



UNIVERSIDAD DEL BÍO-BÍO

Diseño e Implementación de un Lenguaje de  
Programación de Tipo Declarativo Inspirado en Prolog  
para Control de Agentes en Videojuegos

Christopher Cromer y Martín Araneda Acuña

29 de abril de 2022

# Índice

<b>Composición del informe</b>	<b>2</b>
<b>1. Introducción</b>	<b>3</b>
1.1. Objetivos	4
1.1.1. Objetivo General	4
1.1.2. Objetivos Específicos	4
1.2. Metodología de trabajo	4
<b>2. Marco Teórico</b>	<b>6</b>
2.1. Desarrollo de Videojuegos	6
2.1.1. Motor de videojuego	6
2.1.2. Game I.A.	8
2.1.2.1. Lenguaje de Programación Compilado	10
<b>3. Descripción del ambiente del software</b>	<b>11</b>
3.1. Lenguaje de programación	11
3.2. Videojuego	11
3.3. Entorno	11
<b>Referencias</b>	<b>12</b>
Prolog	13
LLVM	13
Godot	13
Videojuegos	13

## Índice de figuras

1.	Godot Engine . . . . .	6
2.	Voxel Data Generation . . . . .	7
3.	Mesh Generation . . . . .	8
4.	Programación Lógica vs. Funcional . . . . .	9
5.	LLVM . . . . .	10

## Composición del informe

El presente informe se encuentra dividido en tres capítulos. A continuación se describe brevemente el contenido de cada uno de ellos.

1. **Introducción:** Se presentan los conceptos básicos más importantes para este proyecto, como son el Desarrollo de Videojuegos y la Inteligencia Artificial. También se explican los motivos principales y específicos para el desarrollo de la investigación. Por último, se detalla la metodología de trabajo a aplicar.
2. **Marco Teórico:** Explicación de las herramientas a utilizar, análisis de rendimiento de software similares y descripción de conceptos abarcados en el proyecto.
3. **Descripción del Ambiente del software:** Se explica el software a utilizar, incluyendo la versión y el entorno de desarrollo.

Además, al final del informe se adjuntan las referencias con los artículos utilizados en el proceso de investigación.

# 1. Introducción

La Inteligencia Artificial (I.A) es una disciplina científica relativamente nueva que nació de los avances en las ciencias de la computación y/o la informática. De forma intuitiva, la I.A puede entenderse como la capacidad que tiene una máquina para tomar decisiones de forma autónoma a partir de los estímulos que recibe del exterior. Una de las primeras veces que se habló de la I.A como disciplina fue en el año 1956 cuando John McCarthy, importante matemático e informático, la definió como *"La ciencia e ingeniería de hacer máquinas inteligentes, con mayor énfasis en programas computacionales inteligentes"* [1].

Actualmente, la I.A puede considerarse también una tecnología moderna que ha ido evolucionando y se ha ido incorporando a nuestra vida cotidiana. Un ejemplo bastante importante es el reconocimiento facial introducido en teléfonos inteligentes, lo que permite agregar ciertos efectos a las imágenes, ajustar automáticamente el enfoque o balancear la exposición en condiciones de poca luz.

En el campo del desarrollo de videojuegos, la I.A. ha tenido mucha presencia. Por ejemplo, se ha empleado con éxito para programar los personajes o las entidades de los mismos, haciéndolos más dinámicos y divertidos. Permitiendo que el juego aprenda del usuario y creando adversarios artificiales del nivel de un humano.

Dado lo anterior, tendrá un gran beneficio implementar una I.A. empleando un lenguaje declarativo ya que no indicamos el cómo se debe computar, solo indicamos el qué se quiere computar. Esto último es importante ya que conseguimos que la tarea de programación se aproxime a la lógica humana y el programador pueda implementar el comportamiento de los agentes de una forma más intuitiva y empleando reglas simples. En este sentido, uno de los lenguajes declarativos más importantes ha sido Prolog demostrando sus buenas propiedades en este ámbito. Por lo tanto, vamos a crear un lenguaje declarativo inspirado en Prolog que se pueda utilizar para programar el comportamiento de los personajes de un videojuego que tienen, al menos, que realizar las siguientes acciones: i) superar obstáculos, con la opción de saltar con velocidad; ii) recoger items que se agregan a la puntuación final; iii) evitar enemigos.

## **1.1. Objetivos**

### **1.1.1. Objetivo General**

Desarrollar un lenguaje de programación de tipo declarativo inspirado en Prolog que permita modelar el comportamiento de los agentes de un videojuego empleando reglas lógicas.

### **1.1.2. Objetivos Específicos**

1. Revisar bibliografía sobre Prolog, el motor Godot y programación de videojuegos.
2. Analizar la información recopilada de la bibliografía investigada.
3. Crear el lenguaje de programación de tipo declarativo inspirado en Prolog.
4. Implementar un videojuego usando una inteligencia artificial basada en el lenguaje anteriormente creado.
5. Evaluar el desempeño del lenguaje inventado, verificando el cumplimiento exitoso de superación de obstáculos por parte del agente.

## **1.2. Metodología de trabajo**

La metodología utilizada es la de investigación experimental, la cual consiste en ir implementando ciertas acciones en la que el agente debe reaccionar de acuerdo a los estímulos que están presentes en un ambiente preestablecido, que en este caso es un nivel presente en el videojuego. Con esto, se harán pruebas de causa y efecto, recopilando información en su efectivo comportamiento racional frente a obstáculos para posterior comparación con un jugador real.

Por tanto, el plan de trabajo consistirá en las siguientes fases:

- **Fase 1:** Revisión y descarte de bibliografía relacionada al desarrollo de videojuegos con implementación de inteligencia artificial inspirado en Prolog y motor Godot.
- **Fase 2:** Estudio de la información recopilada para posible implementación en el software.

- **Fase 3:** Creación del lenguaje de programación lógico de tipo declarativo inspirado en Prolog.
- **Fase 4:** Implementación de un videojuego en el motor Godot de estilo plataforma formado por agentes inteligentes programados usando el lenguaje recientemente creado.
- **Fase 5:** Evaluación del correcto desempeño del agente en la superación de obstáculos, con el cumplimiento del nivel versus un jugador real y la comparación de la puntuación y tiempo total en completar el juego entre ambos jugadores.

## 2. Marco Teórico

Este capítulo tiene como propósito describir los conceptos más importantes en el desarrollo del lenguaje de programación para control de la Inteligencia Artificial, así como también su implementación en un juego estilo plataformas.

### 2.1. Desarrollo de Videojuegos

#### 2.1.1. Motor de videojuego

El motor de videojuego tiene una alta importancia en el desarrollo de videojuegos, ya que es responsable de proveer facilidad en utilizar gráficos, sonidos, física, redes y varios otros aspectos que ayudan en el desarrollo de un videojuego. El motor que se utilizará se llama Godot.



Figura 1: Godot Engine

El motor Godot tiene gran valor en el desarrollo del proyecto, principalmente por su capacidad para implementar arte de píxel, en comparación con motores similares como Unity o Unreal porque en Godot las distancias entre los elementos es medido en píxeles mientras que los otros motores utilizan metros, lo que facilita en gran medida la eficacia al trabajar en juegos 2D, que usualmente para este género de videojuegos se usan píxeles y no metros.

Otro factor importante es que Godot posee un elemento llamado GDNative. GDNative es una API que permite usar lenguajes de programación como C, C++ o Rust que son lenguajes que compilan el código a lenguaje máquina.

Tiene especial importancia para el proyecto ya que permite implementar otro lenguaje de programación que hará interacción con GDNative. Por otro lado también permite que la Inte-

Inteligencia Artificial tome decisiones con mayor rapidez dado que el código objeto de máquina se ejecuta más velozmente que un lenguaje interpretado o que se compila a bytecode como se puede notar en los siguientes gráficos que compara GDScript(lenguaje interpretado similar a Python), C# y C++.

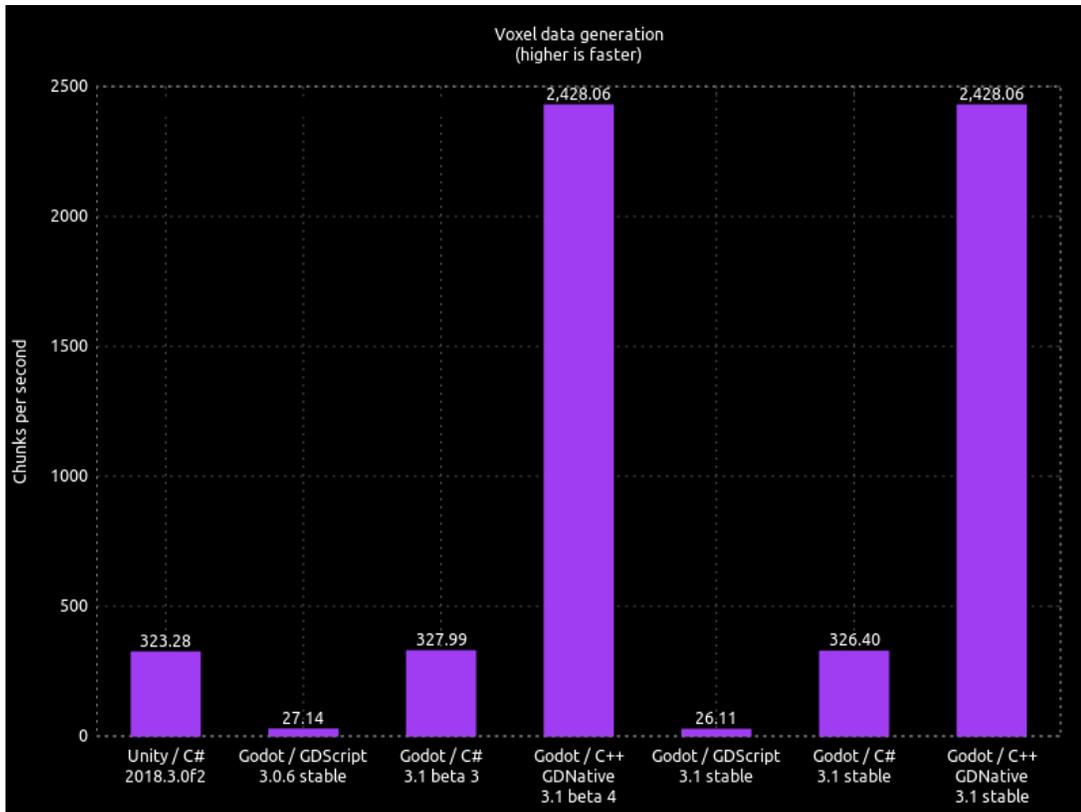


Figura 2: Voxel Data Generation [2]

En el caso de generación de datos voxel, hay un aumento de 8846.43 % en su rendimiento comparando GDNative con GDScript y un aumento de 651.07 % comparando GDNative con C#.

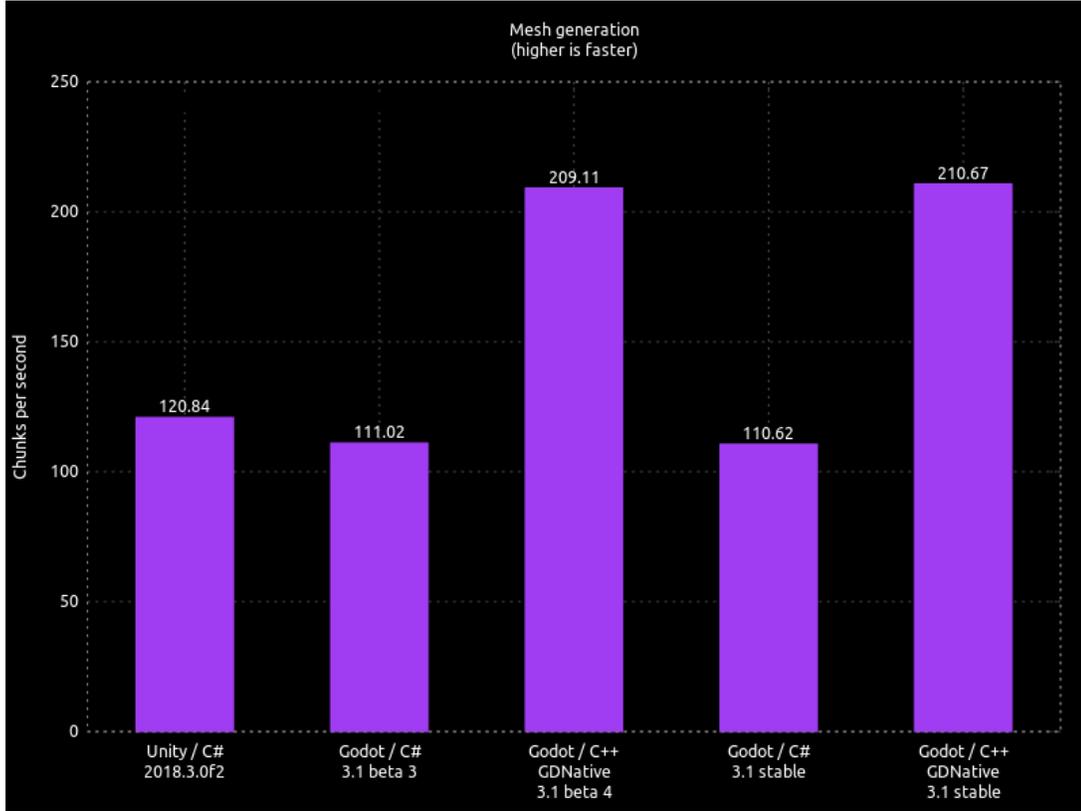


Figura 3: Mesh Generation  
[2]

En la prueba de generación de mallas el GDNative tiene un aumento de 651.07% en su rendimiento comparando con C#.

### 2.1.2. Game I.A.

Para la toma de decisiones de la Inteligencia Artificial, se va a utilizar un lenguaje propio inspirado en Prolog. El Prolog es un lenguaje declarativo y lógico.

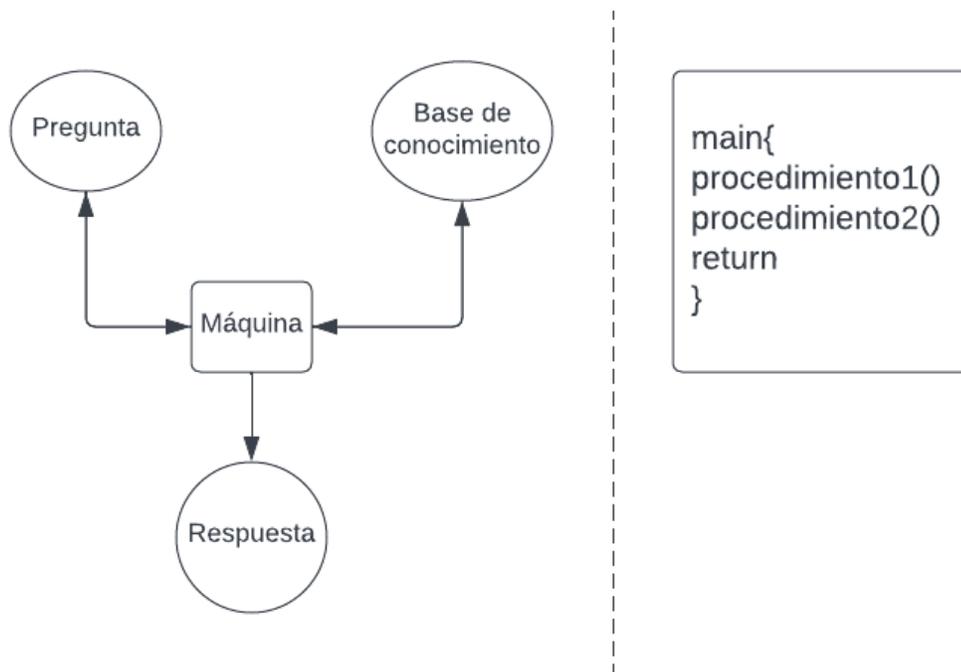


Figura 4: Programación Lógica vs. Funcional

Cuando se trata de programación funcional, la idea principal es ver cómo se puede resolver un problema. Eso se hace de forma de paso por paso para resolver el problema. En programación lógico se usa un base de conocimiento para hacer preguntas y recibir respuestas que se utilizaran para resolver el problema.

El lenguaje tipo Prolog debe tener 3 elementos importantes para funcionar.

1. **Hechos:** Son datos verdaderos. Por ejemplo "el español es una idioma".
2. **Reglas:** Son cláusulas condicionales que conectan los hechos. Por ejemplo "si vives en Chile hablas el español".
3. **Preguntas:** Son necesarias para tener una respuesta de la base de conocimiento. Por ejemplo "¿El español es una idioma?".

### 2.1.2.1 Lenguaje de Programación Compilado

El proyecto LLVM es un conjunto de tecnologías de compilador y toolchain, lo cual permite crear un lenguaje propio de programación.



Figura 5: LLVM

El LLVM consiste de varios sub-proyectos, pero el que será utilizado principalmente es LLVM Core. Este sub-proyecto contiene un optimizador y generador de código, siendo este último llamado LLVM Intermediate Representation (LLVM IR). La funcionalidad es similar a una Virtual Machine de bytecode que es portátil y se puede correr en cualquier sistema que posee el LLVM.

Otro aspecto importante de LLVM es que se puede utilizar el LLVM IR que fue generado para después compilarlo a lenguaje máquina para la arquitectura computacional que se desee. Luego, el código objeto generado se puede utilizar con un linker para crear librerías y binarios, lo que tendrá importancia al querer integrar el código que compila nuestra compilador en el motor Godot.

### **3. Descripción del ambiente del software**

Para la realización del proyecto se necesita los software siguientes.

#### **3.1. Lenguaje de programación**

Para compilar el lenguaje de programación se necesita:

- GCC 5 o clang 5
- LLVM 14

#### **3.2. Videojuego**

Para compilar el videojuego se necesita:

- GCC 5 o clang 5
- Godot 3.5

#### **3.3. Entorno**

El entorno usado para el desarrollo es:

- Linux Mint 21 o Ubuntu 22.04
- Visual Studio Code 1.66.2

## Referencias

- [1] John McCarthy. «What is artificial intelligence?» En: (2007).
- [2] Royal Donut Games. *CPU Voxel Benchmarks of Most Popular Languages in Godot*. Abr. de 2022. URL: <http://www.royaldonut.games/2019/03/29/cpu-voxel-benchmarks-of-most-popular-languages-in-godot/>.
- [3] Grzegorz Jaśkiewicz. «Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots». En: *IEEE Transactions on Computational Intelligence and AI in Games* 8.1 (2016), págs. 82-88. DOI: [10.1109/TCIAIG.2014.2331972](https://doi.org/10.1109/TCIAIG.2014.2331972).
- [4] Robert Nystrom. *Game Programming Patterns*. Genever Benning, 2014. ISBN: 0-99058-290-6.
- [5] Ernest Adams y Joris Dormans. *Game Mechanics: Advanced Game Design*. New Riders, 2012. ISBN: 0-32182-027-4.
- [6] Mayur Pandey y Suyog Sarada. *LLVM Cookbook*. Packt, 2015. ISBN: 978-1-78528-598-1.
- [7] Chris Bradfield. *Godot Engine Game Development Projects*. Packt, 2018. ISBN: 978-1-78883-150-5.
- [8] Ariel Manzur y George Marques. *Sams Teach Yourself, Godot Engine Game Development in 24 Hours*. Pearson, 2018. ISBN: 0-13483-509-3.
- [9] David Vallejo y Cleto Martín. *Desarrollo de Videojuegos: Un Enfoque Práctico*. CreateSpace, 2015. ISBN: 978-1-51730-955-8.
- [10] Max Bramer. *Logic Programming with Prolog*. Springer, 2013. ISBN: 978-1-44715-486-0. DOI: [10.1007/978-1-4471-5487-7](https://doi.org/10.1007/978-1-4471-5487-7).
- [11] tutorialspoint. *Prolog Tutorial*. Abr. de 2022. URL: <https://www.tutorialspoint.com/prolog/>.

## Prolog

- [3] Grzegorz Jaśkiewicz. «Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots». En: *IEEE Transactions on Computational Intelligence and AI in Games* 8.1 (2016), págs. 82-88. DOI: [10.1109/TCIAIG.2014.2331972](https://doi.org/10.1109/TCIAIG.2014.2331972).
- [10] Max Bramer. *Logic Programming with Prolog*. Springer, 2013. ISBN: 978-1-44715-486-0. DOI: [10.1007/978-1-4471-5487-7](https://doi.org/10.1007/978-1-4471-5487-7).
- [11] tutorialspoint. *Prolog Tutorial*. Abr. de 2022. URL: <https://www.tutorialspoint.com/prolog/>.

## LLVM

- [6] Mayur Pandey y Suyog Sarada. *LLVM Cookbook*. Packt, 2015. ISBN: 978-1-78528-598-1.

## Godot

- [2] Royal Donut Games. *CPU Voxel Benchmarks of Most Popular Languages in Godot*. Abr. de 2022. URL: <http://www.royaldonut.games/2019/03/29/cpu-voxel-benchmarks-of-most-popular-languages-in-godot/>.
- [7] Chris Bradfield. *Godot Engine Game Development Projects*. Packt, 2018. ISBN: 978-1-78883-150-5.
- [8] Ariel Manzur y George Marques. *Sams Teach Yourself, Godot Engine Game Development in 24 Hours*. Pearson, 2018. ISBN: 0-13483-509-3.

## Videojuegos

- [2] Royal Donut Games. *CPU Voxel Benchmarks of Most Popular Languages in Godot*. Abr. de 2022. URL: <http://www.royaldonut.games/2019/03/29/cpu-voxel-benchmarks-of-most-popular-languages-in-godot/>.

- [3] Grzegorz Jaśkiewicz. «Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots». En: *IEEE Transactions on Computational Intelligence and AI in Games* 8.1 (2016), págs. 82-88. DOI: [10.1109/TCIAIG.2014.2331972](https://doi.org/10.1109/TCIAIG.2014.2331972).
- [4] Robert Nystrom. *Game Programming Patterns*. Genever Benning, 2014. ISBN: 0-99058-290-6.
- [5] Ernest Adams y Joris Dormans. *Game Mechanics: Advanced Game Design*. New Riders, 2012. ISBN: 0-32182-027-4.
- [7] Chris Bradfield. *Godot Engine Game Development Projects*. Packt, 2018. ISBN: 978-1-78883-150-5.
- [8] Ariel Manzur y George Marques. *Sams Teach Yourself, Godot Engine Game Development in 24 Hours*. Pearson, 2018. ISBN: 0-13483-509-3.
- [9] David Vallejo y Cleto Martín. *Desarrollo de Videojuegos: Un Enfoque Práctico*. CreateSpace, 2015. ISBN: 978-1-51730-955-8.

## Propuesta bibliográfica

Prolog-Scripted Tactics Negotiation and Coordinated Team Actions for Counter-Strike Game Bots [3]

Game Programming Patterns [4]

Game Mechanics [5]

LLVM Cookbook [6]

Godot Engine Game Development Projects [7]

Sams Teach Yourself, Godot Engine Game Development in 24 Hours [8]

Desarrollo de Videojuegos: Un Enfoque Práctico [9]

Logic Programming with Prolog [10]

Prolog Tutorial [11]