DOCUMENTO DE DISEÑO ARQUITECTÓNICO

IIS – UNED – G. ING. TI

ABRIL DE 2020

JORGE ARROJO MACIAS

Índice

INTRODUCCIÓN

Objetivo

Se pretende desarrollar una aplicación que simule la propagación de un virus de manera exponencial en diferentes poblaciones o comunidades.

Ámbito

El sistema que se desarrollará se llamará Simulación Vírica Exponencial y consistirá en un único programa que realizará todas las funciones necesarias. En particular deberá realizar lo siguiente:

- Una simulación de la expansión de un virus en una o varias poblaciones siguiente la siguiente fórmula matemática:
  - $N_{d+1} = N_d + N_d * E * p$
    - Siendo:
      - N_d = Número de infectados en el día
      - E = Número de contactos entre personas
      - p = Probabilidad de contagio
- Además, existirá un número de viajeros (V) entre las diferentes poblaciones o comunidades que será el mismo para todas ellas, calculándose N_v que será el número de infectados a causa de viajeros o de importaciones del virus. Seguirá la siguiente fórmula:
  - $N_v = E * p * V * N_{d\_origen}/P_{origen}$
    - Siendo:
      - P_{origen} = Población en comunidad de origen
      - N_{d_origen} = Número de infectados en la comunidad de origen del viajero
      - V = Número de viajeros diarios de una comunidad
- Se introducirán por teclado los siguientes datos para llevar a cabo cada simulación particular:
  - Número de comunidades, su población y su nombre
  - Porcentaje de habitantes de cada comunidad que viajan (Será igual para todas las comunidades)
  - E, p, V
  - Número de días a simular

Queda resaltado que el número de viajeros en cada comunidad se introducirá por teclado y será el mismo para todas las comunidades.

El programa dispondrá de una Interfaz de Usuario textual.

Definiciones, siglas y abreviaturas

Ninguna

Referencias

Ninguna

Panorámica del Documento

Ver índice al comienzo del mismo

PANORAMICA DEL SISTEMA

La aplicación simulará la expansión exponencial de un virus en base a los coeficientes indicados en el apartado anterior. Mostrará los resultados de manera pausada mediante un menú de resultados. Estos resultados serán:

- Número total de contagiados
- Porcentaje de contagiados de la población total
- Porcentaje de contagiados para cada una de las comunidades

Estos resultados se mostrarán en una tabla ordenada en filas y columnas que facilite su visualización.

Al finalizar la simulación, se le dará la opción al usuario de realizar otra simulación introduciendo nuevos valores como coeficientes del calculo matemático de la simulación.

Requisitos Funcionales

Todos los requisitos se consideran obligatorios salvo que se exprese lo contrario

- Función Interfaz textual de usuario
  - Se dispondrá de una interfaz textual de usuario a través de menús que permitan la operatividad con la aplicación.
  - Los resultados de la simulación deberán mostrarse de forma pausada. Para ello recurriremos a un menú textual en el que el usuario escoja qué resultado desea mostrar, volviendo de nuevo al menú y ofreciendo éste la posibilidad de una nueva simulación o la salida del programa
  - Se establecerá un Menú inicial con dos opciones: 1) **Introducir Datos**; que permitirá comenzar a introducir los datos necesarios para llevar posteriormente a cabo la simulación, y 2) **Salir**, que permite al usuario finalizar la ejecución del programa
  - También se establecerá un menú intermedio, siempre y cuando hayan sido almacenados correctamente (y validados) todos los datos necesarios que ofrecerá al usuario las siguientes opciones: 1) **Ver datos almacenados**; El usuario podrá revisar los datos que ha introducido antes de llevar a cabo la simulación. 2) **Volver a Introducir Datos**; el sistema permitirá de este modo volver a introducir los datos necesarios para la simulación en caso de una debida corrección de los datos introducidos con anterioridad. 3)**Comenzar simulación**; el usuario podrá comenzar la simulación con los datos almacenados.
- Función Introducción de Datos
  - El programa debe permitir la introducción de los coeficientes necesarios (datos) para llevar a cabo cada simulación concreta.
  - El programa debe contemplar la posibilidad de existencia de más de una comunidad/población

- El programa debe contemplar un número de viajeros expresado en porcentaje de una población, que viajarán de una población a otra y tendrán un efecto en los resultados finales de la simulación de acuerdo a una expresión matemática que los regulará (ver *Introducción>ámbito*)
- El sistema debe validar los datos antes de que éstos sean almacenados de acuerdo con las siguientes restricciones:
  - E (número de contactos): >=0
  - V (% de viajeros): >=0
  - Población (de una comunidad): >0
  - Días de simulación: >0
  - P (probabilidad de contagio): 0<=p<=1

➢ Función Simulación

- El programa debe realizar los cálculos de acuerdo con los coeficientes introducidos en la introducción de datos de acuerdo a lo previsto en el apartado *Introducción* de este mismo documento.
  - El sistema calculará el número de infectados por comunidad y almacenará este dato en memoria.
    - El sistema calculará la muestra representativa de este dato en porcentaje y la almacenará.
  - El sistema calculará el número de infectados debido a los viajeros que se introduzcan en la fase de introducción de datos.
    - El sistema calculará la muestra representativa de este dato expresada en porcentaje y la almacenará en memoria.
  - El sistema calculará el número total de infectados y almacenará este dato en memoria.

➢ Función Muestra de Resultados

- El programa debe mostrar los resultados de forma pausada. Para ello se elaborará un menú textual con tres opciones: 1) **Mostrar número total de infectados**, 2) **Mostrar porcentaje de infectados en función del total de población**, 3) **Mostrar el porcentaje de infectados en función de la población de cada comunidad**. También existirán las siguientes opciones de usuario en este menú: 4) **Repetir simulación**; esta opción permitirá al usuario introducir nuevos datos para llevar a cabo una nueva simulación con los nuevos datos introducidos. 5)**Salir**, el usuario podrá finalizar la ejecución del programa a través de esta opción.

CONTEXTO DEL SISTEMA

NO APLICABLE

DISEÑO DEL SISTEMA

Metodología de Diseño de Alto Nivel

Para diseñar esta aplicación partiremos de diagramas estructurales tales como el diagrama de clases y el diagrama de Casos de Uso. Éstos nos darán una visión estática del programa y propondrán un acercamiento al espacio de la solución sin llegar a ser diagramas de implementación en su totalidad. Estarán enfocados en un nivel de abstracción media que permita la posterior elección de las estructuras de datos y de almacenamiento de los mismos, conforme al lenguaje de programación posteriormente escogido.
Acompañará a estos diagramas estructurales, un diagrama dinámico. En concreto un diagrama de secuencia que nos dará cuenta de la evolución de los objetos implicados a lo largo del ciclo de vida de la aplicación, desde que el usuario lo ejecuta hasta que la simulación termina y permite realizar otra nueva simulación introduciendo de nuevo, nuevos coeficientes.

Descomposición del Sistema

El sistema está compuesto por los siguientes componentes:

Abstracciones
Lanzador (main)
<<static>>Menus (interna)
IntroduccionDeDatos
Comunidad
Simulación
<<static>>MenuResultados (interna)

Existirán relaciones de dependencia entre todas las abstracciones salvo la relación Simulación-Comunidad que será una relación de asociación ya que la abstracción Simulación necesita de la abstracción Comunidad para llevar a cabo su función. Podrá verse visualmente este modelado en el diagrama de clases correspondiente que acompaña este documento.

Además, modelaremos una clase Lanzador, en caso de ser de necesidad para Lenguajes de Programación que precisen una Clase Principal o Main, desde donde arranque toda la ejecución del programa.

DISEÑO DE LOS COMPONENTES

En cuanto a los datos participantes tenemos varios: E, p, V, díasSimulación, Comunidad y su población, N_d, N_v y TotalDeInfectados.

Ubicados estos datos en sus abstracciones correspondientes, disponemos lo siguiente:

Abstracciones - *Datos*
Lanzador – Comunidades, simu
Menus
IntroduccionDeDatos – *E, p, díasSimulación, NumComunidad, Comunidad[0..*]*
Comunidad – *población, V, Nombre*
Simulación – N_v, N_d, *TotalDeInfectados, PorcentajeInfectadosPorComunidad, totalPorcentajeInfectados*
MenuResultados

Tipos de Datos:

Tipos de Datos
Comunidades, simu, Comunidad – **IntroduccionDatos, Simulación, Comunidad**
E, díasSimulación, Población, NumComunidad**: Entero**
:p, N_v, N_d, *V:* ***Coma Flotante de amplio rango***
Nombre: ***CadenaDeTexto***
*TotalDeInfectados, PorcentajeInfectadosPorComunidad, totalPorcentajeInfectados:****Coma Flotante de amplio rango***

Objetivo y Función de cada componente:

Abstracciones	**Objetivo y Función**
Menus	Mostrar **MenuInicio**: Podrá comenzar la introducción de los datos/coeficientes necesarios para la simulación, o salir de programa **MenúPreSimulación**: Una vez todos los datos estén introducidos, se mostrara al usuario un menú donde podrá **1)**ver los datos almacenados. **2)** Volver a introducir los coeficientes dando la opción de corregir errores de entrada de datos. **3)**Comenzar la Simulación. Forma parte de la UI
IntroduccionDeDatos	Permitir la introducción de los coeficientes y datos necesarios para llevar a cabo la simulación. Forma parte de la UI.
Comunidad	Establece una estructura para guardar datos con una estructura concreta adecuada a los requerimientos, con los datos anteriormente descritos.
Simulación	Realizará los cálculos necesarios en función de los coeficientes introducidos y las fórmulas anteriormente indicadas en este documento Almacenará los Resultados y Calculará porcentajes y Totales de infectados.
MenuResultados	Permitirá al Usuario una vista pausada y secuencial de los resultados (UI): 1) Numero total de infectados 2)Porcentaje de la población total infectada 3)Porcentaje de infectados en cada comunidad

Descripción de operaciones:

ABSTRACCIÓN: IntroducciónDatos
OPERACIÓN: pedirComunidad(): void
REQUISITOS: La operación debe permitir introducir los datos pertinentes según la estructura de información que establece la clase Comunidad. Deben indicarse e introducirse por teclado, el número de comunidades que intervienen en la simulación. El porcentaje de viajeros entre comunidades (será igual para todas las comunidades), expresado con valores entre [0-1] en coma flotante. Introducir un nombre identificativo para cada comunidad, introducir un valor entero representativo de la población de cada comunidad. La operación almacenará la información en una colección de tipo de Datos Comunidad para su posterior recuperación y tratamiento. Esta colección será un campo de la abstracción, permitiendo a través de métodos getter, tomar la referencia de dicha colección en dicha abstracción para su uso en clases que lo precisen, como por ejemplo Simulación.
FUNCIÓN: La operación va a realizar la petición de datos al usuario por la entrada de teclado estándar y repetirá esta petición siempre que los datos no se ajusten a los valores permitidos, es decir, la operación no solo permite la introducción de datos para formar una instancia de tipo Comunidad, sino que antes de formarla, validará esos datos para que el objeto creado cumpla con las restricciones de los datos de esta estructura de datos, Comunidad, para su adecuado funcionamiento en la abstracción Simulación. Los casos de flujo alternativo están regulados mediante excepciones en operaciones de entrada de datos por la entrada estándar, donde se repite la introducción hasta que el valor sea adecuado, y esta operación, mediante las validaciones de los datos documentadas en los siguientes puntos, no permitiendo la entrada de valores fuera de las restricciones estipuladas y obligando al usuario a introducir nuevamente el dato inválido.
La operación seguirá los siguientes pasos:

1. Petición del numero de comunidades que intervienen en la simulación; validación de que este dato sea un número mayor que cero, y almacenaje de este entero (int) como campo de la abstracción para su posterior uso en otras abstracciones.
2. En el caso de que el número de comunidades sea igual a 1, no es necesario que el usuario introduzca el número de viajeros entre comunidades, y la operación lo establecerá en 0. En caso contrario, numero de comunidades sean al menos 2 o más, la operación pedirá al usuario la introducción de un valor en coma flotante dentro del rango válido 0<=v<=1 . La operación se asegura que el valor introducido esté dentro de ese rango, en caso contrario repite la introducción de datos mostrando un mensaje con los valores permitidos.
3. Una vez establecido en el primer paso el número de comunidades que formarán parte de la simulación, la operación debe solicitar tantos nombre y poblaciones como número de comunidades haya introducido el usuario. Para ello recurriremos a una iteración definida por el número de comunidades. Se permitirá cualquier carácter para que forme parte de la cadena de caracteres dentro de la codificación UTF-8, mientras que para indicar la cantidad de población de cada comunidad, solo se permitirá valores superiores a cero. El subprograma u operación sigue permitiendo la introducción de datos y su validación. Dentro de esta interacción definida, almacenamos en un agrupamiento de objetos que resulta ser campo de abstracción, y es de tipo Comunidad, los datos que ha introducido el usuario mediante los métodos correspondientes a añadir objeto a la colección -add()- y apelando al constructor diseñado en la clase comunidad que establecerá la población y el nombre de cada una de las comunidades como instancias de la clase Comunidad particulares. Añadiremos el objeto construido a la colección como objeto anónimo ya que será la propia colección quien a través

de sus propias operaciones nos permita acceder a las diferentes instancias de la abstracción Comunidad, una vez la vinculemos por referencias con sus clases asociadas (Simulación).

ABSTRACCIÓN: Simulación
OPERACIÓN: calcularNd : void
REQUISITOS: Esta operación debe rescatar los datos que ha introducido el usuario y proceder a realizar los cálculos descritos en la Introducción de este documento. Para ello, aplicaremos la siguiente fórmula:

$N_{d+1}=N_d*(1*E*p)$

FUNCIÓN: La operación se diseña partiendo de la idea de una tabla ordenada por filas y columnas. Donde las filas representan los valores N_d (Número de infectados en un día) y las columnas representan la comunidad que presenta dicho número de infectados.
Para implementar este concepto, recurriremos a una estructura de datos estática, donde las **columnas** estén regidas por el número de comunidades que rescatamos de la colección Comunidades (indexada en esta clase Simulación por referencia), o lo que es lo mismo y más seguro de cara a controlar excepciones de exceso de índice en dichas estructuras: estarán determinadas por el **tamaño de la colección comunidades**, que almacena todas y cada una de las comunidades que ha introducido el usuario previamente; y donde las **filas** se rigen por el **número de días a simular** que ha introducido el usuario en la entrada y validación de datos.
A través de iteraciones definidas por los valores anteriormente mencionados se procederá (con iteraciones anidadas), a

1. Realizar el cálculo de Nd+1, partiendo de la premisa que el en d=0, existe 1 infectado. Realizamos el cálculo y lo almacenamos en una variable. Es de señalar que como comenzamos a calcular un segundo día, el primer índice de la estructura de datos a iterar sea 1 y no cero. Preservando el índice cero para almacenar el número de infectados inicial, cuyo valor mencionado anteriormente es 1. De este modo aplicamos la fórmula, guardamos el resultado en una variable y esta variable se guarda en filas de la matriz expuesta. Cuando el número de días a simular a terminado, la doble iteración, nos sitúa en una nueva comunidad donde se vuelven a calcular los valores de infectados para esa nueva comunidad, tantas veces, como comunidades haya establecido el usuario.
2. Se establecerá que la estructura de datos estática, indicada en la descripción de la operación sea un campo de la abstracción Simulación, en donde esta operación almacene en memoria los cálculos, permitiendo a otras clases su acceso a estos datos a través de métodos Getters, para recuperar y así poder mostrar los resultados en pantalla cuando el usuario los solicite a través de MenuResultados, ubicado como abstracción estática anidada de la clase Lanzador.

Restricciones de los datos:

Tipos de Datos
E → ***Mayor o igual que cero***
díasSimulación, Población → **Mayor que 0**
P, V → ***Mayor o igual a Cero y Menor o igual a 1***

VIABILIDAD DE RECURSOS ESTIMADOS

Se precisa un tiempo de implementación de unas 50 o 60 horas, además de equipos informáticos adecuados y una conexión a internet como fuente de documentación oficial (API) y de webgrafía complementaria.

MATRIZ DE REQUISITOS COMPONENTES

Requisitos	**Componentes**				
	Menus	**Introd.Datos**	**Comunidad**	**Simulación**	**MenuResultados**
Simular proliferación virus				X	
Interfaz de Usuario	X				X
Introdución de coeficientes		X			
Estructura de datos de tipo Comunidad			X		
Mostrar Resultados Pausados tipo tabla					X
Permitir nueva simulación (nuevos Datos)	X				X

DIAGRAMAS UML

Diagrama de Casos de uso

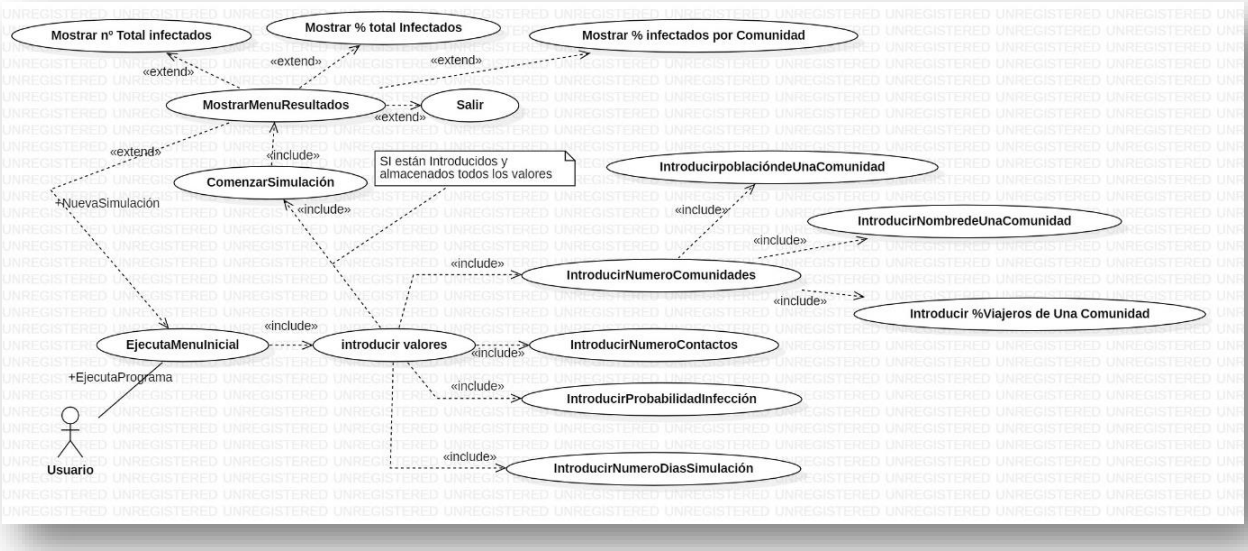


Diagrama de Clases

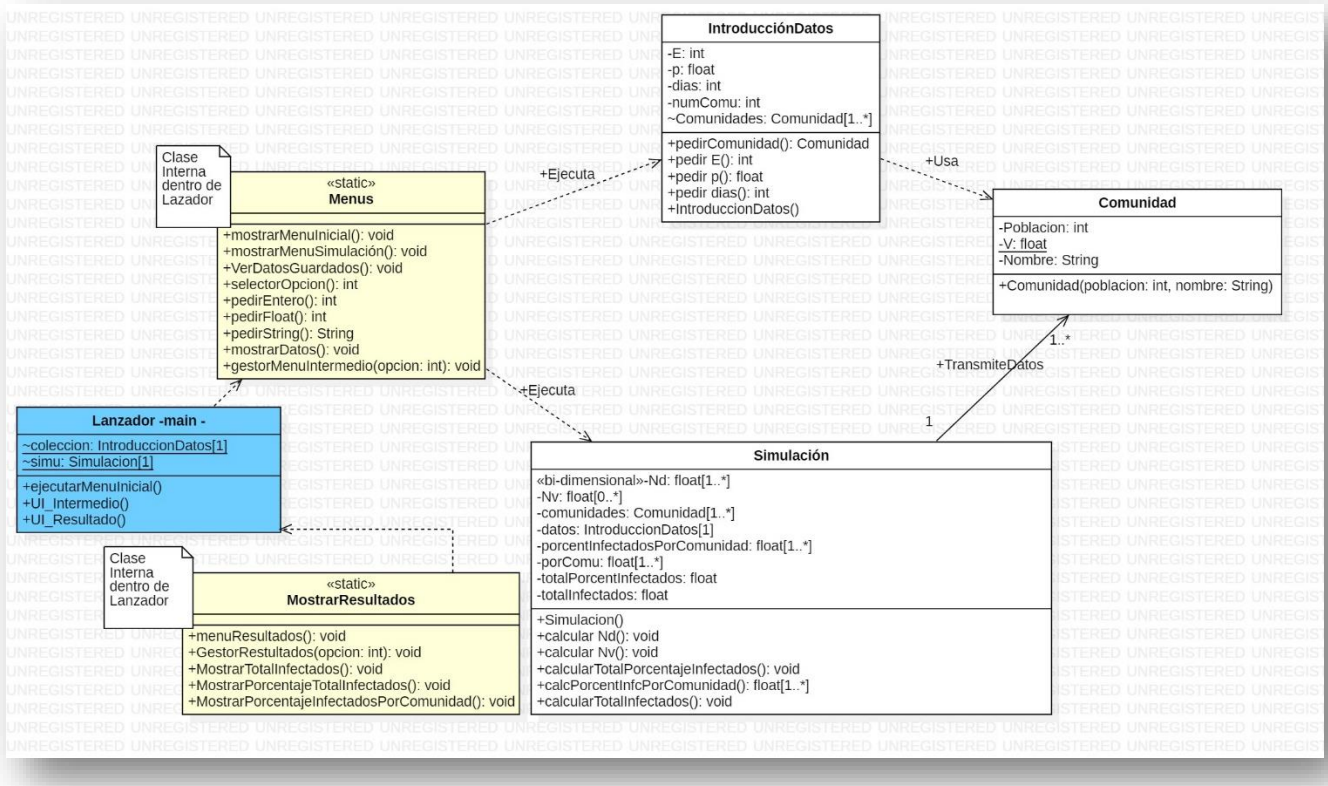
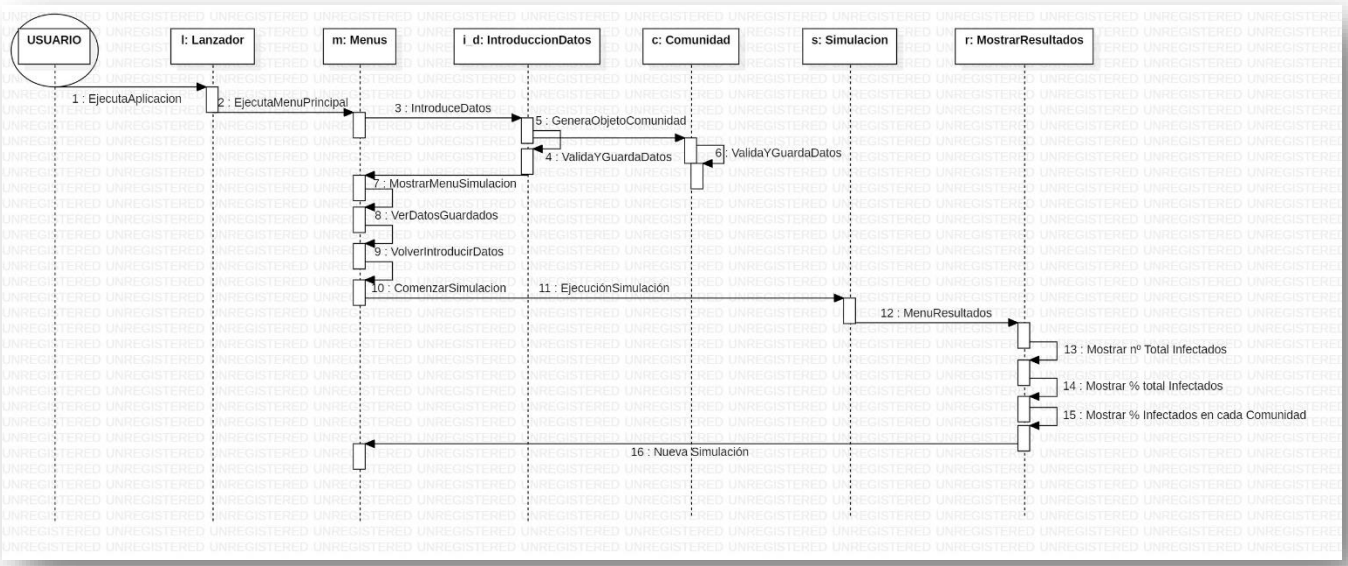


Diagrama de Secuencia



Breve explicación de la secuencia de ejecución del programa:

El usuario arranca y ejecuta la aplicación. Ésta localiza la clase principal que contiene el método que da comienzo a la ejecución donde se hace una llamada al MenuPrincipal. Este Menú permitirá ejecutar la Introducción de Datos con los coeficientes mencionados a lo largo de este Documento de Diseño. Uno de esos datos será el número de comunidades, sus poblaciones y nombres respectivos, así como el porcentaje de viajeros "estático" que se traslada a otra comunidad. Se producen la validación de los datos y el almacenamiento en memoria si éste es admitido. Posteriormente el programa, ejecutará el menú simulación, en donde permitimos al usuario 1) Ver los datos almacenados, 2) Volver a introducir nuevos datos, 3) Comenzar la simulación. Una vez elegida la opción "ComenzarSimulación", la abstracción Simulación, a través de su instancia 's' , realizará las operaciones matemáticas descritas en este documento y los cálculos pertinentes para desembocará en el menú MostrarResultados donde podremos elegir 1) Ver el nº total de infectados, 2) Ver el porcentaje total de infectados, 3) Ver el porcentaje de infectados en cada Comunidad, y 4) Volver a realizar una nueva simulación con nuevos datos, volviendo al la abstracción Menus en su instancia 'm', que repetirá el proceso hasta que en el menuInicio o MenuResultados se pulse la tecla que indique la opción de salir del programa.

Diagrama de transición de estados

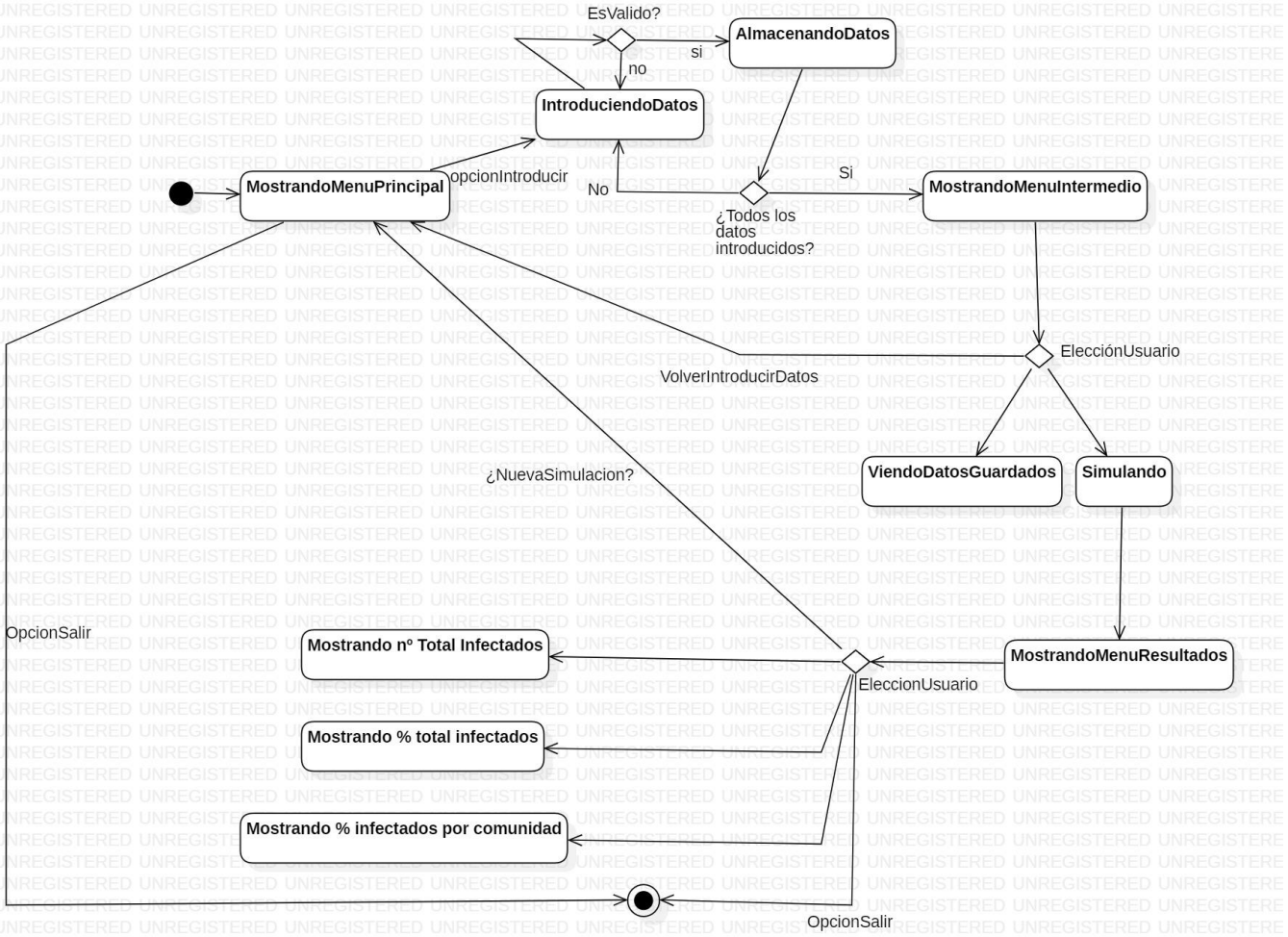
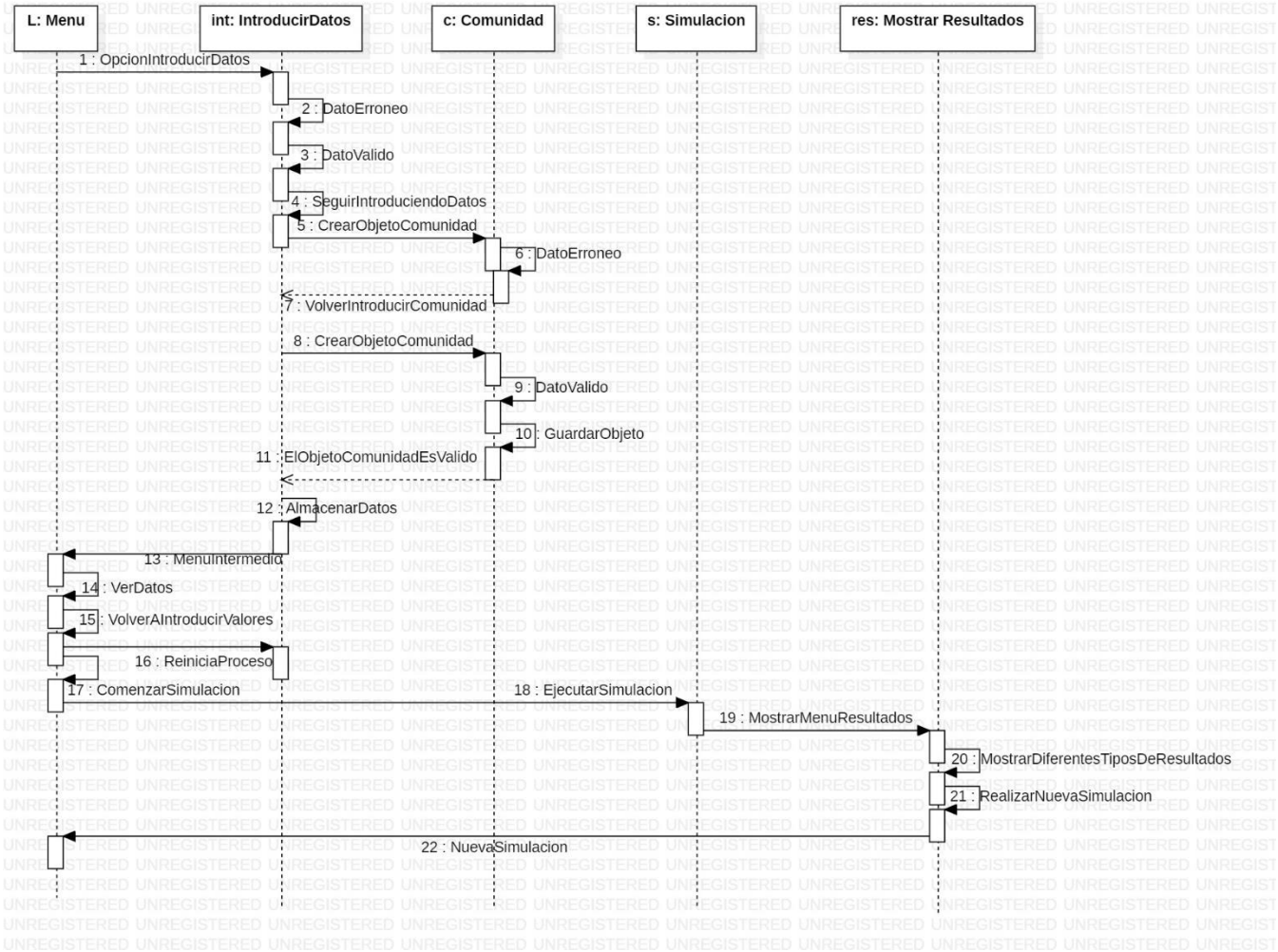


Diagrama de Secuencia

Acerca de la ejecución con introducción inválida de datos



Breve explicación de la secuencia de ejecución del programa bajo datos inválidos:

El usuario ejecuta el programa y éste ejecuta su menú principal. El usuario escoge la opción *introducir datos*, lo que da paso al módulo responsable de esta función, que validará cada dato en función de las restricciones especificadas en este documento hasta que el dato introducido sea válido, procediendo a su almacenaje en memoria. En el caso del tipo de dato *Comunidad*, existe una abstracción con funcionalidad específica que validará estos datos y responderá a la abstracción *Introducir datos* si los datos introducidos son válidos o no, procediéndose a su almacenaje o a la repetición de la introducción, respectivamente. Una vez estén almacenados los datos, significará que todos han resultado válidos conforme a las restricciones establecidas y continuará el flujo de ejecución del programa a través del menú intermedio, con sus correspondientes opciones. Una vez que el usuario escoja comenzar simulación, el módulo encargado de tal función realizará la simulación mediante los cálculos descritos en la *introducción* de este documento y procederá a dar paso al *menúResultados*, con sus funciones especificadas, existiendo la posibilidad de salir de programa o comenzar una nueva introducción de datos para llevar a cabo una nueva simulación.