INSTITUTO TECNOLÓGICO DE COSTA RICA
ÁREA DE INGENIERÍA EN COMPUTADORES
ALGORITMOS Y ESTRUCTURAS DE DATOS I (CE 1103)

ANEXO DEL PROYECTO

Profesor:
José Isaac Ramírez

Estudiantes:
Rodríguez Pérez Adrián Fernando
2016005123
Zamora Espinoza Gustavo Andrés
2017137089
Lopez Yaritza
2019039508
Valverde Ureña Sofía
2020161005
Lorenzo Ignacio
2019068171

25 de Junio, 2021

I Semestre 2021

Tabla de contenidos

Introducción

Para el tercer proyecto del curso Estructuras y Algoritmo de Datos I de la carrera “Ingeniería en Computadores” se solicitó el diseño e implementación de un REST API con formato JSON y en c#(Utilizando .NET 5).Este tiene como propósito modelar grafos a partir de información ingresada por el usuario.Los datos en este caso están centrados en la relación de llamadas telefónicas, el nombre de las personas partícipes de la llamada y la duración de la llamada. En este documento se analizarán los métodos utilizados para el desarrollo de la aplicación, así mismo la división de las tareas entre los integrantes del grupo y restricciones necesarias para llevar a cabo el trabajo.

Según los requerimientos impuestos en el instructivo se tenía como tareas principales: Todo lo relacionado con implementación y creación de REST api y el formato JSON, la implementación de grafos en c#, junto a todas las funciones que son necesarias.

Centrados más en el grafo este debe de contener tres partes fundamentales: Nodo, que esto se refiere a los objetos que contienen los nombres de las personas partícipes en la llamada; Edge, que hace referencia al tiempo de llamada entre las personas; y el grafo el cual contiene los elementos anteriores.

A partir de esta información se dividió en tareas por medio de Github para que cada integrante pudiera avanzar por su cuenta en las distintas áreas del proyecto, así optimizando el tiempo. Se realizaron reuniones de grupo y entre compañeros independientemente para explicar los avances y para trabajar conjuntamente para solucionar errores y comprender el trabajo completo.

Diseño del proyecto

Historias de usuario

- Yo como usuario/a quiero poder importar un archivo CSV para construir un grafo que haga un análisis del contenido.
- Yo como usuario/a , una vez creado los grafos, quiero poder agregar nodos para analizar datos previos con los nuevos.
- Yo como usuario/a quiero ser capaz de actualizar Edges, Nodos y Graphs,para tener más control de las posibles eventualidades que se den en el transcurso de los análisis.
- Yo como usuario/a quiero ser capaz de agregar y/o eliminar Edges para crear nuevos enlaces de análisis.
- Yo como usuario/a quiero poder actualizar y/o eliminar Nodos para actualizar información en análisis o borrar información de esta.
- Yo como usuario/a quiero que el grafo que se forme muestre los nodos como teléfonos con su id y el tamaño de enlace entre cada nodo, dependiendo de la duración de la llamada.
- Yo como usuario/a quiero poder eliminar un grafo o todos los grafos para poder filtrar análisis con mayor facilidad
- Yo como usuario/a quiero poder observar los grafos y la información y partes que este contiene para hacer debidamente los análisis.

Diagrama de solución del problema

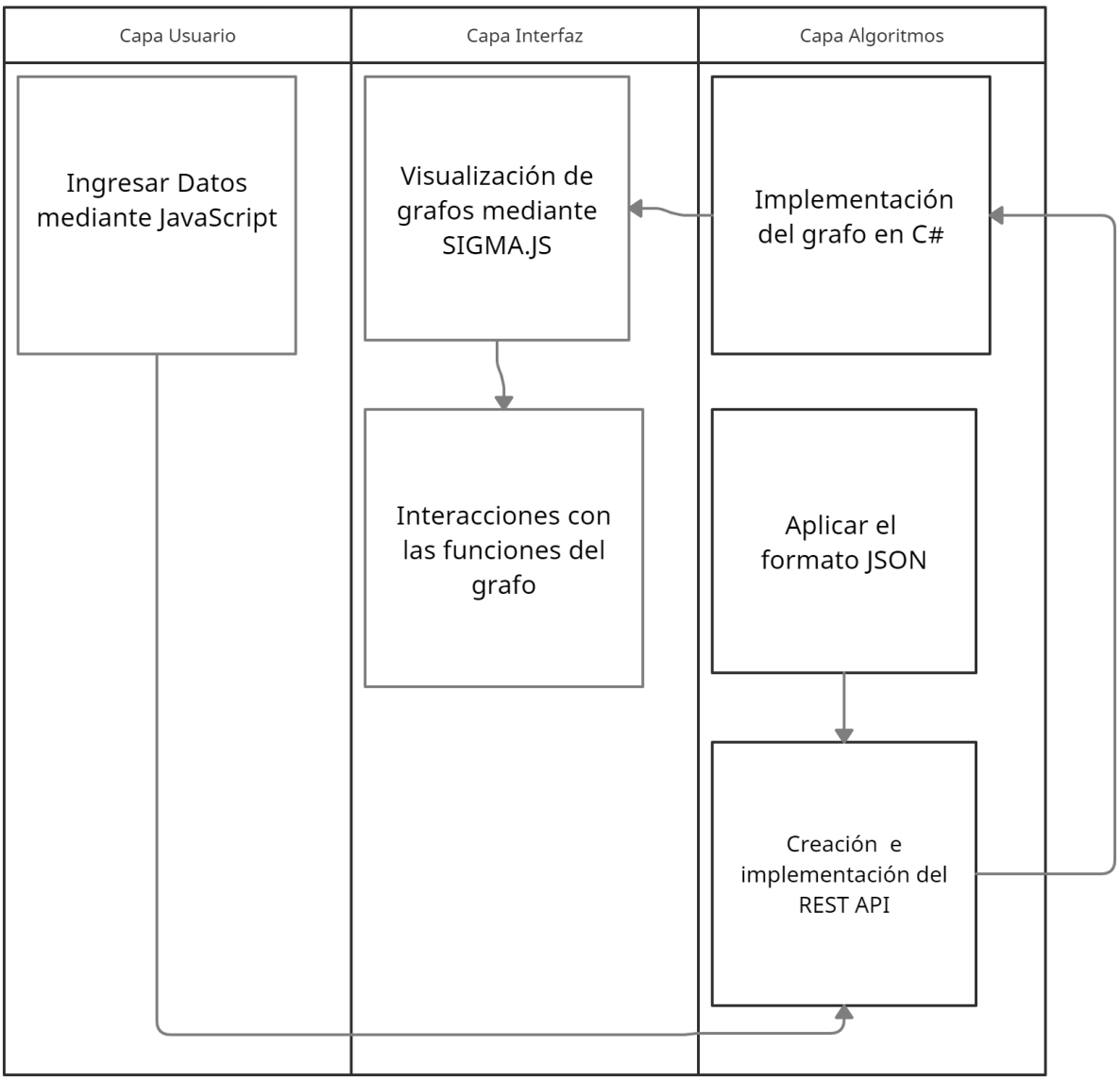

Diagrama de arquitectura de la solución

Trabajo en equipo

Para el primer plan de iteraciones se buscó cumplir con las tareas más generales del proyecto, las propiedades que más importante eran y dividirlas entre los estudiantes del proyecto.
Prioritariamente se basaba en investigar acerca del funcionamiento de aquellos formatos con los que no se contaba experiencia alguna, las tareas seleccionadas para el primer plan fueron:

- Conocer el funcionamiento de Postman
- Investigar sobre Sigma y JavaScript
- Investigar sobre implementación de grafos en c#
- Investigar sobre diseño e implementación de REST API con c#
- Investigar sobre el formato JSON y cómo implementarlo en la aplicación

La división de tareas fue más enfocada a realizar la implementación del graph y todo lo necesario para este como los nodos y edge, además de varias rutas http para ver su funcionamiento en Postman, debido a un mal entendido y mala comunicación dos integrantes hicieron lo mismo, estos estudiantes fueron Ignacio Lorenzo y Sofía Valverde, en donde se decidió utilizar el código de Sofía ya que estaba mejor hecho y más completo. Se logró la implementación del api exitosamente.
Todos los estudiantes realizaron una investigación sobre implementación del REST API y de c# para entender cómo funcionaba el proyecto y poder ayudar en errores pequeños que surgieran en el transcurso del proyecto
Las historias del usuario que se tomaron en cuenta para el primer pal fueron:

- Yo como usuario/a quiero poder eliminar un grafo o todos los grafos para poder filtrar análisis con mayor facilidad
- Yo como usuario/a quiero poder actualizar y/o eliminar Nodos para actualizar información en análisis o borrar información de esta.
- Yo como usuario/a quiero ser capaz de agregar y/o eliminar Edges para crear nuevos enlaces de análisis.Yo como usuario/a quiero ser capaz de actualizar Edges, Nodos y Graphs,para más control de las posibles eventualidades que se den en el transcurso de análisis.
- Yo como usuario/a , una vez creado los grafos, quiero poder agregar nodos para analizar datos previos con los nuevos.

- Yo como usuario/a quiero ser capaz de actualizar Edges, Nodos y Graphs, para tener más control de las posibles eventualidades que se den en el transcurso de los análisis.

Para el segundo plan de iteraciones se continuó realizando investigaciones en este caso para ciertas funcionalidades más específicas de Sigma para poder graficar los grafos creados a través del REST API que tenemos creado en c#, también se mencionó para investigar dos funcionalidades para el grafo las cuales serian

- Degree: Que ordenaría los nodos por su grado promedio en el orden indicado por el parámetro fuera DESC o ASC, se plantea la idea de que el grado promedio sea la suma de los Edge que tiene enlazados a él, para saber en este caso cuanto es la duración de llamada y organizarlo por tiempo de llamada
- Dijkstra : se obtendría la ruta más corta entre dos nodos, el parámetro a pasar serían el id de 2 nodos y el resultado sería el valor total del camino más corto entre estos 2 nodos

Las historias de usuario anteriormente no contienen exactamente estas dos funcionalidades del grafo por lo que se añadieron 2 nuevas historias de usuario al proyecto las cuales serian :

- Yo como usuario/a quiero poder tener una función para ordenar según el tiempo de llamada los nodos, para saber cuánto tiempo de llamada ha hecho en total cada persona para el análisis
- Yo como usuario/a quiero poder tener una función para saber el camino más corto entre dos nodos, siendo el camino el tiempo de las llamadas

Para el tercer plan de iteraciones se probó que funcionara lo que teníamos hasta el momento del API, funciono todo hasta el momento, en esto también se probaron y sirvió el Degree y el Dijkstra, con la implementación realizada se comenzó a plantearse el siguiente paso que sería implementar Sigma para poder graficar los grafos, investigando y probando con Sigma se encontraron varios problemas o

contratiempos, entre ellos el tener que usar css y html para poder hacer funcionar sigma, además de tener que entender lo básico de javascript para que funcionara la librería de sigma, además se tiene que investigar cómo unir la página creada en html con todo lo creado en C# para que se pueda llamar los grafos y que estos puedan aportar los datos a la librería para poder graficar, teniendo el mayor problema el no saber como conectar C# con html para pasar los datos de los grafos que están guardados en el programa de c# las tareas seleccionadas para este plan de interacciones fue:

- Investigar sobre HTML y su funcionamiento con otros lenguajes como c# y javascript
- Investigar más a fondo sobre Sigma.js y su funcionamiento
- Investigar cómo agregar una pagina a c# hecha en html
- Investigar cómo usar funciones o llamar objetos/propiedades hechas en c# desde HTML

Las historias de usuario tomadas para este plan de iteracciones fueron:

- Yo como usuario/a quiero poder importar un archivo CSV para construir un grafo que haga un análisis del contenido.
- Yo como usuario/a quiero que el grafo que se forme muestre los nodos como teléfonos con su id y el tamaño de enlace entre cada nodo, dependiendo de la duración de la llamada.
- Yo como usuario/a quiero poder observar los grafos y la información y partes que este contiene para hacer debidamente los análisis.

Ignacio Lorenzo Martínez

-06/06/2021
Se realizaron investigaciones sobre c# y REST API para conocer un poco más sobre estos y decidir cual IDLE utilizar para más facilidad del proyecto, además se vieron varios IDLES para poder utilizar, entre estos Visual Studio Code, Visual Studio y Rider. Se inició el primer intento del REST API con dotnet new wepapi, pero dando errores en Visual Studio, se busco la manera de arreglarlo en google.com pero no se logro arreglar el error
-10/06/2021
Se arreglo el error para crear nuevos proyectos con dotnet new wepapi (.NET 5), para arreglar esto se utilizo Visual studio y se instaló varios complementos para que pudiera funcionar, aunque a momento sigo utilizando Visual studio code para adelantar el proyecto
-11/06/2021
Debido a un conocimiento básico y no poder revisar los errores fácilmente para la implementación del API elegí utilizar rider ya que las herramientas ayudan un poco más al arreglo de errores de novato.
Se crearon diferentes clases como Graph,Node,edge y LinkedList y LinkedListNode, además de investigar la posibilidad de utilizar Hashmap y UUID en c#, los cuales serían Dictionary y GUID correspondientemente, esto para facilitar el guardado de graph y utilizar un ID único para estos
-13/06/2021
Se inició el proyecto en github para poder ver, compartir y unir el código con los compañeros.
-15/06/2021
Se continuó creando nuevas funciones para LinkedList,graph y las otras clases, funcionando las funciones iniciales de HTTP como get y post, tanto para graph,node y edge
-19/06/2021
Se inició una investigación sobre Sigma.js para decidir si usar esta librería de javascript con la el REST API creado en c#, esto para poder graficar los grafos, se vieron ejemplos de sigma y se encontró también información sobre vis.js que es otra librería para graficar y poder visualizar los grafos, se inició varias pruebas con los 2 para determinar cuál sería el mejor y que mejor se acople al proyecto

Adrian Rodríguez Perez

- 12 de Junio: Investigación sobre C#, REST, sus aplicaciones en la industria y técnicas y convenciones para realizar el proyecto, además de los tipos de graphs especificados (POST, GET, DELETE y PUT).
- 13 de Junio: Investigación sobre algoritmos para traversar grafos y los más aptos para la aplicación. Además, se investigó sobre estándares y normas acerca de estos y la aplicación del proyecto en general.
- 19 de Junio: Pruebas y modificaciones a funciones de grafos, controllers y singletons.
- 20 de Junio: Desarrollo y formato inicial del anexo.
- 25 de Junio: finalización del anexo y revisión del funcionamiento.

Sofía Valverde Ureña

- 14 de Junio: Investigación sobre C# y grafos.
- 15 de Junio: Inicio del desarrollo de la implementación API, iniciando con la creación de los grafos.
- 16 de Junio: Terminar los Post, Get, Delete y Put de los grafos.
- 18 de Junio: Creación de los Post, Get y Delete de los nodos.
- 19 de Junio: Creación de los Post, Get, Delete y Put de los edges. Este día se sube todo lo que se había creado anteriormente.
- 20 de Junio: Se revisa que todo funcione de forma correcta y se realizan ciertos cambios a los nodos y edges. Se actualizan los cambios en Git y se da por concluida la implementación del API.

Yaritza Lopez Bustos

- 11 de Junio: Se realizó la investigación de REST y HTTP, C# y grafos para poder realizar el diseño y la implementación de RESTAPI

- 16 de Junio: Se ayudó en la creación de los grafos.
- 18 de Junio: Creación de los Post, Get y Delete de los nodos.
- 19 de Junio: Se revisa que todo funcione de forma correcta y se realizan ciertos cambios a los nodos y edges.
- 20 de Junio: Pruebas y modificaciones a post, get y delete de los nodos y se desarrolló parte del documento de anexo.
-
- 25 de Junio: Se finalizó el documento de anexo.

Gustavo Zamora Espinoza

- 10 de Junio: Se investigó sobre C# y cómo crear proyectos con dicho lenguaje. Además se estudiaron los grafos para entender mejor su funcionamiento y propósito.
- 12 de Junio: Se investigó acerca de el REST API y cómo desarrollarlo para modelar grafos.
- 13 de Junio: Se buscó información sobre el formato JSON y cómo es utilizado para el intercambio de datos.
- 15 de Junio: Se realizó una pequeña investigación sobre las posibles librerías de Javascript y así poder decidir cuál utilizar.
- 18 de Junio: Pruebas y solución de errores encontrados.
- 20 de Junio: Se inició y desarrolló el Anexo.
- 25 de Junio: Se finalizó el Anexo.