3.2.2. Especificación de requisitos no funcionales

RNF-001	Configuración del cluster para trabajar con Apache Flink
Descripción	Debe instalarse y configurarse Apache Flink en el cluster a utilizar para poder realizar tanto la implementación de la aplicación como las pruebas de rendimiento sobre la aplicación final.
Importancia	Vital.
Criterios de aceptación	Se considera validado cuando el cluster este totalmente configurado y adaptado para poder comenzar la fase de implementación de la aplicación.

Cuadro 3.3: Especificación RNF-001

RNF-002	Paralelización del BWA con Apache Flink
Descripción	Implementación de la aplicación paralela FlinkBWA.
Importancia	Vital.
Criterios de aceptación	Se considera validado cuando se tenga una versión final y funcional del BWA paralelizado con Flink.

Cuadro 3.4: Especificación RNF-002

RNF-003	Realización de pruebas de rendimiento sobre FlinkBWA
Descripción	Planteamiento y realización de pruebas de rendimiento sobre la aplicación FlinkBWA.
Importancia	Vital.
Criterios de aceptación	Se considera validado cuando posean datos empíricos sobre la eficiencia temporal de FlinkBWA.

Cuadro 3.5: Especificación RNF-003

RNF-004	Determinación de la influencia de Flink en el rendimiento de la aplicación
Descripción	Estudio comparativo entre la aplicación inicial del BWA y su versión paralelizada con Apache Flink.
Importancia	Importante.
Criterios de aceptación	Se considera validado en el momento en que se posean datos comparatativos entre el rendimiento temporal de una y otra aplicación.

Cuadro 3.6: Especificación RNF-004

RNF-005	Utilización de software libre o gratuito para uso estudiantil
Descripción	Se deberá emplear durante todo el proyecto software considerado open-source o libre para fines estudiantiles.
Importancia	Deseable.
Criterios de aceptación	Se considera validado hasta que surja la necesidad de pagar una cantidad determinada por el uso de un servicio o producto necesario para el desarrollo del proyecto.

Cuadro 3.7: Especificación RNF-005

RNF-006	**Utilización de Java como lenguaje de desarrollo**
Descripción	Se deberá emplear el lenguaje Java para el desarrollo del módulo correspondiente a Flink.
Importancia	Deseable.
Criterios de aceptación	Se considera validado en el momento que se finalice el desarrollo de FlinkBWA y dicha aplicación tenga las mismas funcionalidades que el BWA inicial.

Cuadro 3.8: Especificación RNF-006

Capítulo 4

Análisis

Este capítulo está dedicado a introducir al lector al contexto que engloba el presente trabajo, que sería: por una parte, las aplicaciones de alineamiento genético; por otra, el mundo de las tecnologías *Big Data*. Se hará especial hincapié en presentar tanto el alineador de secuencias genéticas como la plataforma *Big Data* que se emplean en este proyecto: el BWA y *Apache Flink*.

4.1. Aplicaciones de alineamiento genético

La biotecnología es un área multidisciplinar en auge con múltiples aplicaciones en la actualidad. Tomando como referencia el Artículo 2 de la "Convención sobre la diversidad biológica" de las Naciones Unidas de 1992 [13], se puede definir la biotecnología como toda aquella aplicación tecnológica que utilice sistemas biológicos y organismos vivos o sus derivados para la creación o modificación de productos o procesos para usos específicos.

Una sus múltiples aplicaciones es el alineamiento de secuencias genéticas. En los ámbitos de la bioquímica, genética y biología molecular, se conoce como homología [14] la similitud entre dos o más secuencias de proteínas o ácidos nucléicos por mostrar un mismo origen evolutivo. Determinar estas homologías proporciona información de gran utilidad: una similitud puede señalar relaciones funcionales o evolutivas entre los genes o proteínas consultados; además, si tienen un ancestro común, las no coincidencias pueden interpretarse como mutaciones puntuales. Cabe suponer la notoria cantidad de información que reside en dichas secuencias, por lo que, para encontrar rápidamente discrepancias o similitudes entre ellas se han llevado acabo aplicaciones bioinformáticas que facilitan potencialmente esta comparación.

Los alineadores de secuencias genéticas se encargan de mapear los datos almacenados contra una secuencia de referencia conocida y previamente indexada,

con el fin de acelerar las búsquedas de los algoritmos de alineamiento. Existen distintos tipos de alineadores según el algoritmo que emplean internamente, el uso de memoria que requieren, su velocidad, etc. Según el algoritmo que utilizan, se distinguen dos grupos [15]: los basados en *hashing* y los basados en la transformación *Burrows-Wheeler* [16]. Teniendo en cuenta que en este proyecto se trabaja con un alineador basado en este último algoritmo, la explicación se centrará en este segundo grupo.

Los alineadores *Burrows-Wheeler* se basan en este algoritmo, el cual consiste en la transformación de una cadena de caracteres en otra mucho más sencilla de comprimir. Dicho reordenamiento permuta los caracteres, haciendo que la cadena resultante agrupe los caracteres similares en la cadena original, siendo más sencillo comprimirla. Por tanto, se puede decir que es un algoritmo de compresión de datos sin pérdida, ya que los caracteres no cambian su valor tras la transformación. Otro punto a favor de la transformación es que su resultado es reversible, por lo que a partir del resultado final es posible volver a la cadena inicial realizando una serie de operaciones. Recientemente, se han desarrollado varios alineadores que emplean este algoritmo para tratar de reducir el uso de memoria que se emplea en los alineamientos, con el que se trabaja en el proyecto actual es el *Burrows-Wheeler Aligner* (BWA).

4.1.1. *Burrows-Wheeler Aligner*

El *Burrows-Wheeler Aligner* (BWA) [17] es un paquete de *software* libre que usa la transformación *Burrows-Wheeler* (BWT, *Burrows-Wheeler Transformation*) para indexar un genoma de referencia grande, como el humano, y mapear secuencias de baja divergencia (es decir, muy similares entre sí) contra dicho genoma.

Este alineador utiliza tres algoritmos para conseguir optimizar el alineamiento y la rapidez del mismo: BWA-*backtrack* [18], BWA-SW [19] y BWA-MEM [20]. El primero, está diseñado para leer secuencias de un tamaño pequeño (hasta 100 pares de bases), mientras que los otros dos se centran en secuencias más largas. BWA-MEM es el más reciente y ha mostrado mejor rendimiento en comparación con otros alineadores de lectura de última generación a la hora de mapear lecturas de 100 pares de bases o más.

Dado que la alineación de secuencias es un proceso costoso a nivel temporal, como ya se ha mencionado, BWA cuenta con una implementación paralela que permite reducir este tiempo. El problema es que dicha implementación sólo es compatible con máquinas de memoria compartida, haciendo que la escalabilidad

siempre esté limitada por el número de núcleos y la memoria disponible en el nodo de computación.

El BWA puede aceptar como entrada tanto archivos BAM [21] no alineados como archivos en formato FASTQ. El formato BAM es una versión comprimida del formato SAM, que permite realizar un indexado para tener acceso directo a las posiciones genómicas. Básicamente, un fichero SAM (*Sequence Alignment/Map format*) se compone de texto tabulado cuyo fin es la representación de alineamientos de secuencias contra un genoma o secuencia de referencia, compuesto por una sección de cabecera (opcional) y una sección de alineamiento. Por su parte, el formato FASTQ es su entrada habitual y puede considerarse el *input* estándar de las herramientas bioinformáticas encargadas del alineamiento. A continuación, se incluye un ejemplo que muestra la estructura de este tipo de ficheros:

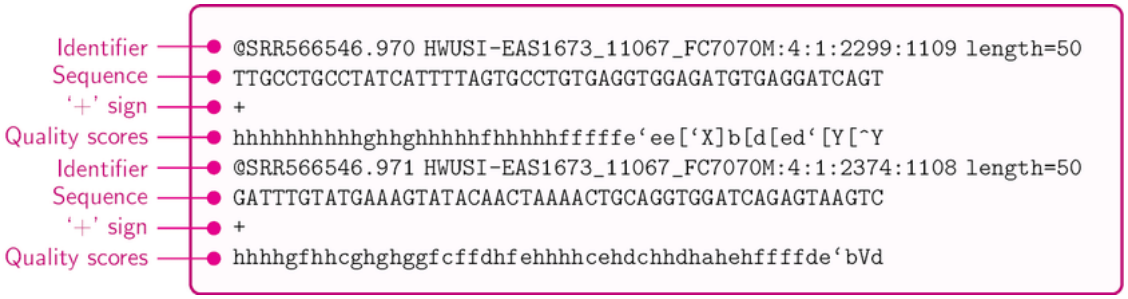

Figura 4.1: Ejemplo del formato FASTQ

La figura 4.1 contiene dos lecturas, dado que en cuatro líneas se describe una secuencia o lectura, incluyendo como información: un identificador único, la secuencia e información de calidad de la lectura, separada del resto de datos por el símbolo "+".

Cabe destacar que este alineador es capaz de aceptar como entrada un único fichero FASTQ (lectura de un sólo extremo) o dos archivos FASTQ (lecturas de extremo emparejado). En el segundo caso, están disponibles dos secuencias que se corresponden a los dos extremos de un mismo fragmento de ADN. Dichas lecturas se corresponden con distintos ficheros de entrada, como ya se mencionó, pero comparten identificador.

El formato SAM mencionado anteriormente es, de hecho, el formato que emplea el BWA como *output*, ya que es más legible que el BAM de cara a ser interpretado por un humano.

4.2. Estado del arte: tecnologías *Big Data*

Pese a que hoy por hoy el término *Big Data* tiene un uso extendido y sus tecnologías están en pleno auge, no existe un consenso claro a la hora de dar una definición exacta. Muchos profesionales determinan que el *Big Data* recoge todos los procesos de extracción, tranformación y carga (operaciones ETL: *extraction, transform and load*) que se realizan sobre grandes volúmenes de datos. Otra descripción comúnmente utilizada se basa en tres atributos de los datos, conocidos como "3Vs": volumen, velocidad y variedad. Pero ninguna de estas definiciones engloba realmente todos los aspectos que implica el término *Big Data*. Por lo pronto, en lo que respecta a este trabajo, basta con entender el término *Big Data* como el conjunto de tecnologías específicas que facilitan la tarea de almacenamiento, procesado y análisis de datos cuyo volumen y/o complejidad supera las capacidades de cómputo de una máquina convencional.

Figura 4.2: Cronograma sobre la evolución de *Big Data*

En la figura 4.2 extraída del libro "*Big Data: Principles and Paradigms*" [22] se muestra un cronograma que recoge los hitos más importantes de la historia del

Big Data hasta el año 2015.

Se podría concluir que estas tecnologías surgen a partir de la necesidad de las principales empresas tecnológicas de finales de los 90 y principios del 2000 de explotar su banco de datos para potenciar su negocio (lo que hoy se conoce como *Data Mining*). Un ejemplo sería el caso de *Google*, quien desarrolló su propia solución para automatizar eficientemente la distribución de grandes volúmenes de datos en un conjunto de máquinas. Dicho *software* fue bautizado como *MapReduce* y permitió reducir las tareas del programador a definir los detalles de dos funciones: "map" y "reduce", las cuales aluden a las dos etapas principales del procesamiento de datos. La primera se encargaría de transformar un conjunto de datos inicial en otro conjunto intermedio estructurado de la forma "clave-valor" y, la segunda, procesa los valores de los datos intermedios correspondientes a una misma clave proporcionando un resultado final. En la figura 4.3 [23] se muestra un ejemplo de este modelo de programación para contar el número de apariciones de ciertas palabras en un documento:

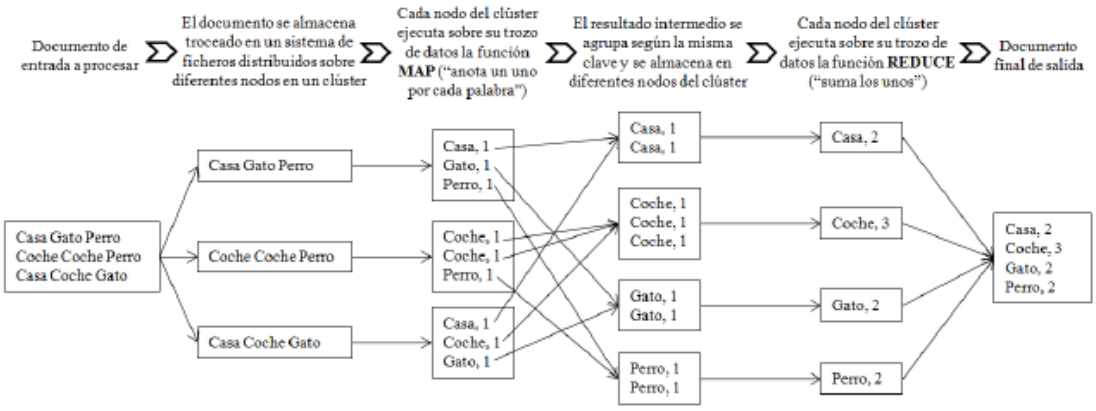

Figura 4.3: Ejemplo del modelo *Map Reduce*

La publicación de *MapReduce* por parte de *Google* inspiró a otros proyectos similares. Es el caso de *Apache Hadoop*[1], un sistema que implementa *MapReduce*, siendo capaz de procesar de forma distribuida enormes volumenes de datos. Esta plataforma de código abierto facilitó la creación de nuevas tecnologías que aprovechaban su funcionalidad, de hecho, hoy en día existe un gran ecosistema de herramientas que lo utilizan como base. Algunas de estas herramientas serían: *Apache Pig, Apache Hive, Apache Storm*, etc.

Con el tiempo, el modelo *MapReduce* empezó a presentar limitaciones en cuanto a su eficiencia en determinadas situaciones. Consecuencia de este hecho, empiezan

[1]https://hadoop.apache.org/

a abrirse nuevas vías de desarrollo. Una, orientada a mejorar dicho modelo; y otras, orientadas a crear modelos alternativos de procesamiento que mantengan los principios de *MapReduce* pero mejoren sus capacidades. Como ejemplo de la primera vía se podría destacar la evolución de *Hadoop*, quien mejora su estructura y gestión interna, sustituyendo su versión anterior por una nueva denominada YARN[2]. En la segunda vía, se destacaría la aparición de *Apache Spark*[3] en 2010, basado en un uso prioritario de la memoria principal frente al almacenamiento en disco y la utilización de la abstracción de datos *Resilient Distributed Datasets* (RDD[4]) para ofrecer mejores resultados que *Hadoop* ante procesos iterativos o casos de consultas extremos.

4.2.1. *Apache Flink*

Como se mencionaba anteriormente, la aparición de tecnologías de almacenamiento de grandes volúmenes de datos, como *Hadoop* cambió totalmente la dinámica de dichas empresas que, en lugar de descartarlos, pudieron empezar a trabajar con ellos. Posteriormente, aparecieron herramientas cuyo objetivo era el procesado de estos datos, como: *Pig*, *Hive* y *Spark*.

Es en este punto donde las empresas descubrieron que la gran cantidad de datos con los que contaban se debía a que los recibían de manera continua en forma de flujos, por lo que el procesado por lotes no bastaba para extraer el máximo potencial a dichos datos. Por tanto, aparecen las soluciones de *streaming*, destacando *Apache Spark Streaming, Apache Storm* y, por supuesto, *Apache Flink*.

Apache Flink es un *framework* de código abierto desarrollado por la *Apache Software Foundation*, cuya primera versión oficial data del año 2015. *Flink* permite el procesado de datos tanto en flujos como en lotes. Cabe destacar que las soluciones para el procesado de flujos de datos deben enfrentarse a una serie de problemas que no afectaban a las herramientas de procesado por lotes [24]:

- Las transformaciones (p.e.: troceado de palabras) y el procesamiento de los datos deben tener en cuenta que estos llegarán de manera indefinida, ya no se puede esperar a leer todas las líneas para empezar a tratar la información.
- En caso de indisponibilidad del servicio, se deben disponer de mecanismos para no perder ningún dato del flujo. Es necesario separar la etapa de adquisición y la de procesamiento.
- Existen requisitos de tiempo de procesamiento.

[2]https://yarnpkg.com/lang/en/
[3]https://spark.apache.org/
[4]https://spark.apache.org/docs/1.6.2/api/java/org/apache/spark/rdd/RDD.html

- Resistencia frente a fallos que asegure que cada evento sólo se procesa una vez (mecanismos de sincronización y control de errores).

La principal diferencia entre *Flink* y tecnologías similares es que su implementación es la que, según varios estudios, mejor resuelve dichos problemas sin perder capacidad y velocidad de procesamiento. *Flink* ofrece un procesamiento *streaming* nativo, mientras que *Spark*, quien fue ideado en un principio sólo para el procesamiento de datos por lotes, se ha adaptado al procesamiento en *streaming* utilizando *micro-batching*.

En este punto, cabe recordar que la mejor tecnología siempre será la que mejor se adapte a las condiciones del problema concreto que se esté resolviendo [25] y puede ser una tecnología u otra según el caso. De todas formas, lo cierto es que *Flink* presenta grandes ventajas en cuanto al procesamiento de flujos de datos, pero todavía no cuenta con la gran comunidad de usuarios que posee, por ejemplo, *Apache Spark*. Por este motivo, este trabajo pretende ser un estudio más que determine las posibilidades de esta plataforma *Big Data*.

Capítulo 5

Diseño e implementación

Este capítulo está dedicado a describir tanto el diseño de la aplicación como aspectos vinculados a su fase de implementación y a su funcionamiento.

5.1. Diseño

A continuación, se tratarán cuestiones relacionadas con el diseño de la aplicación, describiendo su estructura interna mediante una serie de diagramas y figuras que pretenden facilitar su comprensión.

5.1.1. Arquitectura

El flujo de trabajo de *FlinkBWA* se basa en dos fases principales: el mapeo de los datos y la reducción de los mismos. En *Flink*, los *Dataset* funcionan como los RDDs de *Spark* y, en este caso, se forman a partir de los ficheros FASTQ de entrada que son almacenados usando HDFS. Cabe mencionar que se asume HDFS como un sistema de ficheros distribuido, repartiendo los datos entre los distintos nodos de computación para poder procesarlos en paralelo durante la fase de mapeo. Además, los identificadores de los ficheros FASTQ (mencionados en la fase de análisis, capítulo 4, concretamente en la figura 4.1) son utilizados como clave de los Dataset.

Pero, como ya se explicó con anterioridad, el BWA original también es capaz de tomar como entrada dos archivos FASTQ, en cuyo caso deberá crearse un *Dataset* por fichero, los cuales serán distribuidos a través de los nodos. En este caso no existen garantías de que ambas estructuras de datos sean procesadas por un mismo mapeador. Para este problema se presentan dos operaciones como solución:

- **Join y *sortByKey*:** consiste en utilizar la operación *join* de Flink para combinar dos Datasets, agrupando los elementos cuya clave coincide. Si

los ficheros de entrada tienen la misma clave, tras el *join* serán un mismo Dataset, el problema es que no se conserva el orden inicial de los ficheros. De todas formas, con la operación *sortByKey* se volverán a ordenar según esta clave. El problema es que esta última operación es muy costosa en cuanto a consumo de memoria.

- **SortHDFS:** esta operación puede considerarse como parte del preprocesado ya que se reordenan los archivos FASTQ en un mismo fichero HDFS. Para distinguir una secuencia de otra y facilitar el mapeo, se utiliza un *string* como separador. Esta solución es costosa temporalmente hablando, pero ahorra gran cantidad de memoria en comparación con la primera.

A continuación, la figura 5.1 incluye un esquema genérico de la arquitectura de *Flink* trabajando en un entorno *Hadoop*, como en nuestro caso. Dicha figura muestra el procesado de dos Dataset sobre los que posteriormente se aplica una operación de tipo *join* para combinarlos:

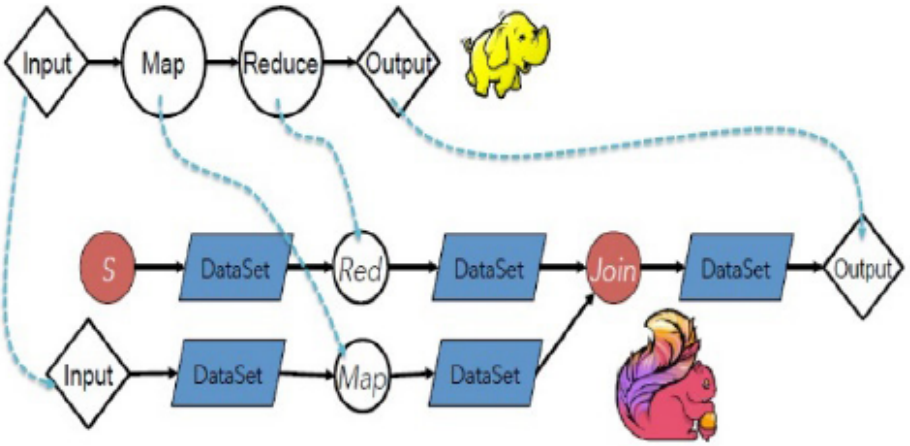

Figura 5.1: Arquitectura de *Flink*

5.1.2. Diagrama de clases

En la figura 5.2 se pueden observar las principales clases que conforman la aplicación y cómo se relacionan entre sí:

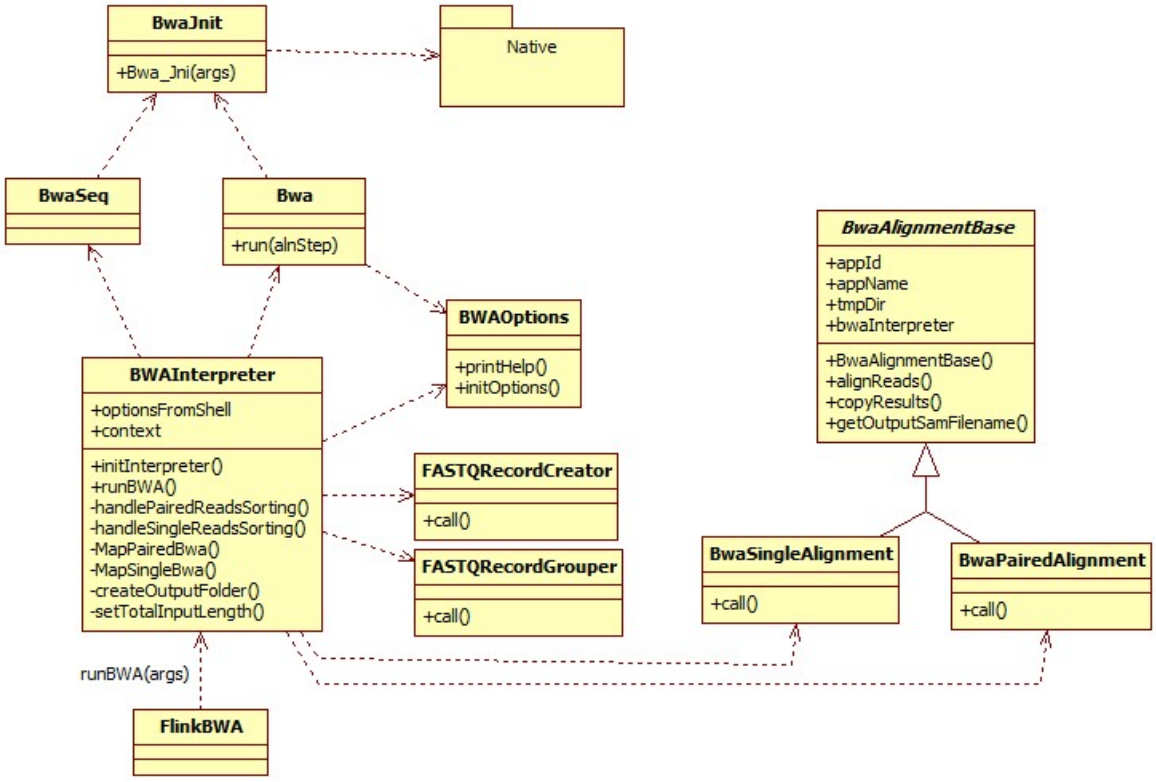

Figura 5.2: Diagrama de clases

El paquete "native" que aparece en la figura 5.2 hace referencia al código pertinente a la aplicación original del BWA que, como ya se mencionó anteriormente, no ha sido alterada. Este paquete simplemente aparece en el diagrama para indicar que la clase *BwaJnit* es la encargada de comunicarse con la aplicación nativa, haciendo uso de la *Java Native Interface* (JNI), la cual permite incorporar código nativo en C/C++. De esta forma, se consigue separar la implementación original del BWA escrita en C del código de la nueva aplicación que incluye *Flink*, plataforma que sólo interpreta los lenguajes Java, Scala y Python (pese a que no existe aun ninguna versión estable que soporte este último).

Por otra parte, *BWAInterpreter* funciona como una clase "controlador", puesto que es la encargada de invocar a las correspondientes funciones según los datos introducidos y las opciones señaladas. Básicamente es la clase encargada de controlar el resto de clases, como ya se puede interpretar a partir de las dependencias que muestra el diagrama.

5.2. Implementación

Capítulo 6

Pruebas de rendimiento

Capítulo 7

Conclusiones

Apéndice A

Manuales técnicos

Manuais técnicos: en función do tipo de Traballo e metodoloxía empregada, o contido poderase dividir en varios documentos. En todo caso, neles incluirase toda a información precisa para aquelas persoas que se vaian a encargar do desenvolvemento e/ou modificación do Sistema (por exemplo código fonte, recursos necesarios, operacións necesarias para modificacións e probas, posibles problemas, etc.). O código fonte poderase entregar en soporte informático en formatos PDF ou postscript.

Apéndice B

Manuales de usuario

Manuais de usuario: incluirán toda a información precisa para aquelas persoas que utilicen o Sistema: instalación, utilización, configuración, mensaxes de erro, etc. A documentación do usuario debe ser autocontida, é dicir, para o seu entendemento o usuario final non debe precisar da lectura de outro manual técnico.

Apéndice C

Licencia

Se se quere pór unha licenza (GNU GPL, Creative Commons, etc), o texto da licenza vai aquí.

Bibliografía

[1] Gantz, J. and Reinsel, D. (2012). Executive Summary: A Universe of Opportunities and Challenges. Artículo de EMC (*https://www.emc.com/leadership/digital-universe/2012iview/executive-summary-a-universe-of.htm*). Consultado el 1 de febrero del 2019.

[2] Documentación sobre el *Burrows-Wheeler Aligner* (BWA). Página de *BWA* (*http://bio-bwa.sourceforge.net/*). Consultada el 20 de febrero del 2019.

[3] Documentación completa sobre *Apache Flink*. Página oficial de *Apache Flink* (*https://flink.apache.org/*). Consultada el 29 de enero del 2019.

[4] José M. Abuin, Juan C. Pichel, Tomás F. Pena and Jorge Amigo. "SparkBWA: Speeding Up the Alignment of High-Throughput DNA Sequencing Data". PLoS ONE 11(5), pp. 1-21, 2016.

[5] Project Management Institute, Guía de los Fundamentos para la Dirección de Proyectos (Guía del PMBOK), 5ª edición.

[6] Calendario de lectura de TFG 2018-2019. (*http://www.usc.es/etse/files/u1/CALENDARIOLECTURA_TFG_GREI_CURSO1819.pdf*). Consultada el 21 de febrero del 2019.

[7] Principios del Manifiesto Ágil (*https://agilemanifesto.org/iso/es/principles.html*). Consultada el 22 de febrero del 2019.

[8] Guía de *Scrum*. *Scrum Guide* (*https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-es.pdf*). Consultada el 22 de febrero del 2019.

[9] Táboas de retribucións de Personal Docente e Investigador. USC. (*http://www.usc.es/gl/servizos/sxp/taboas/tabPDI19.html*). Consultada el 27 de febrero de 2019.

[10] Bases y tipos de cotización 2019. Seguridad Social. (*http://www.segsocial.es/wps/portal/wss/internet/Trabajadores/CotizacionRecaudacionTrabajadores/36537*). Consultada el 27 de febrero de 2019.

[11] Justificación de Costes Indirectos. USC. (*http://www.usc.es/export9/sites/webinstitucional/es/congresos/xiiiencontroredeugi/descargas/COSTES_INDIRECTOS_AEI_Carmen_Penas_Justificacion_Costes_Indirectos.pdf*). Consultada el 26 de febrero de 2019.

[12] Sitio oficial de UML. (*http://www.uml.org/*). Consultada el 27 de febrero de 2019.

[13] Convention on Biological Diversity. United Nations (1992). (*https://www.cbd.int/doc/legal/cbd-en.pdf*). Consultada el 9 de marzo de 2019.

[14] Apuntes de Bioquímica I. Universidad Complutense de Madrid (UCM). *http://webs.ucm.es/info/biomol2/bioquimicaI/WTA/Homologia.html*. Consultada el 9 de marzo de 2019.

[15] Biotecnología para no iniciados, Capítulo III. Genetaq, Centro de Genética Molecular (2015). *http://genetaq.com/es/blog/bioinformatica-para-no-iniciados-capitulo-iii*. Consultada el 9 de marzo de 2019.

[16] M. Burrows, D.J. Wheeler (May 10, 1994). "A Block-sorting Lossless Data Compression Algorithm". (*https://www.hpl.hp.com/techreports/Compaq-DEC/SRC-RR-124.pdf*). Consultada el 10 de marzo de 2019.

[17] Li H. and Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics, 25:1754-60. [PMID: 19451168].

[18] Li H, Durbin R. Fast and Accurate Short Read Alignment with Burrows-Wheeler Transform. Bioinfor- matics. 2009; 25(14):1754–1760. doi: 10.1093/bioinformatics/btp324 PMID: 19451168.

[19] Li H, Durbin R. Fast and Accurate Long-Read Alignment with Burrows-Wheeler Transform. Bioinformat- ics. 2010; 26(5):589–595. doi: 10.1093/bioinformatics/btp698 PMID: 20080505.

[20] Li H (2013). Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv:1303.3997.

[21] The SAM/BAM Format Specification Working Group (2019). "Sequence Alignment/Map Format Specification". (*http://samtools.github.io/hts-specs/SAMv1.pdf). Consultada el 10 de marzo de 2019.*

[22] Rajkumar Buyya, Rodrigo N. Calheiros y Amir Vahid Dastjerdi. Big Data: Principles and Paradigms, 3ª edición, Morgan Kaufmann (Elsevier), 2016.

[23] Niño, M., Illarramendi, A.. (2015). (UNDERSTANDING BIG DATA: ANTECEDENTS, ORIGIN AND LATER DEVELOPMENT). DYNA New Technologies, 2(1). [8 p.]. DOI: http://dx.doi.org/10.6036/NT7835.

[24] José Carlos Baquero, Pablo González. El futuro de las tecnologías de Streaming: Apache Flink. https://openexpoeurope.com/es/tecnologias-streaming-apache-flink/.Consultada el 12 de marzo de 2019.

[25] Jeyhun Karimov, Tilmann Rab, Asterios Katsifodimos, Roman Samarev, Henri Heiskanen, Volker Markll. "Benchmarking Distributed Stream Processing Engines". (https://arxiv.org/ftp/arxiv/papers/1802/1802.08496.pdf). Consultada el 14 de marzo de 2019.