

Grado en Ingeniería Informática
Computación

Trabajo de Fin de Grado

**Transpilador de un lenguaje de modelado
personalizado de sistemas de simulación
dinámicos discretos a Python**

Autor

Baldwin David Rodríguez Ponce

2022

BORRADOR. Nota: Márgenes del documento dibujados explícitamente.

--

Grado en Ingeniería Informática
Computación

Trabajo de Fin de Grado

**Transpilador de un lenguaje de modelado
personalizado de sistemas de simulación
dinámicos discretos a Python**

Autor

Baldwin David Rodríguez Ponce

Directora

María Begoña Losada Pereda

BORRADOR. Nota: Márgenes del documento dibujados explícitamente.

--

Tabla de contenidos

Índice de figuras	v
1. Introducción	1
2. Contexto	3
2.1. Motivación	3
2.2. Antecedentes	4
2.2.1. TFG-UGR	4
2.2.2. Lenguajes de simulación	4
2.2.3. Software de simulación	4
3. Gestión del proyecto	5
3.1. Alcance	5
3.1.1. Objetivos	6
3.1.2. Requisitos	7
3.1.3. EDT	7
3.2. Metodología	8
3.2.1. Trello como herramienta de gestión	9
3.3. Estimación de dedicaciones y tareas	9
3.3.1. Estimación de dedicación	10
3.3.2. Descripción de tareas a realizar	10
3.4. Análisis de riesgos y viabilidad	14
3.4.1. Análisis de riesgos	14
3.4.2. Análisis de viabilidad	14
3.5. Sistema de información y comunicaciones	14
3.5.1. Sistema de Información	14
3.5.2. Comunicaciones	15
4. Marco Teórico	17
4.1. Simulación	17

4.1.1. Definiciones	17
4.1.2. Ventajas e inconvenientes	17
4.1.3. Diseño basado en eventos	17
4.1.4. Sistemas dinámicos discretos	17
4.2. Compilación	17
4.2.1. Conceptos básicos	17
4.2.2. Herramientas	17
5. Desarrollo	19
5.1. Diseño	19
5.1.1. Lenguaje	19
5.1.2. Núcleo	19
5.1.3. CLI	19
5.2. Implementación	19
5.2.1. Lenguaje	19
5.2.2. Núcleo	19
5.2.3. CLI	19
Bibliografía	21

Índice de figuras

3.1. EDT del Proyecto 7

--

CAPÍTULO 1

Introducción

El modelado matemático o modelado analítico se aprovecha de las características del problema cuya respuesta se desea obtener para llegar a la mejor conclusión. Sin embargo, muchas veces nos encontraremos con problemas que no se pueden modelar analíticamente debido a su complejidad en tiempo o espacio. En estas situaciones, debemos hacer uso de técnicas de aproximación de resultados para dar con soluciones que, aunque no se pueden garantizar óptimas, sí que se pueden considerar lo suficientemente buenas. Una de estas técnicas vendría a ser la simulación de sistemas, la cual a través de modelado simbólico/lógico intenta reproducir el comportamiento de un determinado sistema con el fin de analizar una serie de resultados a escoger.

Nosotros hablaremos de una definición más exacta de “Simulación de Sistemas” y “Modelado” más adelante, pero por ahora podemos citar la definición encontrada en

Una simulación es la imitación del comportamiento de un proceso o sistema del mundo real en el tiempo. **Simulation involves the generation of an artificial history of the system, and the observation of that artificial history to draw inferences concerning the operating characteristics of the real system that is represented.**

Actualmente hay muchos tipos distintos de sistemas de simulación, como por ejemplo los “Modelos de Montecarlo” y “Modelos en ecuaciones diferenciales”. No obstante, el principal objeto de estudio de este proyecto será una categoría muy importante a la que se conoce como “Modelos de simulación dinámicos discretos”, específicamente aquellos que se pueden modelar a través de un diseño orientado a eventos.

Sin embargo, es necesario recalcar que existe ya una multitud de aplicaciones orientadas totalmente a la simulación de sistemas: Arena, AutoMod, Extend, entre otros. El hecho de que exista software específico para esto nos da un indicativo de lo importante que es esta área para el estudio de resultados y el modelado de procesos. De hecho, el tipo específico de

Toca traducir lo que queda. Además asegúrate de que la cita sea correcta, Jerry Banks aparece dos veces en tus referencias

Tienes que hacer una cita aquí, esto no lo estás sacando de la nada.

No olvides
citar esta lista
de aplicacio-
nes.

Has dicho
software, pero
en realidad
sí que hay
lenguajes de-
dicados a esto

Un poco seco,
cambia la
redacción

modelos del cual hablaremos es muy utilizado en sistemas de salud, de transporte, control de inventarios, líneas de montaje o ensamblaje, entre otros.

Como mencionábamos antes, ya hay lenguajes de programación enfocados a la simulación de sistemas. Lo cual nos indica la unión de este campo con el de compilación y diseño de procesadores de lenguajes formales. Ahora mismo parece ser el momento más idóneo para usar un procesador de este tipo ya que debemos seguir una serie de tokens y una gramática fija en estos lenguajes.

Se puede ver, por tanto, que la unión entre los campos de compilación y simulación de sistemas puede generar herramientas diseñadas específicamente para agilizar el proceso de creación e implementación de modelos sin preocuparse mucho por detalles irrelevantes para el programador de la simulación.

Sabiendo todo esto, procederemos hablar en este capítulo sobre la propuesta de nuestro proyecto, sus objetivos y una serie de conceptos básicos necesarios para la simulación de sistemas y la compilación en general.

CAPÍTULO 2

Contexto

2.1. Motivación

En el área de simulación de sistemas, una familia de modelos son los “modelos dinámicos discretos”, siendo estos “dinámicos” porque su comportamiento cambia en el tiempo y “discretos” porque estos cambios ocurren en instantes determinados en vez de ocurrir continuamente. Dichos sistemas se suelen modelar usando un diseño “basado en eventos”, el cual nos indica que los cambios del modelo se deben considerar como “eventos” que pueden ocurrir y dar lugar a otros eventos. Para ello, es común generar un “grafo de sucesos” que represente todos los eventos posibles que pueden ocurrir en el sistema y las relaciones que tienen entre estos.

Podríamos decir que cada modelo tendrá sus peculiaridades y diferencias específicas a la hora de generar su implementación. Sin embargo, se da el hecho de que todos los modelos de esta familia comparten elementos en común independientemente del sistema a simular: el reloj de la simulación, el temporizador de eventos, la lista de sucesos, entre otros. Por tanto, podemos abstraer el desarrollo de estos programas de tal forma que el desarrollador sólo deba encargarse de implementar todo aquello que sea único del modelo, quitándole así la responsabilidad de generar los elementos en común y agilizando el desarrollo en el proceso.

Tomando en cuenta que la principal herramienta de diseño de estos modelos serán los grafos de sucesos, nos encontraremos con el hecho de que éstos siguen una estructura representable a través de una gramática independiente de contexto. Por tanto, es posible generar una serie de analizadores léxico, sintáctico y semánticos que nos permitan procesar un lenguaje formal a otro código fuente. Por esta razón hemos considerado una buena opción el uso de herramientas de generación de compiladores como lo son Flex y Bison. Vemos que es posible crear estos analizadores de forma que este hipotético nuevo lenguaje sea traducido a código Python.

Siento que estás repitiéndote mucho, tienes que cambiar la calidad de este discurso

2.2. Antecedentes

2.2.1. Software para la construcción automática de programas de Simulación Continua

2.2.2. Lenguajes de simulación

Mencionar que hay dos categorías de lenguajes de programación para desarrollar los simuladores: específicos y generales. El nuestro será específico.

Aquí mencionaré algunos lenguajes de programación creados con el fin de desarrollar simuladores

2.2.3. Software de simulación

Aquí mencionaré algunas aplicaciones creadas para desarrollar simulaciones (como Arena)

CAPÍTULO 3

Gestión del proyecto

3.1. Alcance

El alcance de este proyecto incluye el trabajo necesario para diseñar, implementar y documentar un framework de modelado y desarrollo de simulación de sistemas dinámicos discretos basados en eventos. Dicho framework se dividirá en tres módulos principales:

- **Lenguaje:** Un lenguaje de modelado específico pensado para ser usado por usuarios que no tengan mucha experiencia en programación. Se hará uso de las herramientas de desarrollo de compiladores Flex y Bison para generar un transpilador que traduzca ficheros de este lenguaje a código Python. A través de él se plantea:
 - Permitir la rápida implementación de este tipo de modelos a través de grafos de sucesos.
 - Permitir que el programador del lenguaje se encargue sólo de realizar las implementaciones pertinentes al sistema de simulación que desee desarrollar:
 - Especificación de las variables globales, variables de entrada, contadores estadísticos y medidas de rendimiento propias del modelo.
 - Inclusión de eventos adicionales y sus acciones correspondientes.
 - Creación y eliminación de eventos en función de tiempo y condiciones lógicas.
 - Inclusión de código adicional escrito directamente en Python en caso de ser necesario.
- **Núcleo:** Una serie de módulos que implementarán un microframework de simulación de este tipo de sistemas en específico para Python, pensado para ser usado por programadores y para acotar la traducción del nuevo lenguaje. A través de él se plantea:

Cambiar redacción

Aquí te falta la posibilidad de poner las cosas variables también

- Permitir que la traducción del lenguaje incluya dentro del fichero generado las estructuras de datos, funciones y procedimientos que tienen en común todos los sistemas dinámicos discretos:

- Generadores de datos aleatorios para distintos tipos de distribuciones.
- Reloj y temporizador de simulación para ejecutar los eventos.
- Estructura de datos para almacenar los sucesos según deben ocurrir en el tiempo.
- Las respectivas implementaciones mínimas de los dos eventos que siempre formarán parte de todos los modelos: “Inicio” y “Fin”.
- Generador de informes final que se ejecutará al finalizar la simulación y mostrará los resultados que se deseaban estudiar con ésta.

- **CLI**: Una interfaz de comandos por terminal que se usará para gestionar, configurar y ejecutar los proyectos desarrollados con este framework.

3.1.1. Objetivos

Objetivo general

Generar un lenguaje de modelado de simulación de sistemas dinámicos discretos basados en eventos junto con un transpilador que lo traduzca a Python, un microframework para acotar la traducción y un CLI para gestionar proyectos desarrollados con este producto.

Objetivos específicos

1. Diseñar un nuevo lenguaje de modelado y simulación de sistemas dinámicos discretos basados en eventos usando las herramientas de desarrollo de procesadores de lenguaje Flex y Bison.
2. Diseñar, implementar y verificar una serie de módulos y funcionalidades realizadas en Python con el fin de generar un microframework para simular estos mismos sistemas.
3. Implementar y verificar un transpilador que traduzca el lenguaje del producto a Python usando las características del objetivo anterior.
4. Diseñar, implementar y verificar una serie de operaciones accesibles desde una CLI de cara a ser usadas para la gestión, configuración y ejecución parametrizada de simulaciones realizadas con el producto.
5. Implementar distintas técnicas de análisis de salidas, experimentación y optimización de modelos dentro del núcleo del framework.

6. Desarrollar un manual de usuario que contendrá toda la documentación necesaria para hacer uso del framework.

3.1.2. Requisitos

El requisito base principal del proyecto consiste en cumplir con un tiempo de dedicación total máximo de 300 horas.

3.1.3. EDT

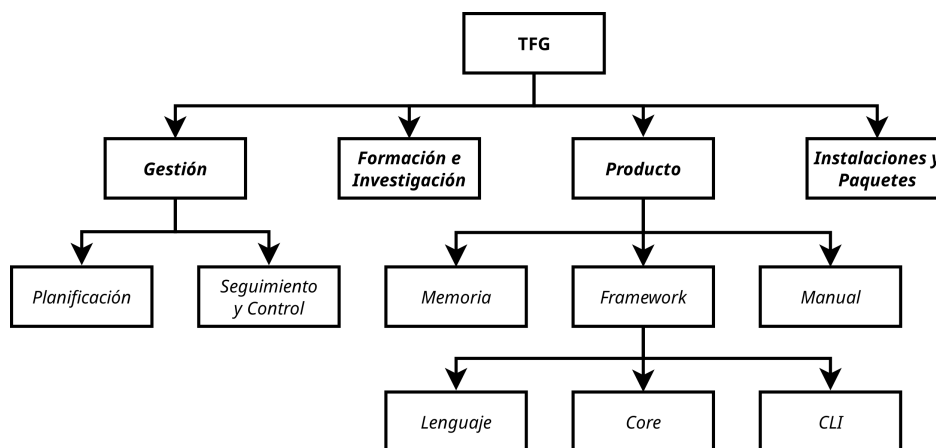


Figura 3.1: Esquema de Descomposición de Trabajo del Proyecto

Rama de Gestión

El paquete de trabajo **Gestión (Ge)** contendrá:

- **Planificación (Ge.P):** Este paquete de trabajo agrupará todas las tareas relacionadas a la realización de la planificación y preparación inicial del proyecto.
- **Seguimiento y Control (Ge.S):** Este paquete de trabajo agrupará todas las tareas necesarias para asegurar que el seguimiento del proyecto está realizando como se plantea en la planificación y, en caso de no ser así, controlar las consecuencias de las desviaciones emergentes.

Rama de Formación e Investigación

El paquete de trabajo **Formación e Investigación (FeI)** contendrá todas las tareas relacionadas con el aprendizaje de herramientas, búsqueda de referencias y recolección de información necesaria para el desarrollo del proyecto.

Rama de Producto (Pr)

El paquete de trabajo **Producto (Pr)** contendrá todas las tareas relacionadas al propio diseño, implementación y verificación de los entregables principales del proyecto. Podemos desglosarlo en tres paquetes más pequeños:

- **Memoria (Pr.Me):** Este paquete de trabajo agrupará todas las tareas necesarias para la realización de la memoria final del proyecto.
- **Framework (Pr.Fr):** Este paquete de trabajo agrupará todas las tareas relacionadas al desarrollo del framework planteado en el alcance del proyecto. Podemos separarlo en sus tres partes principales:
 - **Lenguaje (Pr.Fr.Le):** Este paquete contendrá todas las tareas relacionadas al diseño y desarrollo del nuevo lenguaje de modelado.
 - **Core (Pr.Fr.Co):** Este paquete contendrá todas las tareas relacionadas al diseño y desarrollo del propio núcleo del framework.
 - **CLI (Pr.Fr.Cli):** Este paquete contendrá todas las tareas relacionadas al diseño y desarrollo de la interfaz de comandos proporcionada para facilitar la gestión de proyectos realizados con el a generar.
- **Manual (Pr.Ma):** Este paquete de trabajo agrupará todas las tareas necesarias para la realización del manual de usuario que se generará para facilitar el uso del producto a desarrollar.

Rama de Instalación y Paquetes

El paquete de trabajo **Instalación y Paquetes (IyP)** agrupará todas las tareas relacionadas a la preparación del entorno de desarrollo y la instalación de paquetes y dependencias para la realización del producto principal del proyecto.

3.2. Metodología

Explicar Kanban

Por último, la gestión del proyecto se realizará siguiendo la metodología de desarrollo ágil Kanban para permitir construir de manera iterativa las distintas funcionalidades del proyecto. Hacer uso del método Kanban para el desarrollo y gestión del proyecto.

Desglosar más lo que es Kanban

3.2.1. Trello como herramienta de gestión

Significado de las listas

- **Backlog:**
- **To Do:**
- **Doing:**
- **Testing:**
- **Done:**
- **Approved:**

Significado de etiquetas

- **Bug:**
- **Bloqueado:**
- **Pendiente:**
- **No aceptada:**

Formato de las cartas

3.3. Estimación de dedicaciones y tareas

No se puede predecir porque usamos Kanban. Sólo conocemos las fechas finales de entrega.

3.3.1. Estimación de dedicación

Paquete	Nombre	Estimación (Horas)
Ge.P	Gestión - Planificación	23
Ge.S	Gestión - Seguimiento	20
FeI	Formación e Investigación	46
Pr.Me	Proyecto - Memoria	46
Pr.Fr.Le	Proyecto - Framework - Lenguaje	44
Pr.Fr.Co	Proyecto - Framework - Core	48
Pr.Fr.Cli	Proyecto - Framework - CLI	21
Pr.Ma	Proyecto - Manual	11
IyP	Instalaciones y Paquetes	11
Contención	Horas de contención	30
Total	Total	300

Tabla 3.1: Estimación de tiempos de dedicación por paquetes de trabajo

Explicar que contención no es un paquete de trabajo, pero que es una parte importante de la estimación

3.3.2. Descripción de tareas a realizar

Aquí irán todas las tareas realizadas al final del proyecto. Esto no es fijo, surgirán más tareas a lo largo del proyecto (lo cual es aceptable porque la metodología es ágil)

Gestión

Planificación

1. Definición de la metodología de gestión.
2. Generación de la planificación inicial.
3. Automatización de herramientas de gestión con Trello.
4. Cambios y actualizaciones de la planificación.

Seguimiento y Control

1. Recogida de información sobre el desarrollo del proyecto
2. Contraste de la información de seguimiento con los planes, identificación de desviaciones significativas y actuación ante riesgos emergentes.
3. Preparación de documentos de cara a la próxima reunión.
4. Reuniones de fin de iteración

Formación e Investigación

1. Investigación y formación sobre la metodología Kanban.
2. Profundización sobre conceptos de simulación de sistemas.
3. Profundización sobre conceptos de compilación.
4. Investigación y formación sobre herramientas de desarrollo de CLIs.
5. Investigación y formación sobre generación de paquetes instalables para Python.
6. Investigación y formación sobre funciones de alto nivel y paquetes relacionados de Python.
7. Investigación sobre lenguajes de modelado y simulación.
8. Investigación sobre antecedentes del proyecto.
9. Exploración de alternativas al módulo de lenguaje.

Producto**Memoria**

1. Diseño y preparación de la estructura
2. Redacción del resumen
3. Redacción de la introducción
4. Redactar contexto
5. Redacción de la gestión del proyecto
6. Redacción del marco teórico
7. Redacción del desarrollo
8. Redacción de conclusiones
9. Preparación e inclusión de referencias bibliográficas
10. Redacción de anexos
11. Validación y corrección de los índices
12. Maquetación y diseño de la memoria

Framework**■ Lenguaje:**

1. Diseño del léxico del lenguaje.

2. Diseño de la sintaxis del lenguaje.
3. Diseño de la semántica del lenguaje.
4. Diseño de módulos de cara a la implementación del lenguaje.
5. Implementación del analizador léxico.
6. Implementación del analizador sintáctico.
7. Implementación de la tabla de símbolos.
8. Implementación del análisis semántico.
9. Documentación de los ficheros del lenguaje.
10. Pruebas y verificación del lenguaje.

■ **Núcleo:**

1. Diseño de los módulos y paquetes del núcleo.
2. Implementación de la estructura de datos para almacenar eventos.
3. Implementación del reloj y el proceso de temporización de sucesos.
4. Implementaciones de los procesos de inicialización y finalización.
5. Implementación del generador de informes final.
6. Implementación de distintos generadores de datos aleatorios.
7. Implementación de la inicialización de las variables globales, variables de entrada, contadores estadísticos y medidas de rendimiento para cualquier modelo.
8. Implementación de la inclusión de nuevos tipos de eventos y sus funciones de ejecución correspondientes.
9. Implementación de distintas estrategias de generación y eliminación de eventos dentro de la lista de sucesos.
10. Implementación de la especificación de una función de parada definida por el usuario.
11. Implementación de la funcionalidad de configuración de la simulación.
12. Documentación de los ficheros del núcleo.
13. Pruebas y verificación del núcleo.

■ **CLI:**

1. Diseño de las funcionalidades de la CLI.
2. Implementación del generador de proyectos.

3. Implementación del gestor de configuraciones del proyecto.
4. Implementación del lanzador de modelos de simulación.
5. Documentación de los ficheros de la CLI.
6. Pruebas y verificación de la CLI.

Manual

1. Diseño y preparación de la estructura
2. Redacción de la introducción al producto
3. Especificación de dependencias y requisitos
4. Redacción del proceso de instalación
5. Redacción de la explicación del módulo del lenguaje
6. Redacción de la explicación del núcleo del framework
7. Redacción de la explicación de la CLI
8. Generación de ejemplos de uso
9. Preparación e inclusión de referencias bibliográficas
10. Redacción de anexos
11. Validación y corrección de los índices
12. Maquetación y diseño del manual

Instalación y Paquetes

1. Preparación de paquetes para LaTeX.
2. Preparación del entorno de trabajo.
3. Instalación de paquetes para el desarrollo del lenguaje.
4. Instalación de paquetes para el desarrollo del núcleo.
5. Instalación de paquetes para el desarrollo de la CLI.

3.4. Análisis de riesgos y viabilidad

3.4.1. Análisis de riesgos

Métricas

Probabilidad	Valor	Impacto	Valor	Tipo de Riesgo	Probabilidad x Impacto
Nada probable	0.10	Muy bajo	0.05	Muy bajo	De 0 a 0.04
Poco probable	0.30	Bajo	0.10	Bajo	De 0.05 a 0.09
Probable	0.50	Moderado	0.20	Moderado	De 0.10 a 0.29
Muy probable	0.70	Alto	0.40	Alto	De 0.30 a 0.49
Casi probable	0.90	Muy alto	0.80	Muy alto	Mayor a 0.50

Tabla 3.2: Métricas para el análisis de riesgos

Tabla de riesgos

Nº	Tipo	Descripción	Probabilidad	Impacto	Nivel de Riesgo	Respuesta
1	I	Mala gestión del tiempo (falta de tiempo + pérdida de tiempo)	Probable	Alto	Moderado	Replanificar + hacer uso de la agilidad de la metodología
2	E	Problemas de contacto con [directora] (por problemas de conexión, enfermedad, etc. Tanto en el caso del autor como en el caso de la directora)	Poco probable	Bajo	Muy bajo	Replanificar reuniones?
3	I	No entender la metodología de gestión y desarrollo (Kanban)	Probable	Alto	Moderado	Profundizar en la formación de la metodología y generar cambios en el seguimiento y control
4	E	Pérdida de información (se solventa con gestión de versiones + github)	Nada probable	Muy alto	Bajo	Hacer uso de control de versiones para guardar todos los datos. En este caso GitHub.
5	I	Problemas con la integración de herramientas de desarrollo	Muy probable	Muy alto	Muy alto	Buscar alternativas en caso de no poder integrar las herramientas o profundizar en la formación de esta integración para evitar que ocurra
6	I	Análisis incorrecto de los usuarios finales del producto (Piensa en el lenguaje, por ejemplo).	Probable	Alto	Moderado	Reanalizar los perfiles de usuarios objetivos y replantear los productos
7	I	Encontrado un proyecto idéntico (riesgo positivo (o se aceptan o se explotan))	Poco probable	Moderado	Bajo	Buscar una forma de hacer uso de éste, ya sea para integrarlo en el proyecto o para usarlo como formación.

Tabla 3.3: Riesgos del proyecto

Tenemos que diferenciar entre riesgos externos e internos

Hay riesgos negativos (que afectan malamente al proyecto) y riesgos positivos (que afectan positivamente al proyecto)

3.4.2. Análisis de viabilidad

Considero que el proyecto es viable

3.5. Sistema de información y comunicaciones

3.5.1. Sistema de Información

Estructura

TFG: Directorio principal del proyecto. Se divide en:

- **TFG.Proyecto:** Aquí residen los entregables del proyecto.

- **Memoria:** Los ficheros relacionados a la memoria

- **Framework:** Los ficheros relacionados al framework

Aquí debemos explicar la estructura

- **Lenguaje:** Los ficheros relacionados al lenguaje del framework
- **Core:** Los ficheros relacionados al núcleo del framework
- **CLI:** Los ficheros relacionados al CLI del framework
- **Manual:** Los ficheros relacionados al manual
- **TFG.Gestión:** Aquí residen los ficheros relacionados a la gestión del proyecto
 - **Planificación:** Los ficheros relacionados a la planificación
 - **Seguimiento:** Los ficheros relacionados al seguimiento y control
 - **Actas:** Las actas de cada reunión realizada

3.5.2. Comunicaciones

- [Reuniones a través de Webex]
- [Comunicaciones para responder dudas a través de email]

--

CAPÍTULO 4

Marco Teórico

4.1. Simulación

4.1.1. Definiciones

Modelo

Sistema

4.1.2. Ventajas e inconvenientes

4.1.3. Diseño basado en eventos

4.1.4. Sistemas dinámicos discretos

4.2. Compilación

4.2.1. Conceptos básicos

Analizador sintáctico

Analizador semántico

4.2.2. Herramientas

--

CAPÍTULO 5

Desarrollo

5.1. Diseño

5.1.1. Lenguaje

Léxico

Tabla de tokens Autómata

Sintaxis + Semántica

Gramática Definición de atributos Descripción de abstracciones funcionales ETDS Tabla de símbolos

5.1.2. Núcleo

Diagrama de clases?

5.1.3. CLI

5.2. Implementación

5.2.1. Lenguaje

5.2.2. Núcleo

5.2.3. CLI

--

Bibliografía

- Buss, A. (1996). Modeling with event graphs. *Winter Simulation Conference*, 153-160. <https://doi.org/10.1109/WSC.1996.873273>
- Maria, A. (1997). Introduction to modeling and simulation. *Proceedings of the 29th conference on Winter simulation - WSC '97*. <https://doi.org/10.1145/268437.268440>
- Perros, H. G. (2009). *Computer simulation techniques: the definitive introduction!*
- Banks, J. (2010). *Discrete-event system simulation*. Prentice Hall.
- Wainer, G. (2011). *Discrete-event modeling and simulation: theory and applications*. CRC Press.
- MATLAB. (2017a). Understanding Discrete Event Simulation, Part 1: What Is Discrete Event Simulation.
- MATLAB. (2017b). Understanding Discrete Event Simulation, Part 2: Why Use Discrete Event Simulation.
- MATLAB. (2017c). Understanding Discrete Event Simulation, Part 3: Leveraging Stochastic Processes.
- MATLAB. (2017d). Understanding Discrete Event Simulation, Part 4: Operations Research.
- MATLAB. (2017e). Understanding Discrete Event Simulation, Part 5: Communication Modeling.

--
