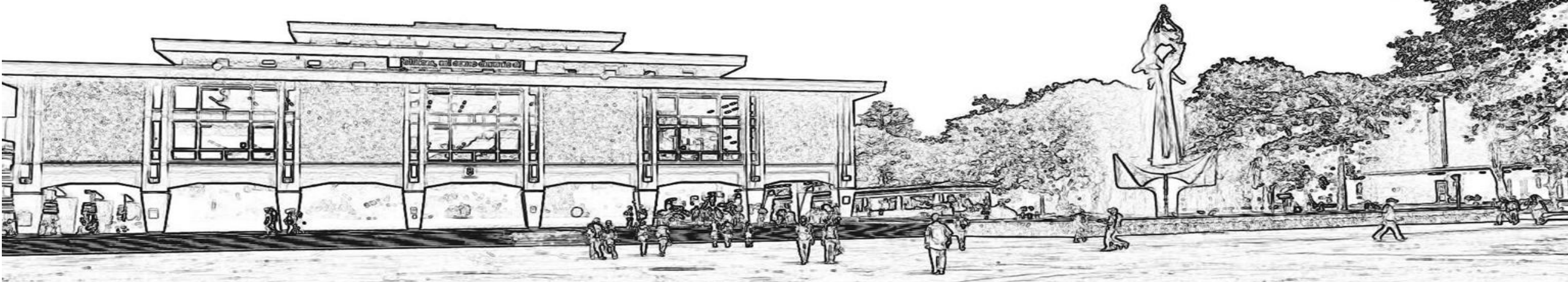




UNIVERSIDAD DE ANTIOQUIA

Facultad de Ingeniería



Tutorial de Simulación Basada en Agentes Parte 1: Una aproximación con ejemplos

Yony Fernando Ceballos, Ph.D
Doctor en Ingeniería - Ingeniero de Sistemas

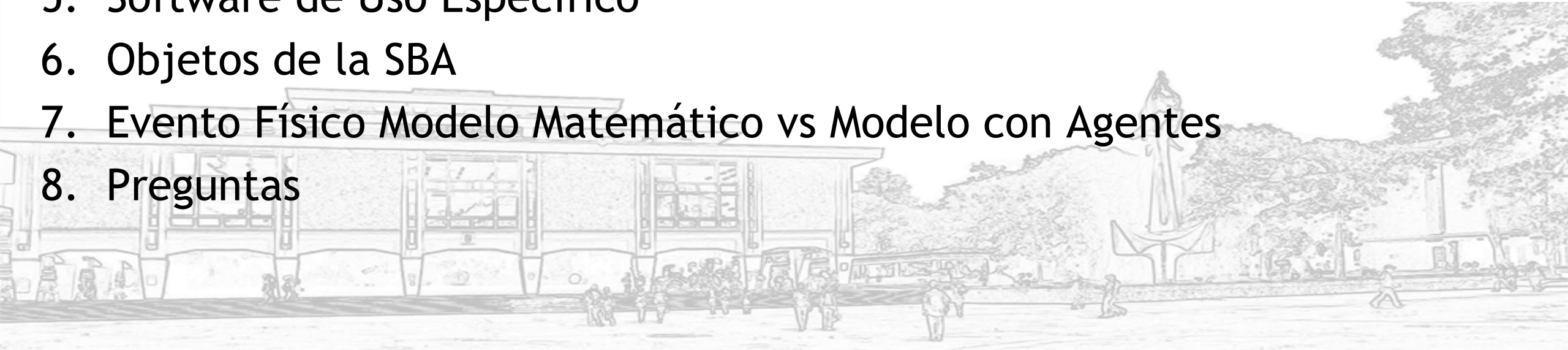
Julián Andrés Castillo G. M.Sc.
Magister en Ingeniería - Ingeniero de Sistemas

Departamento de Ingeniería Industrial

Agenda



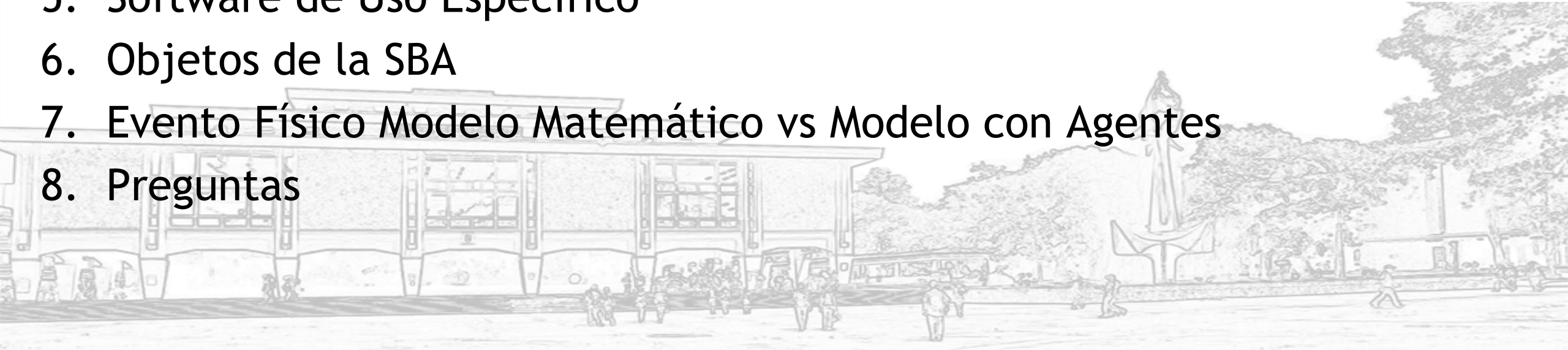
1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



Agenda



1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



Referencias y citaciones

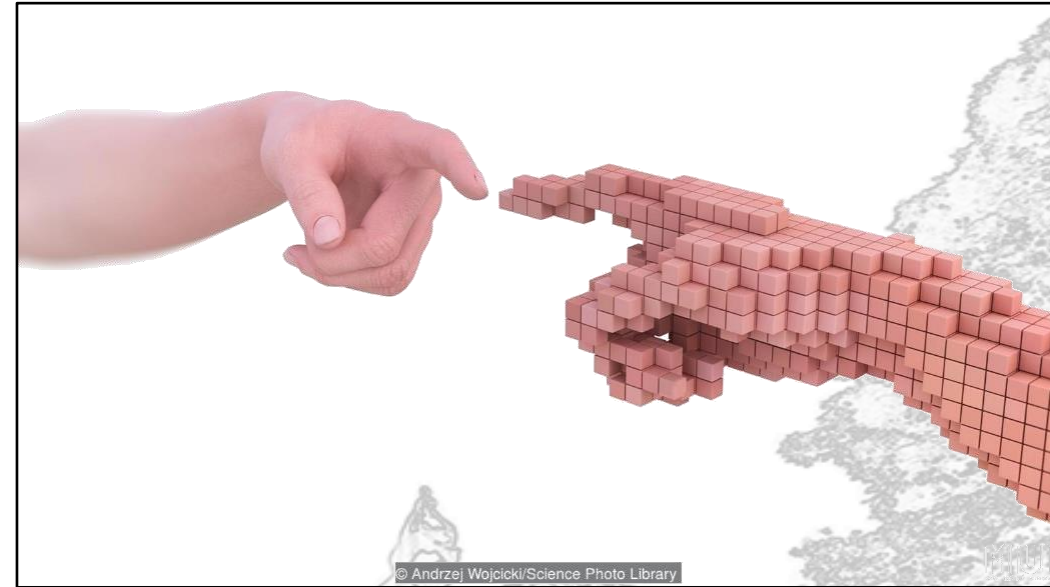
- Todas las referencias y citaciones están disponibles en la parte inferior izquierda de la presentación.
- Solo se muestran las referencias más recientes y actualizadas de los documentos. En algunos casos los años de referencia y citación no coinciden, dado que siempre se presenta el documento más reciente o más completo.



¿Qué es simulación?

Abstraer el comportamiento del mundo real

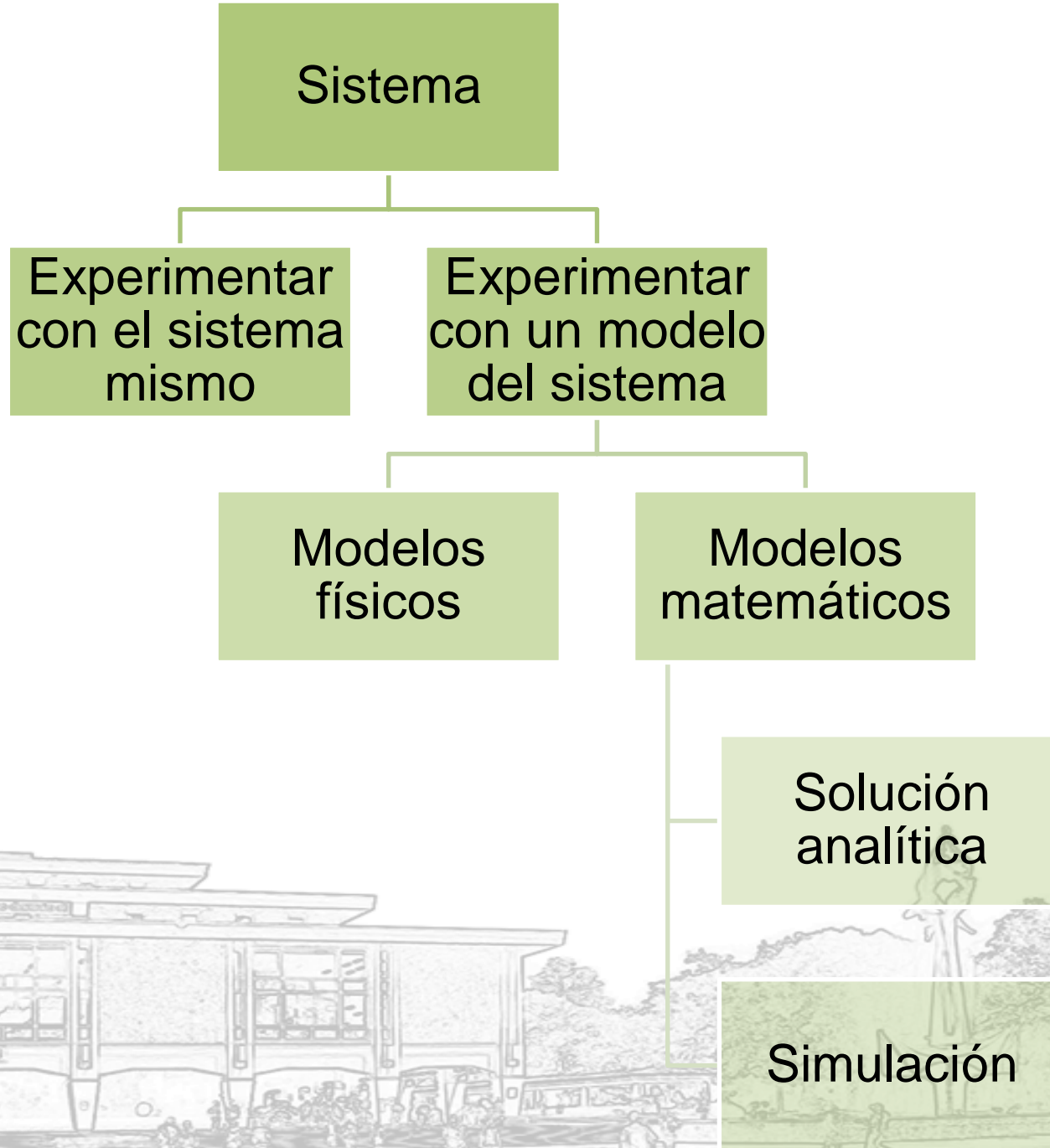
- Modelo es la representación de un sistema mediante un conjunto de relaciones lógico-matemáticas originadas en la observación o percepción del comportamiento del sistema.
- La simulación de un sistema es la creación de un modelo que abstraer el comportamiento del mundo real y que puede ser usado para generar comportamientos artificiales del sistema, de forma tal que permite replicar y predecir diversas situaciones del comportamiento de este.
- Simulación de un sistema (o un organismo) es la operación de un modelo (simulador), el cual es una representación del sistema. Este modelo puede sujetarse a manipulaciones que serían **imposibles de realizar, demasiado costosas o imprácticas**.



<https://www.sciencephoto.com/media/720714/view>

La simulación es una técnica que puede utilizarse para resolver una amplia gama de modelos. Su aplicación es tan **amplia** que se ha dicho: "**Cuando todo falle, utilice simulación**".

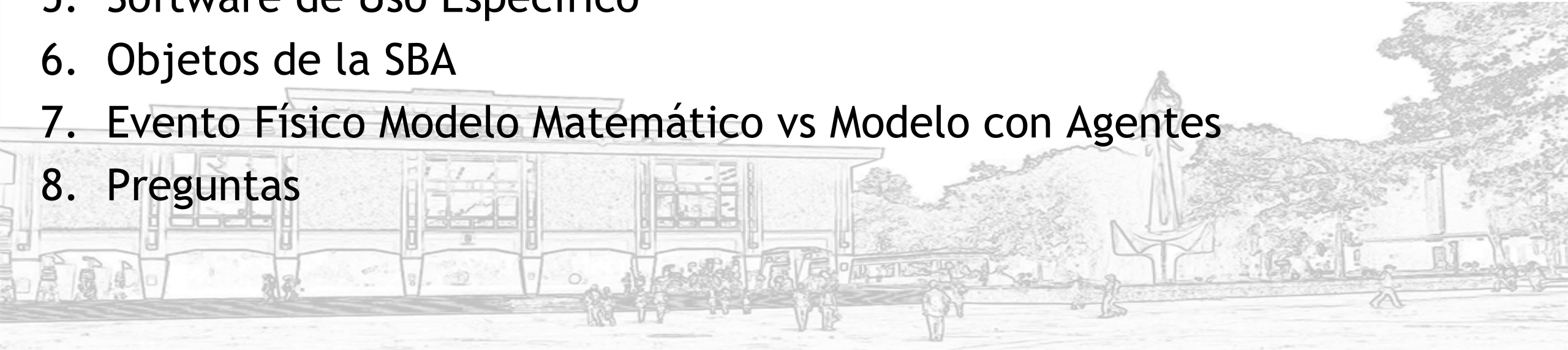
La simulación en los sistemas



Agenda



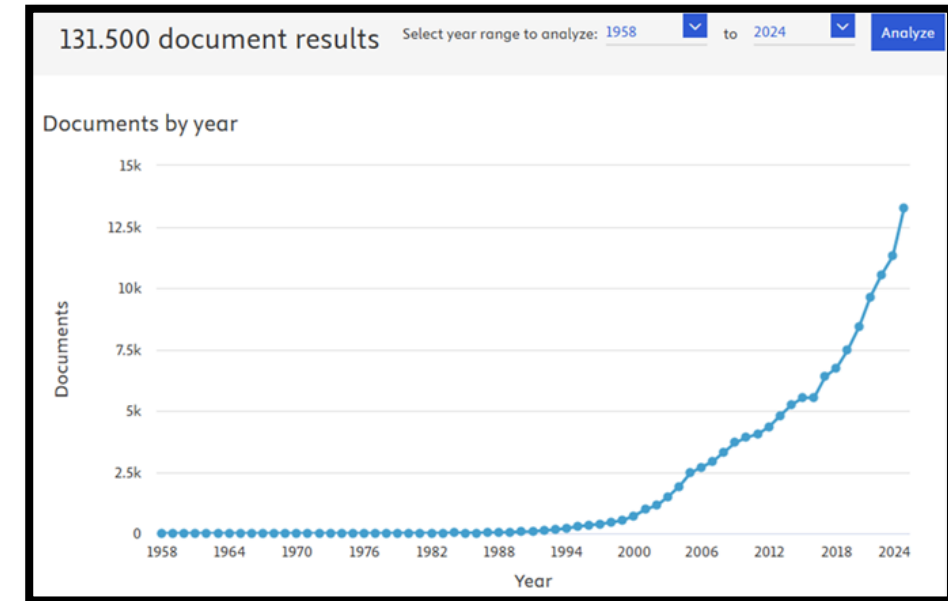
1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



¿Por qué Simulación Basada en Agentes?

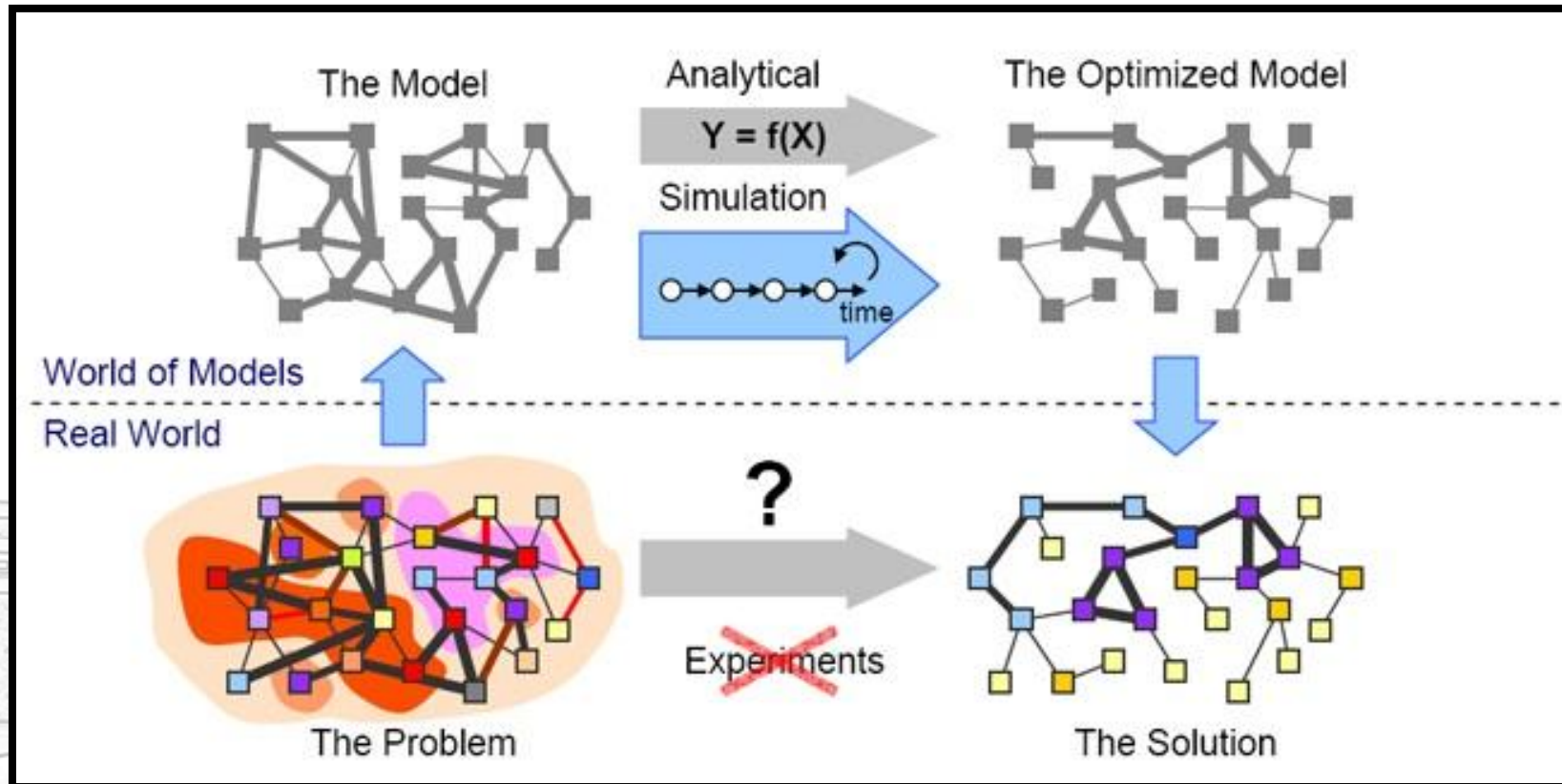
Simulación Basada en Agentes SBA = Agent-based Modelling (modeling) [ABM]

- Un mundo no tan ideal.
 - Tratar de entender la naturaleza y los mundos sociales.
- Un agente es un individuo/objeto computacional autónomo con un conjunto de propiedades y acciones.
 - Agentes en plural
- La simulación basada en agentes es una forma de modelado/simulación computacional en donde un fenómeno es modelado en términos de agentes y sus interacciones.
 - Nuevos tópicos, nuevos mundos y nuevas simulaciones.
- La simulación basada en agentes es una metodología que puede ser aplicada “promiscuamente”.
 - En los últimos 20 años SBA ha crecido casi exponencialmente.



Simulación basada en agentes (SBA)

- SBA analiza el comportamiento de los sistemas a través del estudio de las interacciones entre agentes que siguen ciertas reglas o estados.



Simulación Basada en Agentes

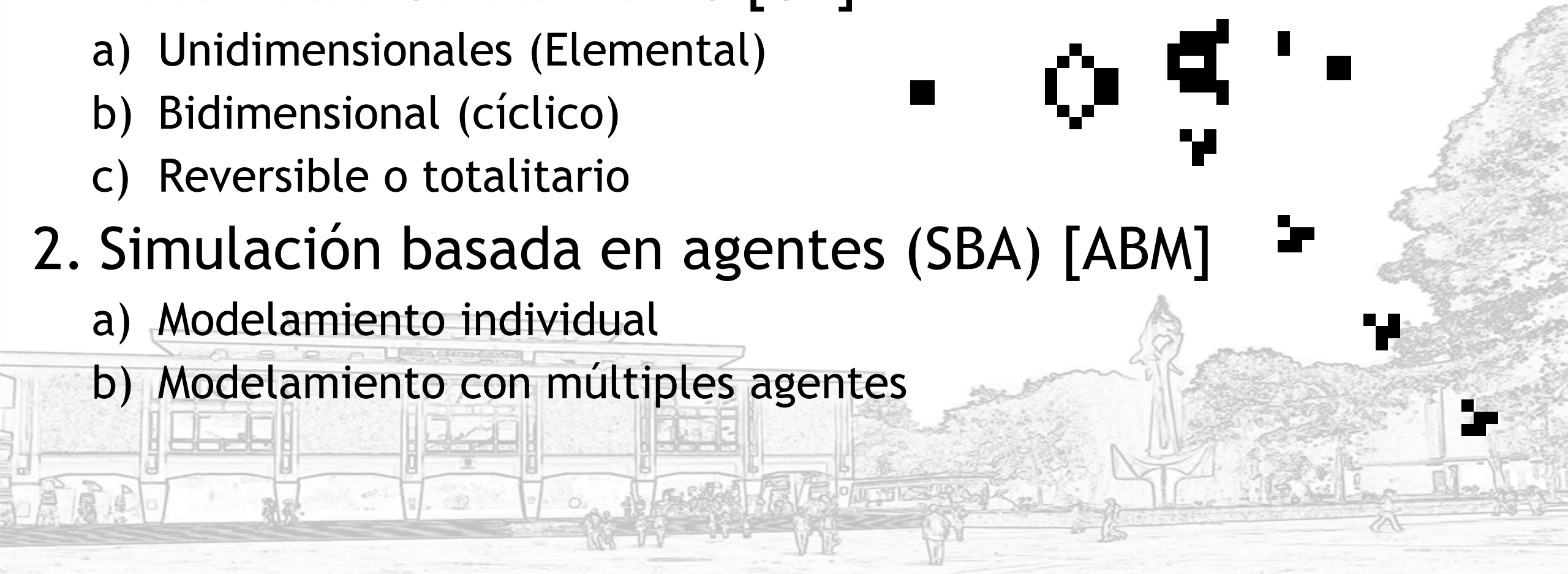
Clasificación

1. Autómatas Celulares AC [CA]

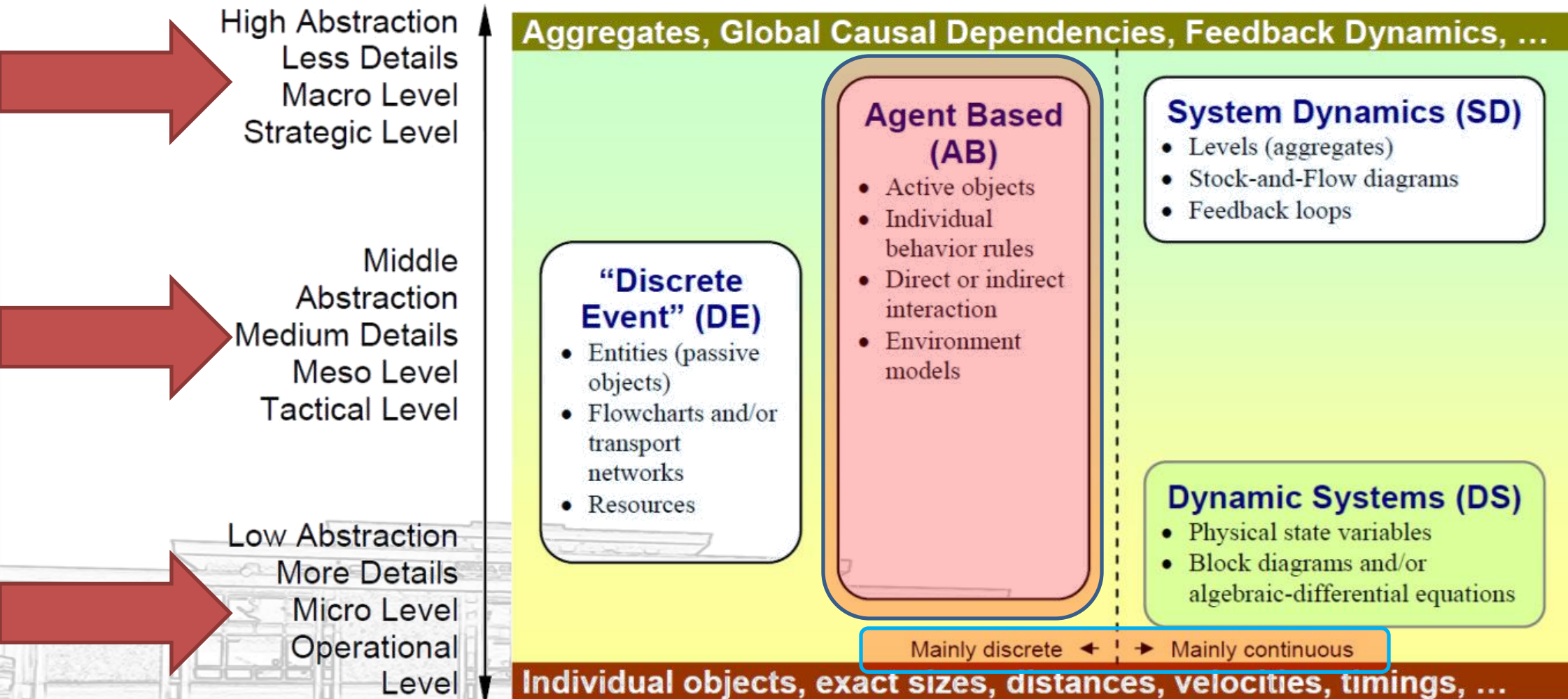
- a) Unidimensionales (Elemental)
- b) Bidimensional (cíclico)
- c) Reversible o totalitario

2. Simulación basada en agentes (SBA) [ABM]

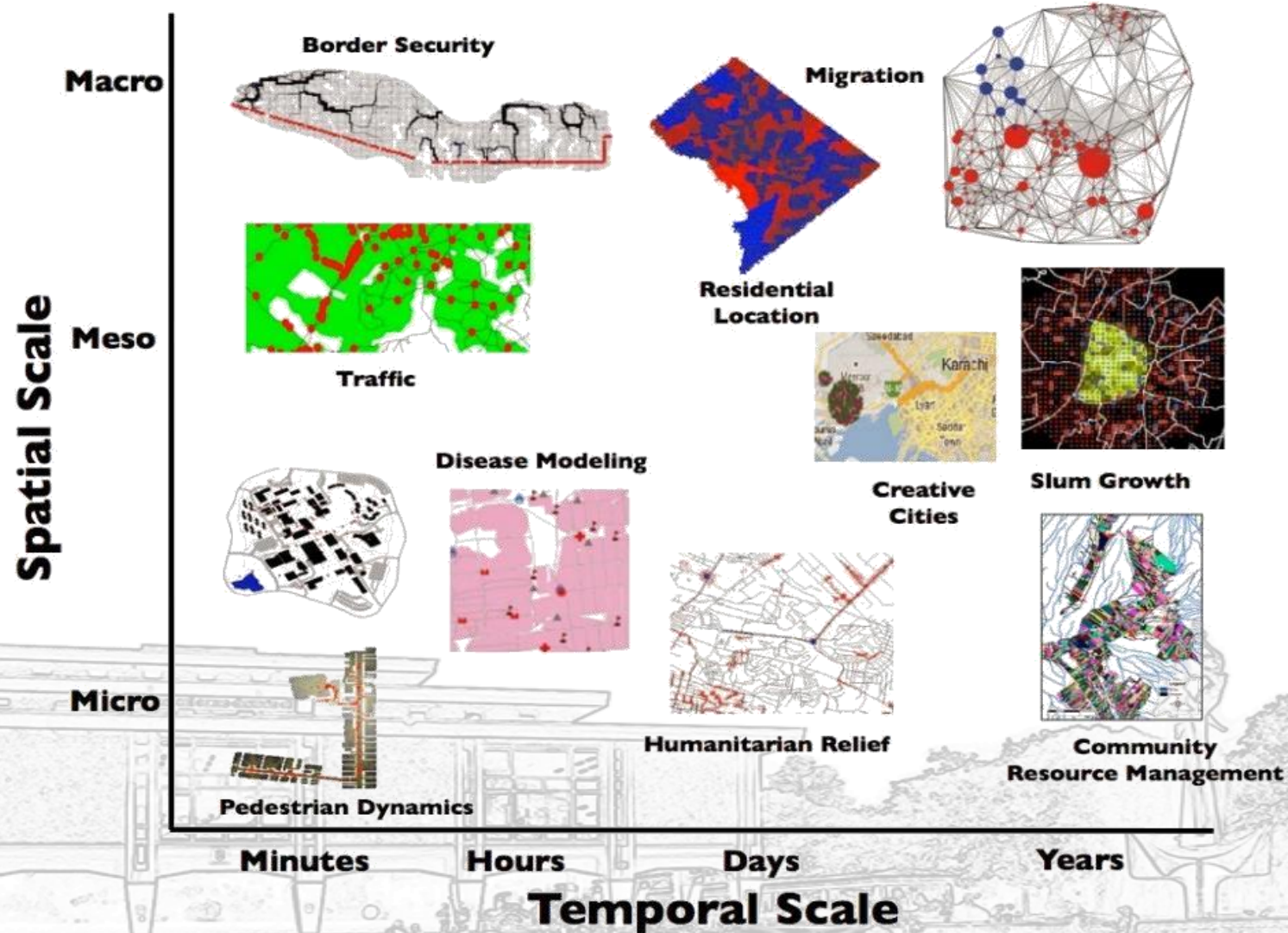
- a) Modelamiento individual
- b) Modelamiento con múltiples agentes



Simulación basada en agentes (SBA)



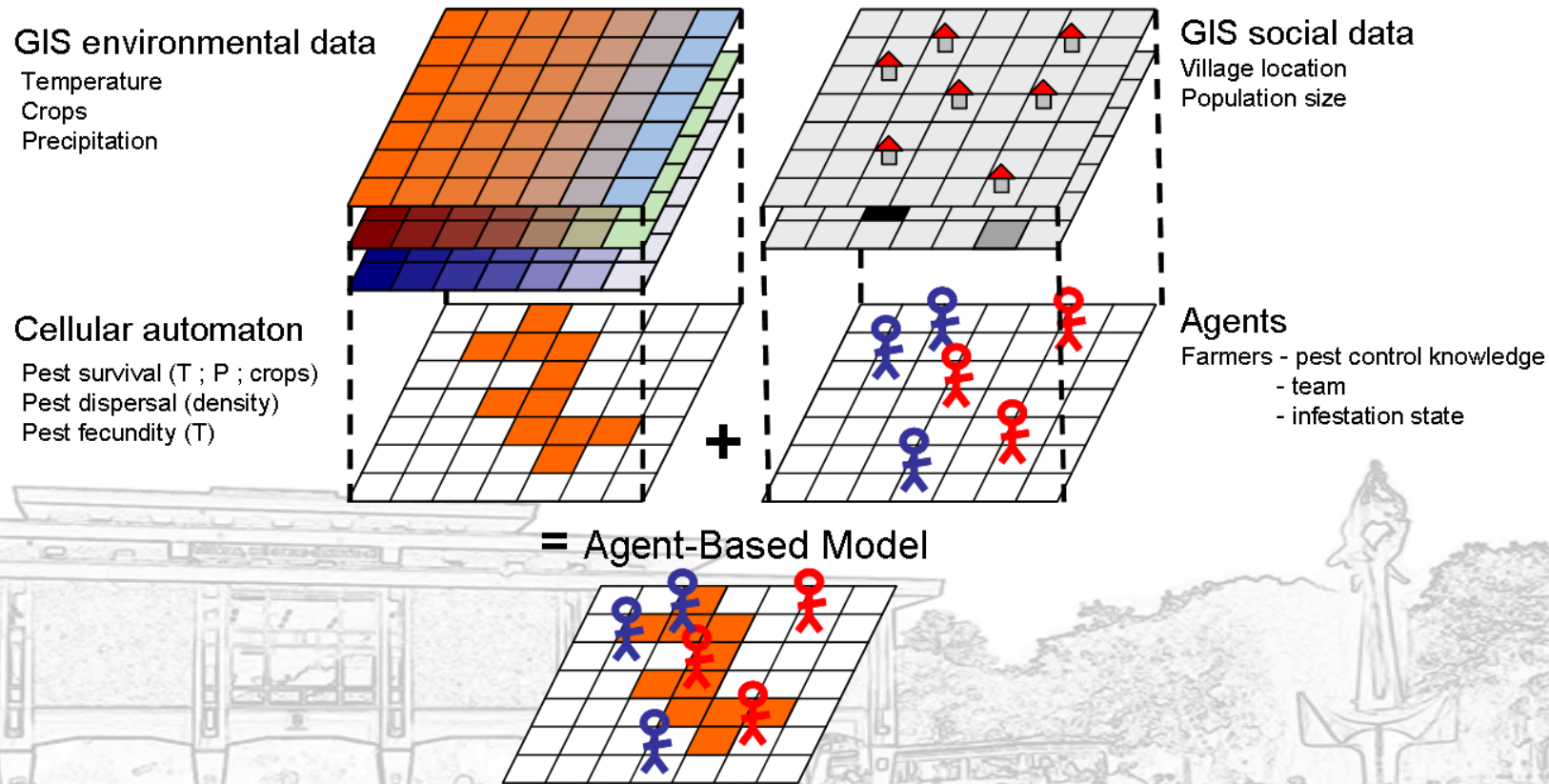
Escala temporal y espacial de SBA



Simulación basada en agentes

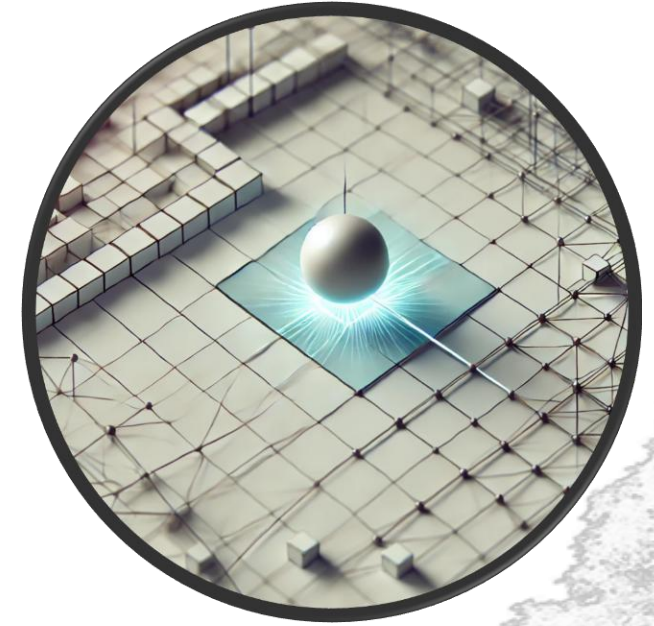
Ejemplo

Un ejemplo de SBA en un control de plagas para la agricultura.



Características de los agentes

Autonomía: Los agentes toman sus propias decisiones con base en información que éste posee y que puede intercambiar con otros agentes o con el entorno.



Heterogeneidad: Los agentes pueden ser diferentes entre sí. Grupos de agentes pueden representar clases diferentes, cada uno con sus propios atributos.

Características de los agentes

Actividad o acciones: Los agentes realizan acciones independientes durante el curso de la simulación y suelen tener: objetivo, reacción, percepción, racionalidad, interacción, comunicación, movilidad entre otros.



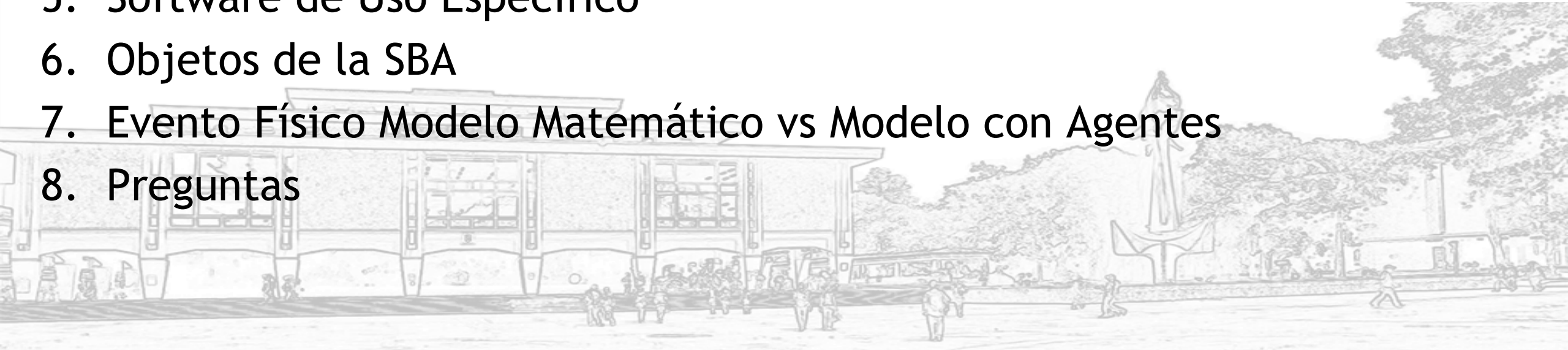
Aprendizaje y adaptación: los agentes pueden tener la habilidad de aprender y adaptarse. Se les puede modelar para tener la capacidad de modificar su estado actual con base en estados anteriores o uso de memoria.

Reglas: conjunto de instrucciones o condiciones que determinan el comportamiento de los agentes dentro del sistema simulado. Estas pueden definir cómo los agentes interactúan entre sí y con su entorno, cómo toman decisiones y cómo evolucionan a lo largo del tiempo (Comportamiento, Interacción y Adaptación).

Agenda



1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



SBA - Historia

- 1940-1950: John Von Neuman desarrolla el concepto de autómatas celulares.
- 1952: Alan Turing propone la teoría de morfogénesis.
- 1969: Thomas Schelling desarrolla el modelo de segregación urbana.
- 1971: John H. Conway crea el juego de la vida.
- 1976: John Holland aporta a la SBA con la Teoría de Sistemas Adaptativos Complejos.
- 1977: Epstein y Axtell empiezan a desarrollar modelos sociales basados en agentes culminando en el libro “Growing artificial societies”

Martín-Sánchez, M., Martín-Sánchez, M. T., & Pinto, G. (2012). Aportaciones de Alan Turing al ámbito de la química: Teoría de la morfogénesis e interpretación de algunas reacciones químicas. *Anales de Química de la RSEQ*, 108(4), 322-322.

Clark, W. A. (1991). Residential preferences and neighborhood racial segregation: A test of the Schelling segregation model. *Demography*, 28, 1-19.

Izhikevich, E. M., Conway, J. H., & Seth, A. (2015). Game of life. *Scholarpedia*, 10(6), 1816.

Holland, J. (1996). *Sistemas adaptativos complejos*.

Epstein, J. M., & Axtell, R. (1996). *Growing artificial societies: social science from the bottom up*. Brookings Institution Press.

SBA - Historia

- 1983: Economistas comienzan a usar modelos basados en racionalidad y metas aplicado a problemas económicos.
- 1986: Christopher Langton introduce el concepto de vida artificial, estudiando cómo reglas simples generan comportamientos emergentes.
- 1992: Joshua Epstein y Robert Axtell publican el modelo Sugarscape, un hito en la simulación social basada en agentes, se define por primera vez el uso de la palabra “agente”.
- 1994: Se lanza la primera versión de Swarm, una de las primeras herramientas especializadas en SBA. Luego se define su versión definitiva en 1997. Usando el lenguaje Objective-C.

SBA - Historia

- 1998: Axelrod usa SBA para modelar la evolución de la cooperación en estrategias de teoría de juegos.
- 1999: Se crea NetLogo un software de modelado basado en agentes usando el lenguaje Logo y LISP.
- 1994-1999: Se comienza a usar el termino “Agent-Based Modeling”.
- 2000: Se crea Repast (Recursive Porous Agent Simulation Toolkit), una plataforma avanzada de simulación basada en agentes usando el lenguaje C++ luego pasando a C#. El mismo año se crea AnyLogic software multiparadigma para la simulación permitiendo escribir en Java modelos de SBA.

SBA - Historia

- 2002-2010: Crece el uso de SBA en múltiples áreas, desde la medicina hasta el comportamiento y crecimiento de las ciudades.
- 2013: Se desarrolla MESA, una librería de Python para el uso en SBA.
- 2015: SBA se usa para elaborar modelos financieros, mercados y desastres naturales con alta precisión.
- 2019: La IA y el aprendizaje profundo comienzan a integrarse en SBA para mejorar la toma de decisiones en entornos complejos.
- 2020-2024: SBA se usa en el modelado de la pandemia de COVID-19, optimización de cadenas de suministro y simulaciones urbanas.

Kazil, J., Masad, D., & Crooks, A. (2020). Utilizing Python for agent-based modeling: The Mesa framework. In R. Thomson, H. Bisgin, C. Dancy, A. Hyder, & M. Hussain (Eds.), *Social, cultural, and behavioral modeling* (pp. 308-317). Springer International Publishing. <https://doi.org/10.1007/978-3-030-61255-9>

Bookstaber, R. (2017). Agent-based models for financial crises. *Annual Review of Financial Economics*, 9(1), 85-100.

Schlögl, M., Richter, G., Avian, M., Thaler, T., Heiss, G., Lenz, G., & Fuchs, S. (2019). On the nexus between landslide susceptibility and transport infrastructure-an agent-based approach. *Natural hazards and earth system sciences*, 19(1), 201-219.

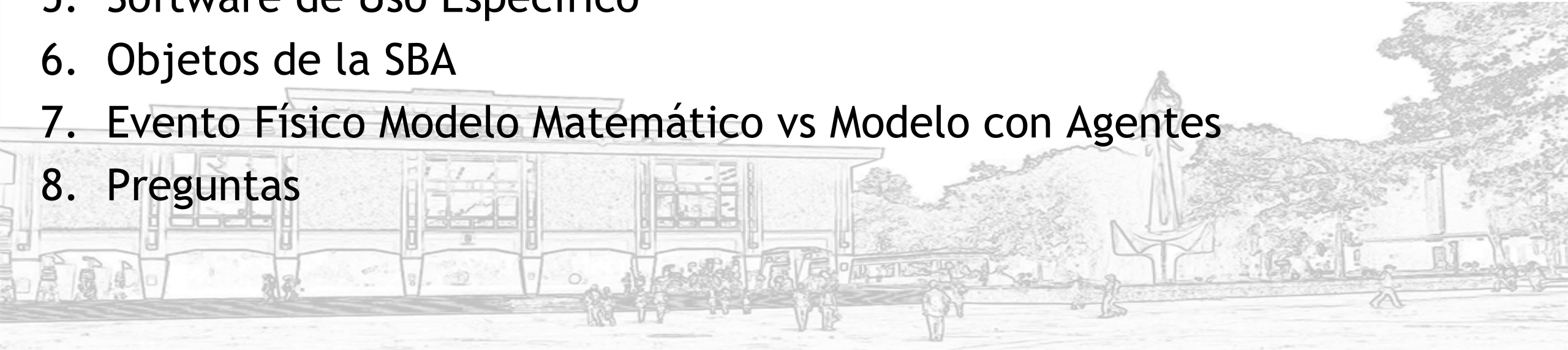
Kerr, C. C., Stuart, R. M., Mistry, D., Abey Suriya, R. G., Rosenfeld, K., Hart, G. R., ... & Klein, D. J. (2021). Covasim: an agent-based model of COVID-19 dynamics and interventions. *PLOS Computational Biology*, 17(7), e1009149.

Omarov, B., Altayeva, A., Turganbayeva, A., Abdulkarimova, G., Gusmanova, F., Sarbasova, A., ... & Omarov, N. (2019). Agent based modeling of smart grids in smart cities. In *Electronic Governance and Open Society: Challenges in Eurasia: 5th International Conference, EGOSE 2018, St. Petersburg, Russia, November 14-16, 2018, Revised Selected Papers 5* (pp. 3-13). Springer International Publishing.

Agenda



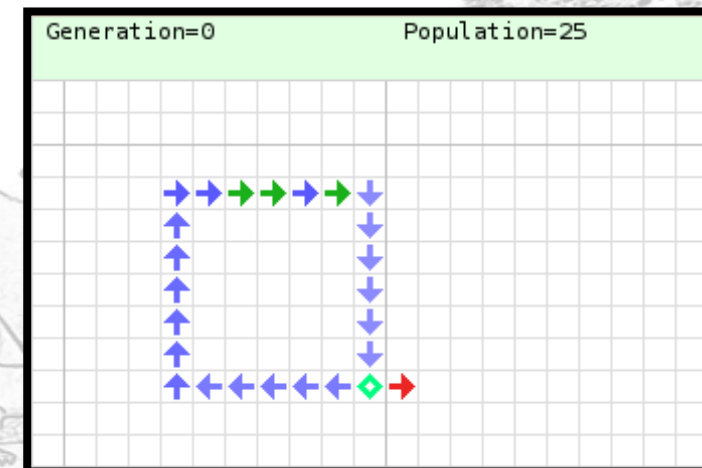
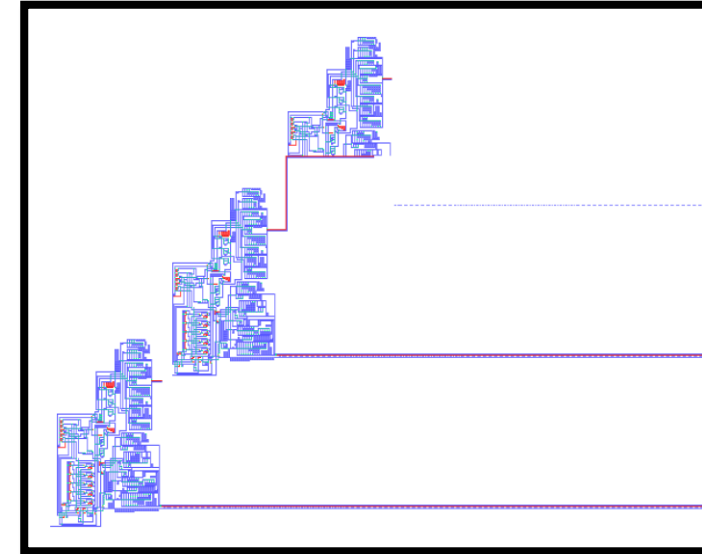
1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. **Autómatas Celulares**
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



Autómatas Celulares

El primer autómatas

- Stan Ulam y John Von Neuman trabajando juntos en “Los Alamos National Laboratory” llegaron al primer concepto de Automata celular (CA - Cellular Automaton).
 - Von Neuman crea el campo de los autómatas celulares en una época en donde no habían computadores, no habían ayudas digitales y todo su trabajo era sobre papel.
- Los primeros conceptos de autómatas.
 - Uno de los primeros conceptos de autómatas es realizado por Ulam y Von Neuman en donde consideraron un líquido como un grupo de unidades discretas y calcularon el movimiento de acuerdo con los movimientos de los vecinos.



Clasificación de los autómatas celulares

Stephen Wolfram crea una clasificación para autómatas celulares detallada en cuatro clases. Las 4 clases se detallan a continuación:

- **Clase 1:** Uniformidad. Los CA de clase 1 terminan después de un cierto número de generaciones.
- **Clase 2:** Repetición. Los CA de clase 2 permanecen estables, pero los estados de celda no son constantes. Más bien, oscilan en algún patrón regular de 0 a 1 a 0 a 1 y así sucesivamente.
- **Clase 3:** Aleatorio. Los CA de clase 3 parecen aleatorios y no tienen un patrón fácilmente distinguible.
- **Clase 4:** Complejidad. Los CA de clase 4 se pueden considerar como una mezcla entre la clase 2 y la clase 3. Los CA de clase 4 exhiben las propiedades de los sistemas complejos.

Autómata celular elemental o unidimensional

Con base en la definición necesitamos tres elementos fundamentales.

1. Celdas: El más sencillo es una línea unidimensional de celdas continuas.



2. Estado: El estado más simple es 1 y 0, encendido y apagado.

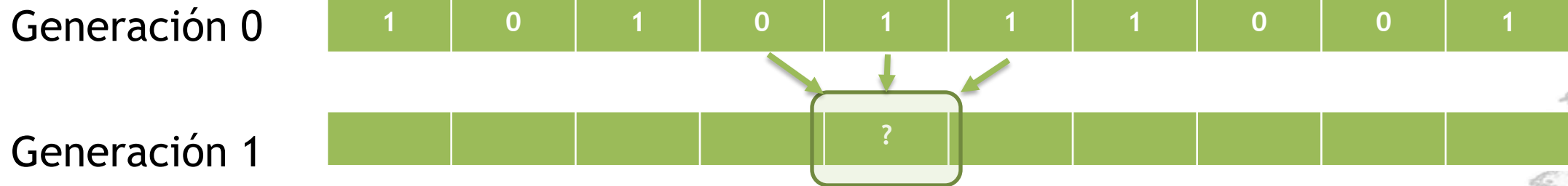


3. Vecindario: El vecindario en una dimensión son sus vecinos de izquierda y derecha.



Autómata celular elemental o unidimensional

Calculamos el presente estado de una CELDA de acuerdo con las reglas del estado de tiempo anterior.



Podemos mirar todas las configuraciones posibles de una CELDA con su vecino y definir el resultado del estado para cada configuración posible.

¿De cuántas formas posibles podemos configurar los estados?

Tres celdas definen un número de 3 bits, y con esos 3 bits puedes contar hasta 8.

000	001	010	011	100	101	110	111
-----	-----	-----	-----	-----	-----	-----	-----

Autómata celular elemental o unidimensional

Debemos definir una regla luego de tener todos los datos de configuración de estados. Para el presente ejercicio la regla será de la siguiente manera.

000	001	010	011	100	101	110	111
↓	↓	↓	↓	↓	↓	↓	↓
0	1	0	1	1	0	1	0

Según Wolfram el estándar para iniciar es tener todas las celdas en cero excepto la celda del centro que inicia en 1. (Se recomienda una configuración impar para tener un centro específico)

0	0	0	0	1	0	0	0	0
---	---	---	---	---	---	---	---	---

Autómata celular elemental o unidimensional

Apliquemos el siguiente set de reglas descritas previamente como se muestra a continuación a un estado inicial

2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7
000	001	010	011	100	101	110	111
↓	↓	↓	↓	↓	↓	↓	↓
0	1	0	1	1	0	1	0

Se define el estado inicial

0	0	0	0	1	0	0	0	0
↓	↓	↓	↓	↓	↓	↓	↓	↓
0	0	0	1	0	1	0	0	0

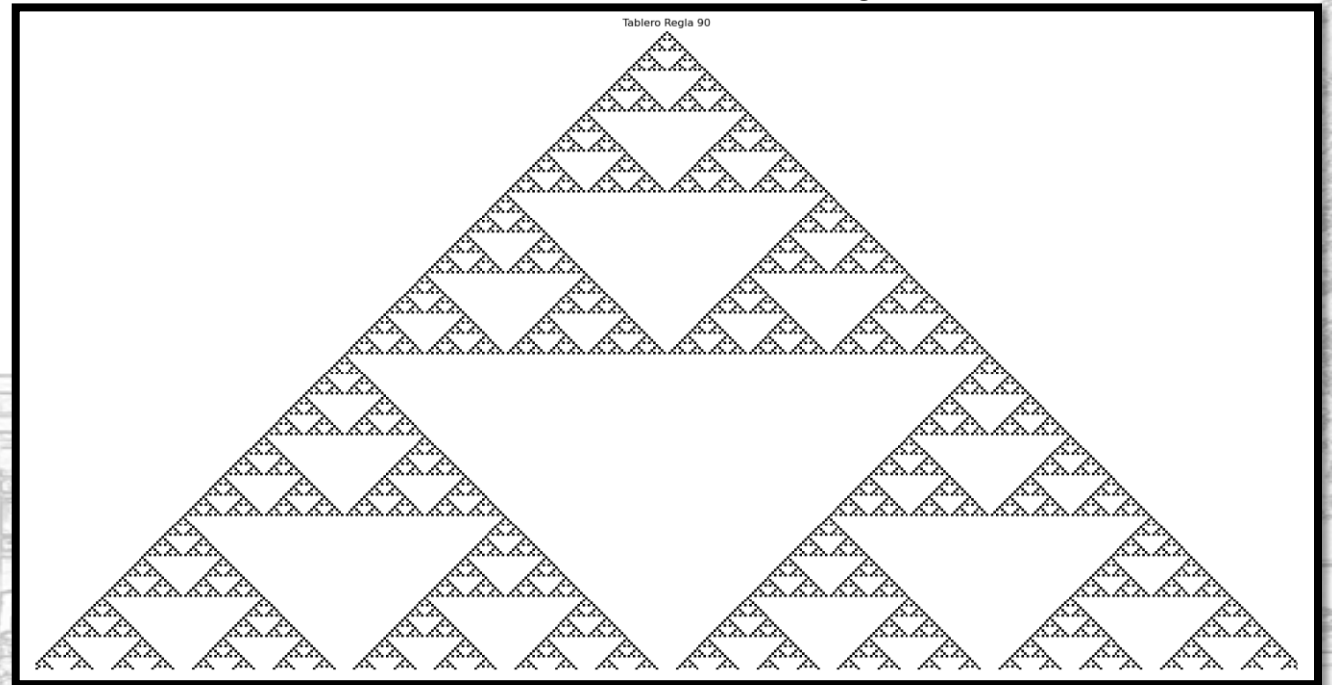
Realizamos las operaciones

El triángulo de Sierpiński

La forma de baja resolución (pocas celdas) que realizamos en la diapositiva anterior es el "triángulo de Sierpiński". Nombrado en honor al matemático polaco Wacław Sierpiński, es un patrón fractal y un sistema increíblemente simple de 0 y 1 con pequeños vecindarios de tres celdas que pueden generar una forma triangular. Esta regla es conocida como 90 debido a su patrón en 8 bits.

$$01011010_2 = 90_{10}$$

El triángulo de Sierpiński es uno de los fractales más famosos y aparece en diferentes ramas de las matemáticas, como la teoría de conjuntos y geometría fractal.



The Rule



Hay 256 autómatas de este tipo, cada uno de los cuales puede ser indexado por un número binario único cuya representación decimal se conoce como la "regla" (the rule) para el autómata en particular. A continuación se muestran ocho autómatas básicos iniciales y su configuración.

Ruta --> src/celular_automata/Reglas.xlsx

Regla 30	00011110							
Valor Regla	0	0	0	1	1	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	0	0	16	8	4	2	0
Suma	30							

Regla 60	00111100							
Valor Regla	0	0	1	1	1	1	0	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	0	32	16	8	4	0	0
Suma	60							

Regla 90	01011010							
Valor Regla	0	1	0	1	1	0	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	64	0	16	8	0	2	0
Suma	90							

Regla 102	01100110							
Valor Regla	0	1	1	0	0	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	64	32	0	0	4	2	0
Suma	102							

Regla 54	00110110							
Valor Regla	0	0	1	1	0	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	0	32	16	0	4	2	0
Suma	54							

Regla 62	00111110							
Valor Regla	0	0	1	1	1	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	0	32	16	8	4	2	0
Suma	62							

Regla 94	01011110							
Valor Regla	0	1	0	1	1	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	64	0	16	8	4	2	0
Suma	94							

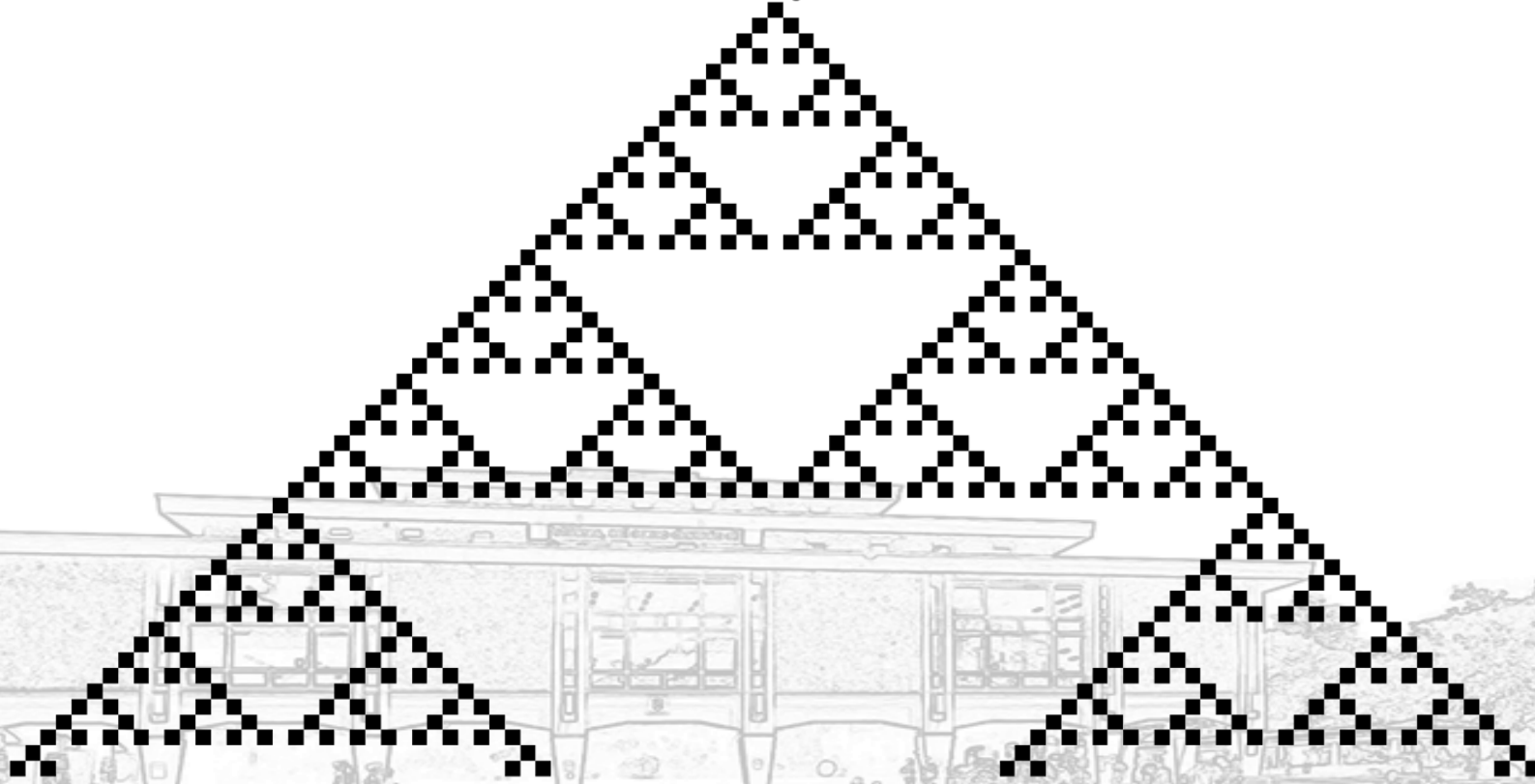
Regla 110	01101110							
Valor Regla	0	1	1	0	1	1	1	0
Valor Potencia Base 2	128	64	32	16	8	4	2	1
Valor Regla y Potencia	0	64	32	0	8	4	2	0
Suma	110							







Regla 90 --> Python



A continuación detallaremos el código en Python de la elaboración de la regla 90 mostrando su estructura y secuencia aplicando condicionales posicionales a su ejecución. Código disponible en GitHub, escanea el QR. Ruta --> [/src/celular_automata/Rule90_BasicLists.py](https://github.com/src/celular_automata/Rule90_BasicLists.py)

Tablero Regla 90



 TIOBE <small>(the software quality company)</small>					
Feb 2025	Feb 2024	Change	Programming Language	Ratings	Change
1	1		 Python	23.88%	+8.72%
2	3	▲	 C++	11.37%	+0.84%
3	4	▲	 Java	10.66%	+1.79%
4	2	▼	 C	9.84%	-1.14%
5	5		 C#	4.12%	-3.41%

El Mundo de SBA

Varios mundos: SBA tiene mundo 1D, 2D y 3D

1D: Autómatas celulares

0	1	0	0	1	0	0	1	0	0	1	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---

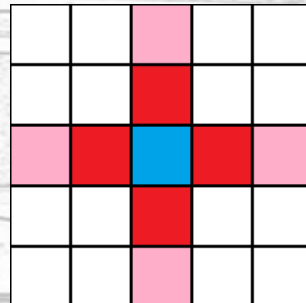
2D: Autómatas celulares y Agentes

3D: Autómatas celulares y Agentes

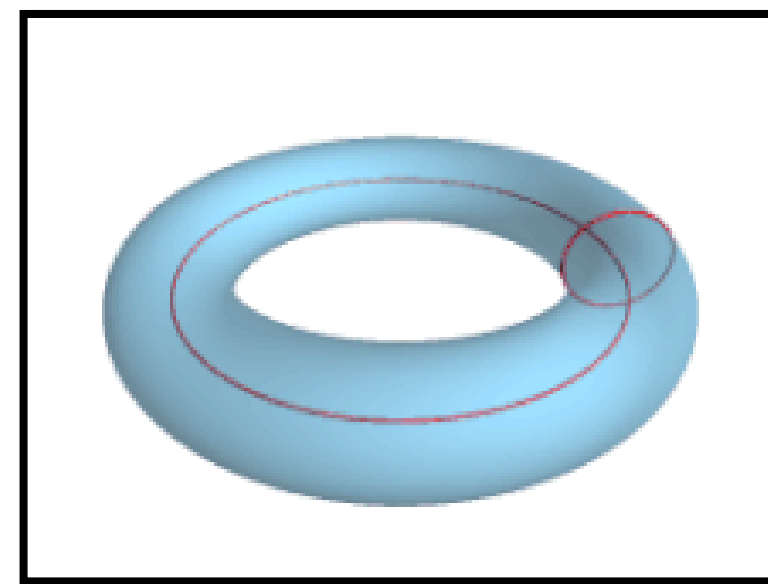
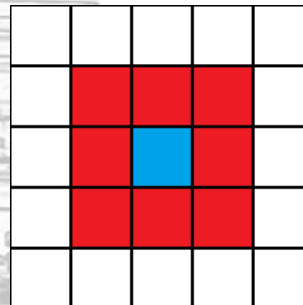
Un Torus es una superficie de revolución generada por la rotación de un círculo en el espacio tridimensional sobre un eje que es coplanar con el círculo y asemeja a un universo continuo utilizado en SBA.

Vecindarios: Los vecindarios de von Neumann y el vecindario de Moore son los dos principales.

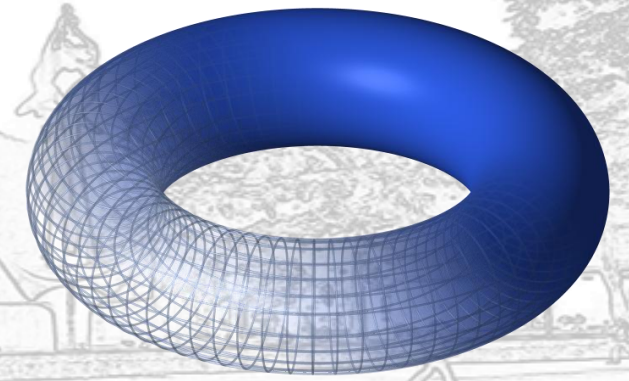
2D von Neuman



2D Moore



0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	1	1	1	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0
1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	1



Autómatas Bidimensionales

Al pasar de una a dos dimensiones agregamos mayor complejidad al sistema, cada celda tendrá un vecindario mayor y como se describió previamente podemos utilizar el vecindario von Neuman, von Neuman extendido, Moore o incluso un vecindario específico.

Para comprender mejor el modelo bidimensional utilizaremos el famoso “**Game of Life**” detallado en 1970 en el artículo “Scientific American” que documentó el matemático John Conway describiéndolo como “recreacional” y que el lector podría sacar un juego de ajedrez y simplemente “jugar”.

The Game of Life

Un autómata clase 4 - Reglas

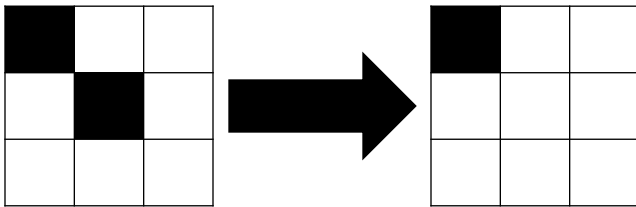
Las reglas son simples y son las siguientes:

- Muerte: una celda viva *estado* = 1 muere y pasa a *estado* = 0 bajo las siguientes circunstancias:
 - Sobrepoblación: si la celda tiene cuatro o mas vecinos vivos muere.
 - Soledad: si la celda tiene uno o cero vecinos muere.
- Nacimiento: una celda muerta *estado* = 0 vive o nace y pasa a *estado* = 1 si tiene exactamente tres vecinos vivos, ni más ni menos que tres.
- Neutral: En los demás casos la celda no cambia, los casos son los siguientes:
 - Mantenerse con vida: una celda con dos o tres vecinos permanece viva.
 - Permanece muerta: una celda muerta con menos de tres vecinos vivos permanece muerta.



The Game of Life

Muerte, solo un vecindario vivo



Vida, tres vecindarios vivos



El juego trata cada generación como un solo fotograma en una animación. Así que en lugar de ver todas las generaciones a la vez, solo vemos una a una. En programación tradicional se debería tener dos matrices para evaluar su ejecución.

A continuación detallaremos el código en Python de la elaboración de la Juego de la Vida mostrando su estructura y secuencia aplicando una estructura con PyGame. Código disponible en GitHub, escanea el QR. Ruta --> [/src/celular_automata/PyGameOfLifeV3.py](https://src/celular_automata/PyGameOfLifeV3.py)

Izhikevich, E. M., Conway, J. H., & Seth, A. (2015). Game of life. Scholarpedia, 10(6), 1816.

(<http://www.ibiblio.org/lifepatterns/october1970.html>) 223 (October 1970): 120-123.

Recomendación: The Art of Code, by Dylan Beattie --> <https://www.youtube.com/watch?v=6avJHaC3C2U>

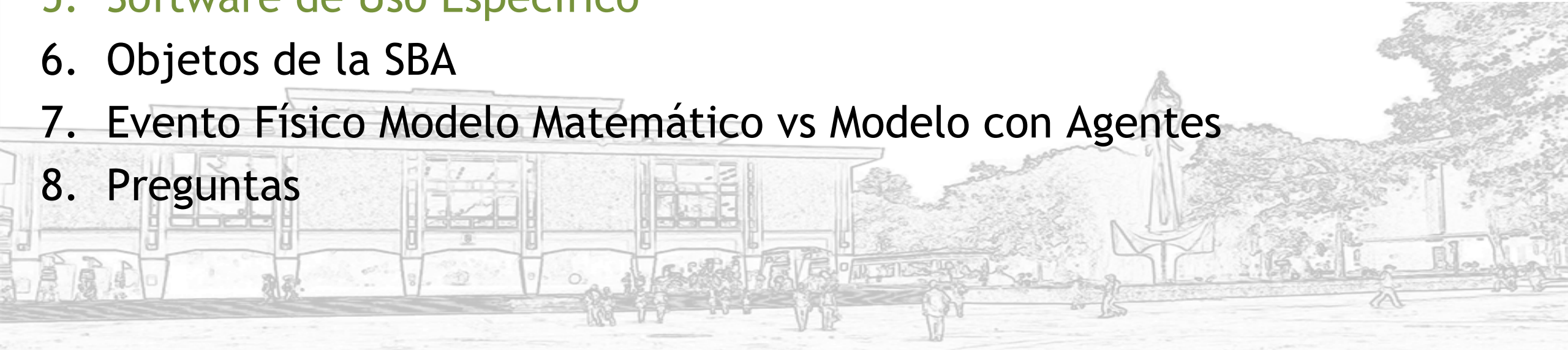
Librerías en Python de Autómatas y Agentes

- [Mesa](#): Mesa es una librería en Python para modelado basado en agentes, que incluye soporte para autómatas celulares. Permite definir reglas personalizadas, visualizar simulaciones en tiempo real y analizar datos. Es útil para simular sistemas complejos, como el tráfico, la propagación de epidemias y modelos económicos.
- [CellPyLib](#): Es una biblioteca muy completa para trabajar con autómatas celulares unidimensionales y bidimensionales. Soporta vecindarios de Moore y von Neumann, y permite crear autómatas celulares elementales y con reglas totalitarias. También incluye funciones para visualizar y analizar la evolución de los autómatas celulares.
[GitHub](#)
- [Golly](#): Aunque es más conocida como una aplicación de escritorio, Golly también tiene una API de Python que puedes usar para trabajar con autómatas celulares. Es especialmente útil para simular el Juego de la Vida de Conway y otros autómatas celulares bidimensionales.

Agenda



1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



Software de Uso Específico

Simulación basada en agentes

El autor Sameera Abar y su equipo de trabajo en el año 2017, realiza un documento en el cual detalla el Software disponible para desarrollo de SBA llamado: Agent Based Modelling and Simulation tools: A review of the state-of-the-art software.



Software de Uso Específico

Simulación basada en agentes






El más utilizado y con un nivel de complejidad bajo y con alto dominio es NetLogo.

ABMS Software Tool ----- License / Availability	Source Code	Type of Agent based on its Interaction Behaviour	Coding Language or Application Programming Interface (API) for Model Development ----- Integrated Development Environment (IDE)	Compiler ----- Operating System (OS) ----- Implementation Platform	Model Development Effort	Modelling Strength / Simulation Models' Scalability Level	ABMS Scope or Application Domain
NetLogo http://cci.northwestern.edu/netlogo/ ----- Open source, GPL, Free	Scala code compilation to Java byte-code (fully interoperable with Java and other JVM codes) ----- For 3D graphical visualisation, NetLogo uses a Java JOGL API for OpenGL rendering	Active objects with simple goals implemented as mobile agents (turtles, patches, links, and the observer)	Models library available ----- NetLogo language	Any Java Virtual Machine with version 5 or later ----- Any platform: Windows 7, Vista, 2000, and XP, Mac OS X, Linux, Unix ----- Desktop computer	Simple/Easy	Medium-scale ~ Large-scale	2D/3D simulations in social and natural sciences, teaching/research

Software de Uso Específico Simulación basada en agentes

Review

Experimenting with Agent-Based Model Simulation Tools

Alessia Antelmi ¹, Gennaro Cordasco ², Giuseppe D'Ambrosio ^{1,*}, Daniele De Vinco ¹ and Carmine Spagnuolo ¹

¹ Dipartimento di Informatica, Università degli Studi di Salerno, 84084 Fisciano, Italy

² Dipartimento di Psicologia, Università degli Studi della Campania "Luigi Vanvitelli", 81100 Caserta, Italy

* Correspondence: gdambrosio@unisa.it

Abstract: Agent-based models (ABMs) are one of the most effective and successful methods for analyzing real-world complex systems by investigating how modeling interactions on the individual level (i.e., micro-level) leads to the understanding of emergent phenomena on the system level (i.e., macro-level). ABMs represent an interdisciplinary approach to examining complex systems, and the heterogeneous background of ABM users demands comprehensive, easy-to-use, and efficient environments to develop ABM simulations. Currently, many tools, frameworks, and libraries exist, each with its characteristics and objectives. This article aims to guide newcomers in the jungle of ABM tools toward choosing the right tool for their skills and needs. This work proposes a thorough overview of open-source general-purpose ABM tools and offers a comparison from a two-fold perspective. We first describe an off-the-shelf evaluation by considering each ABM tool's features, ease of use, and efficiency according to its authors. Then, we provide a hands-on evaluation of some ABM tools by judging the effort required in developing and running four ABM models and the obtained performance.

Keywords: agent-based model; agent-based simulations; agent-based tools; open-source software

Table 7. Summary of whether each model was available for a given platform (●) or has been developed from scratch (○). We only considered the ABM tools that could be installed and used.

↓ Tool/Model →	Flockers	Schelling	Wolf, Sheep, and Grass	ForestFire
ActressMAS	○	●	○	○
AgentPy	●	○	●	●
Agents.jl	●	●	●	●
CppyABM	○	●	○	○
GAMA	●	○	●	○
krABMaga	●	●	●	●
MASON	●	●	○	○
Mesa	●	●	●	●
NetLogo	●	●	●	●
Repat	●	●	●	○

Benchmark configurations. All experiments were performed on the same Ubuntu 22.04 LTS x86_64 machine with kernel version 5.15.0-48-generic and equipped with an Intel i7-8700T (12) @ 4.000 GHz CPU, an NVIDIA GeForce GTX 1050 Mobile GPU, and 16GB RAM. The performance of each framework was tested with different models configurations, starting with a field of size 100×100 , 1000 agents, and 200 steps, while maintaining an agent density of $\cong 10\%$, calculated as $\frac{\text{width} \times \text{height}}{\text{number of agents}}$. We obtained the other configurations by doubling the number of agents and changing the field dimension to preserve the agent density. Table 8 lists all experiment configurations.

NetLogo fue diseñado por Uri Wilensky, con base en el lenguaje de programación Logo y LISP.

Enseña conceptos de programación utilizando agentes en forma de **tortugas**, **parches**, **enlaces** y **el observador**. NetLogo fue diseñado para múltiples audiencias en mente sin una necesidad alta de programación para los fenómenos relacionados con el modelo a representar.



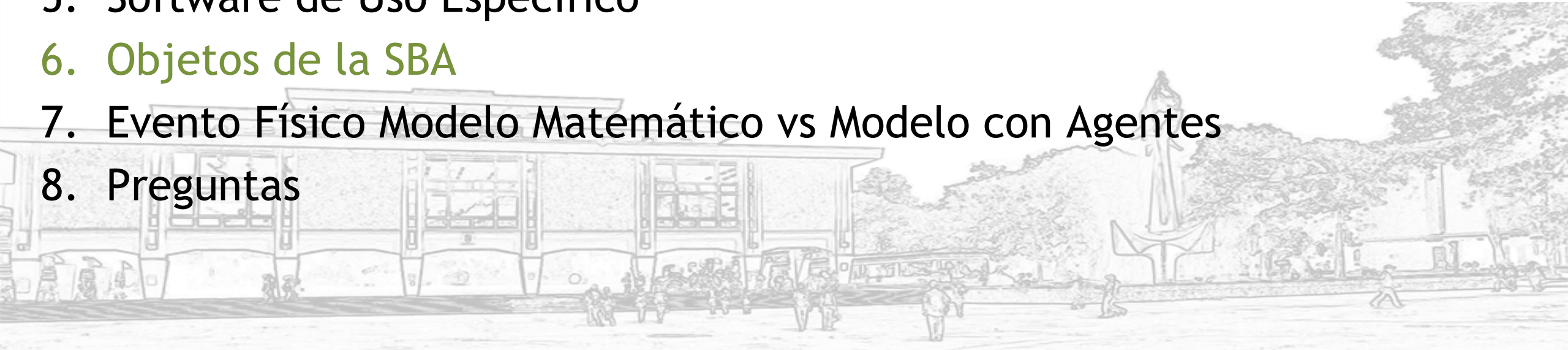
NetLogo graphical user interface	
Paradigms	multi-paradigm: educational, procedural, agent-based, simulation
Family	<u>Lisp</u> , <u>Logo</u>
Designed by	Uri Wilensky
Developer	<u>Northwestern University Center for Connected Learning and Computer-Based Modeling</u>
First appeared	1999;
Stable release	6.4.0 / November 2023
Typing discipline	Dynamic, strong
Scope	Lexical
Implementation language	Scala, Java
Platform	IA-32, x86-64
OS	Cross-platform: JVM
License	GPL (General Public License)
Filename extensions	.nlogo, .nlogo3d, .nls
Website	ccl.northwestern.edu/netlogo

<https://en.wikipedia.org/wiki/NetLogo>

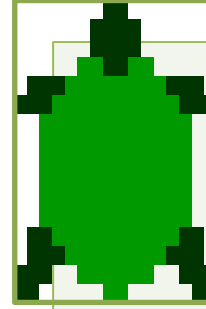
Agenda



1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas

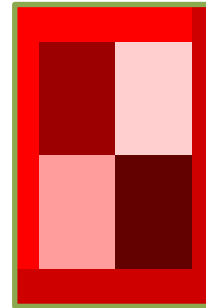


Los agentes en NetLogo tienen características detalladas y datos encapsulados (atributos y propiedades) que detallan métodos de comportamiento. Los agentes se dividen de la siguiente manera:



Turtles

- Agentes
- Movimiento



Patches

- Espacio cuadrado
- Tortugas interactúan con el entorno



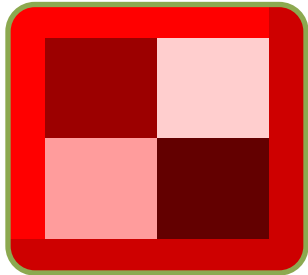
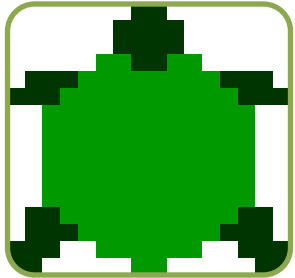
Observer

- Observador externo
- Control

NetLogo



Agentes



- Datos & Variables
 - Humano
 - Género, Edad, Raza, Estatura, etc.
- Procedimientos
 - Agentes – Condiciones
 - Mover si es infeliz, envejecer, cambiar color, cambiar y acumular energía, perder energía.



Parches en color negro y verde.

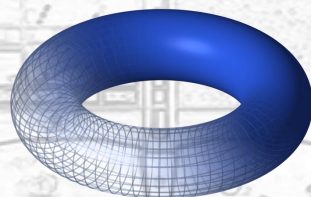
Parches son cuadrados generados aleatoriamente o a gusto por el modelador.



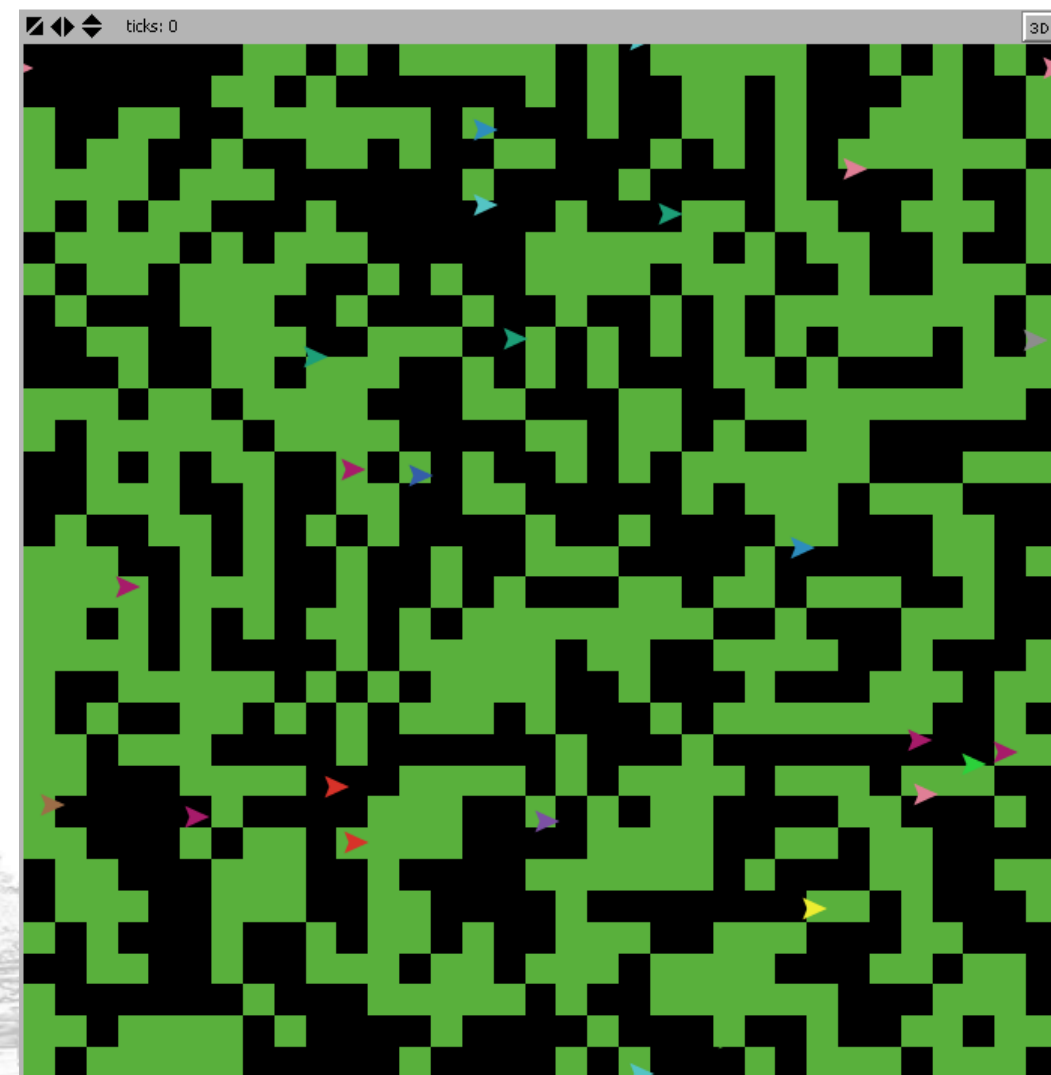
Las tortugas son en varios colores y siluetas, se pueden configurar en diferentes formas, por defecto son flechas y pueden moverse “libremente”.



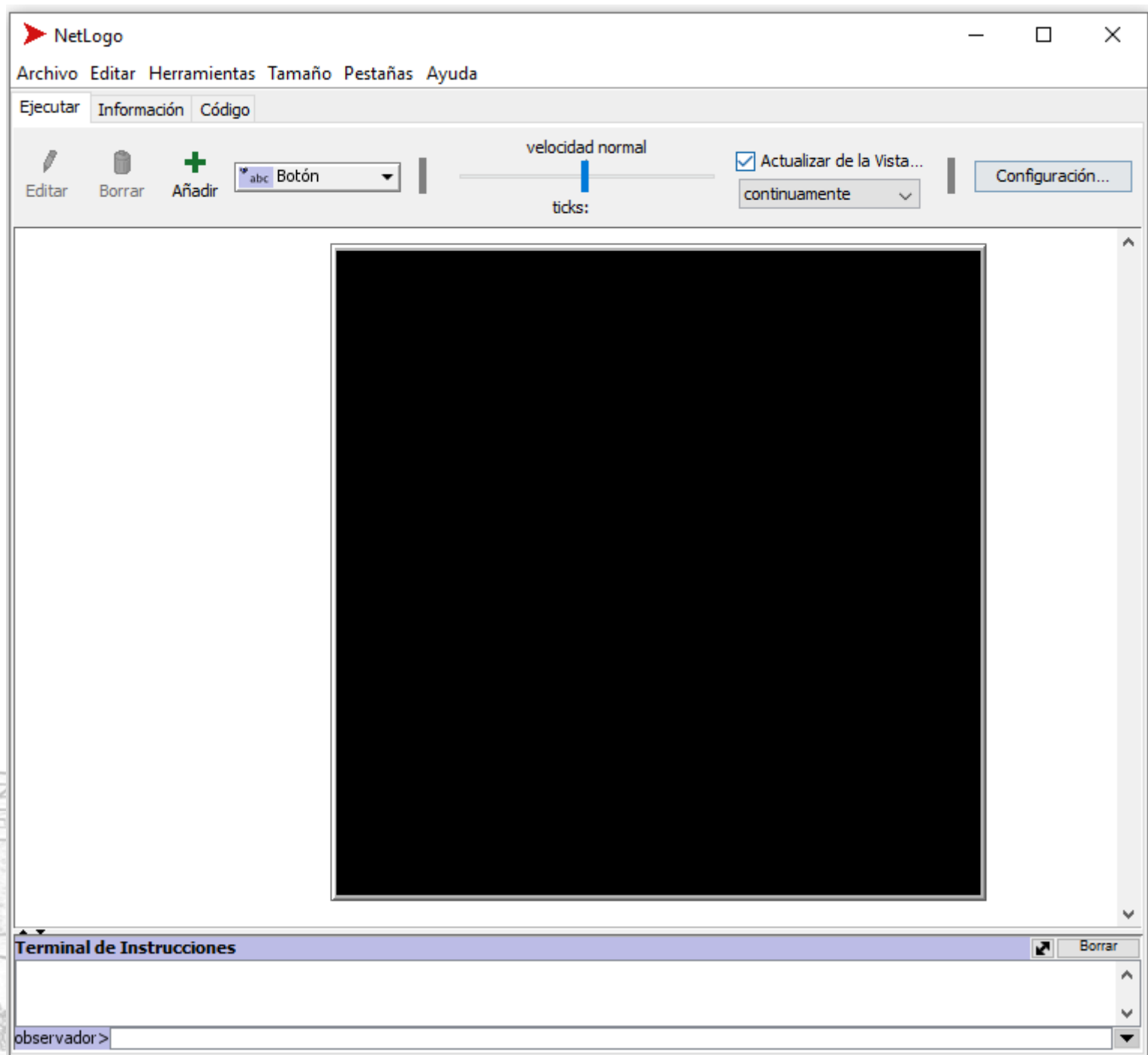
El mundo puede configurarse como un torus o un espacio limitado.



<https://en.wikipedia.org/wiki/File:Torus.png>

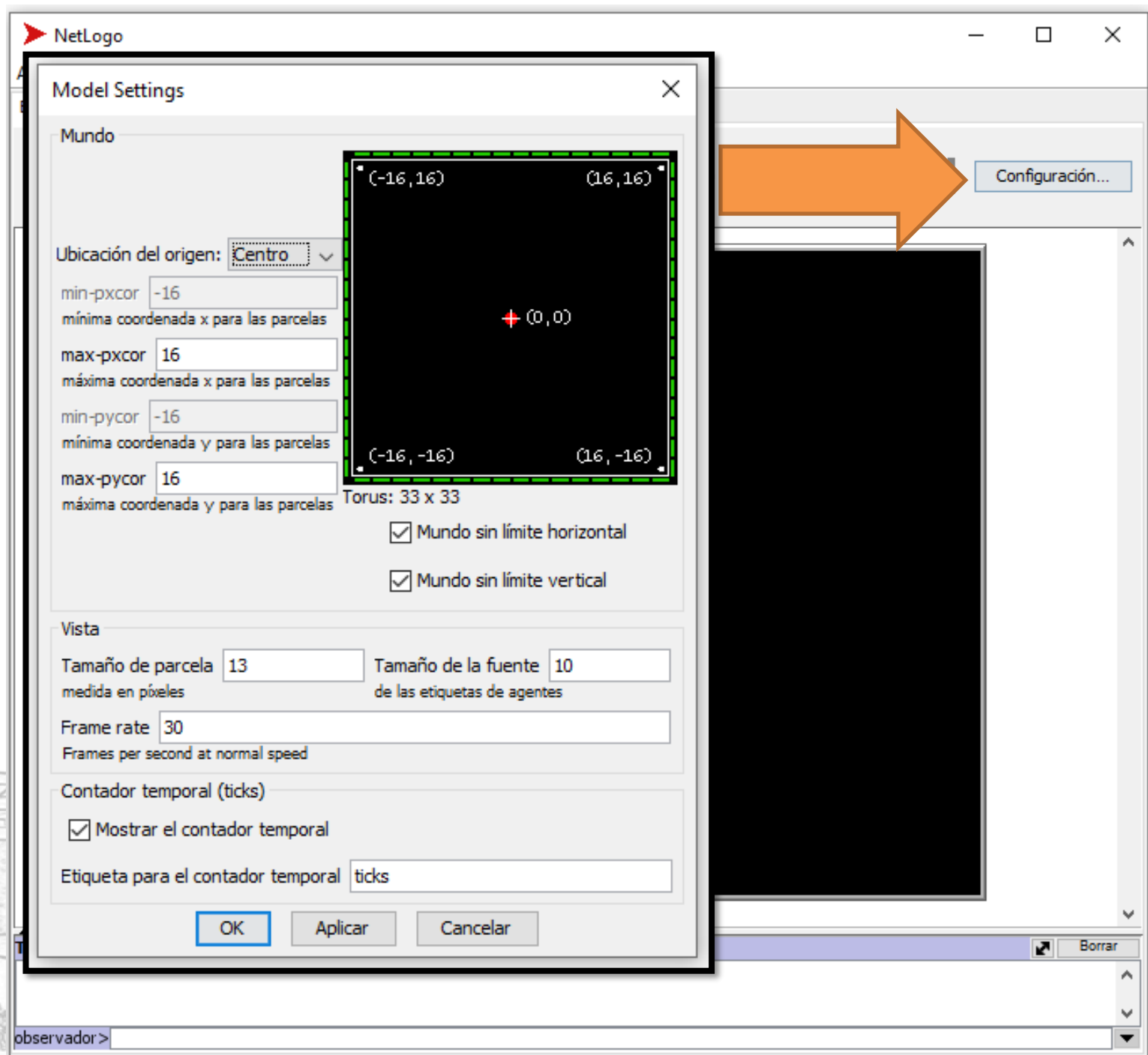


Bienvenido al mundo
de NetLogo.
Interfaz de inicio.



Configuración del mundo del modelo.

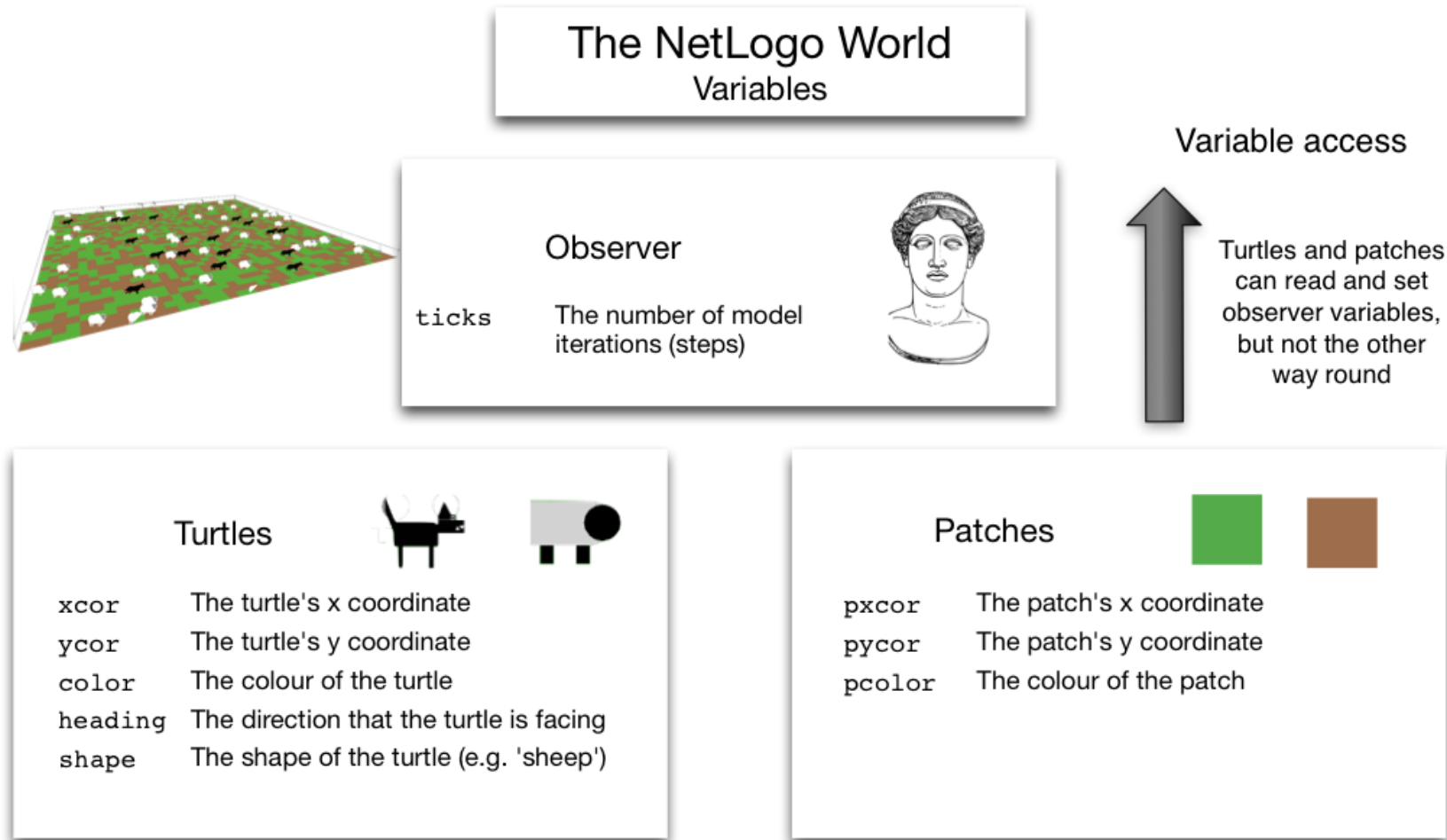
- Se puede definir el origen del modelo.
- Las coordenadas en su menor y mayor valor configuran la extensión del mundo.
- Se define el comportamiento del mundo, conectado o limitado.
- El tamaño del parche o parcela se puede configurar en conjunto con el conteo de refresco de visualización y el control del tiempo donde el estándar es ticks.



Controles y comandos

El control de variables en NetLogo permite almacenar datos, por ejemplo, se pueden configurar elementos visuales que ayudan al modelador a cambiar las acciones de los agentes mediante controles gráficos.

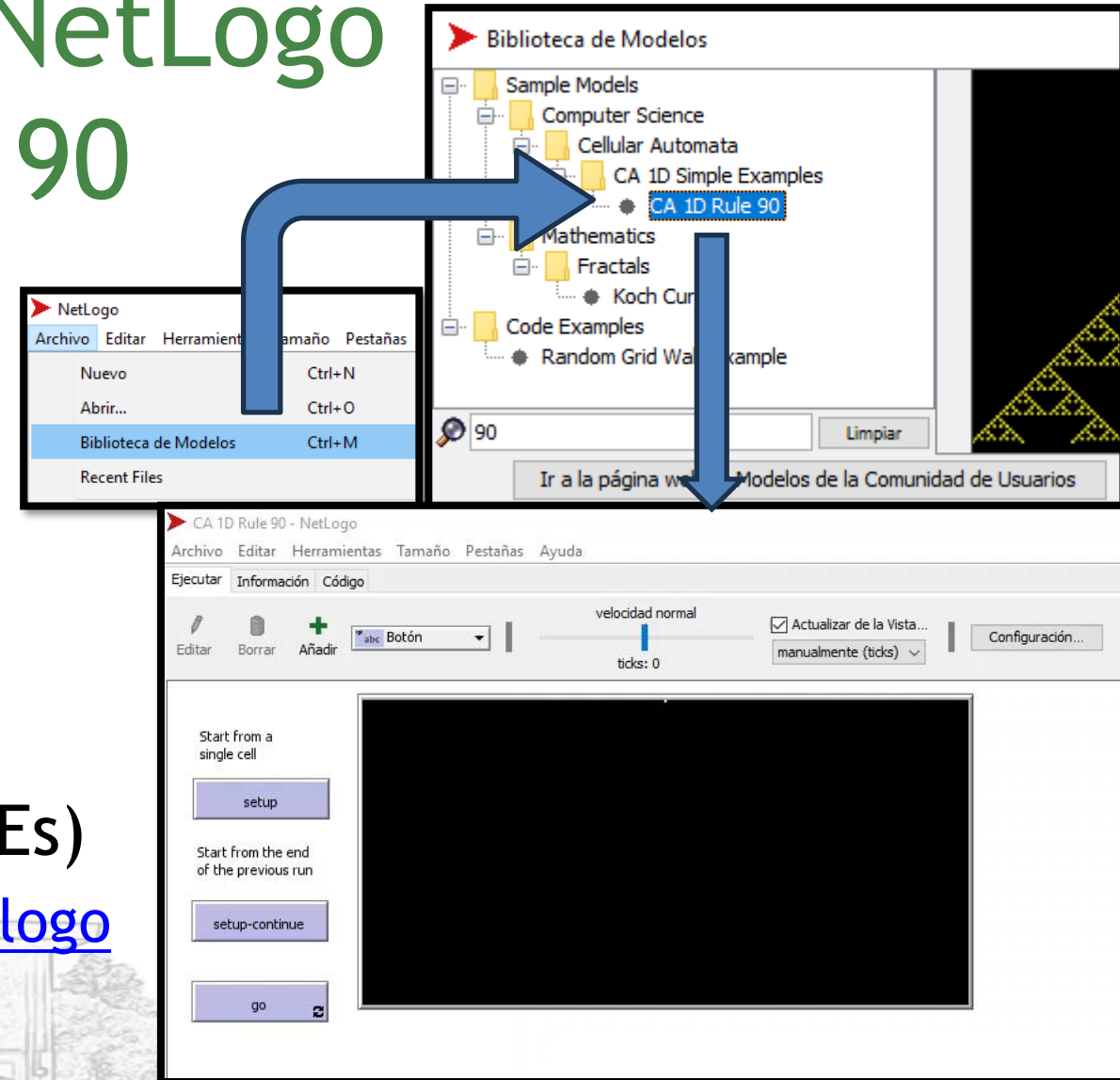
Los agentes también tienen sus propias variables y pueden ser modificadas o creadas en el modelo.



Ahora en NetLogo

Rule 90

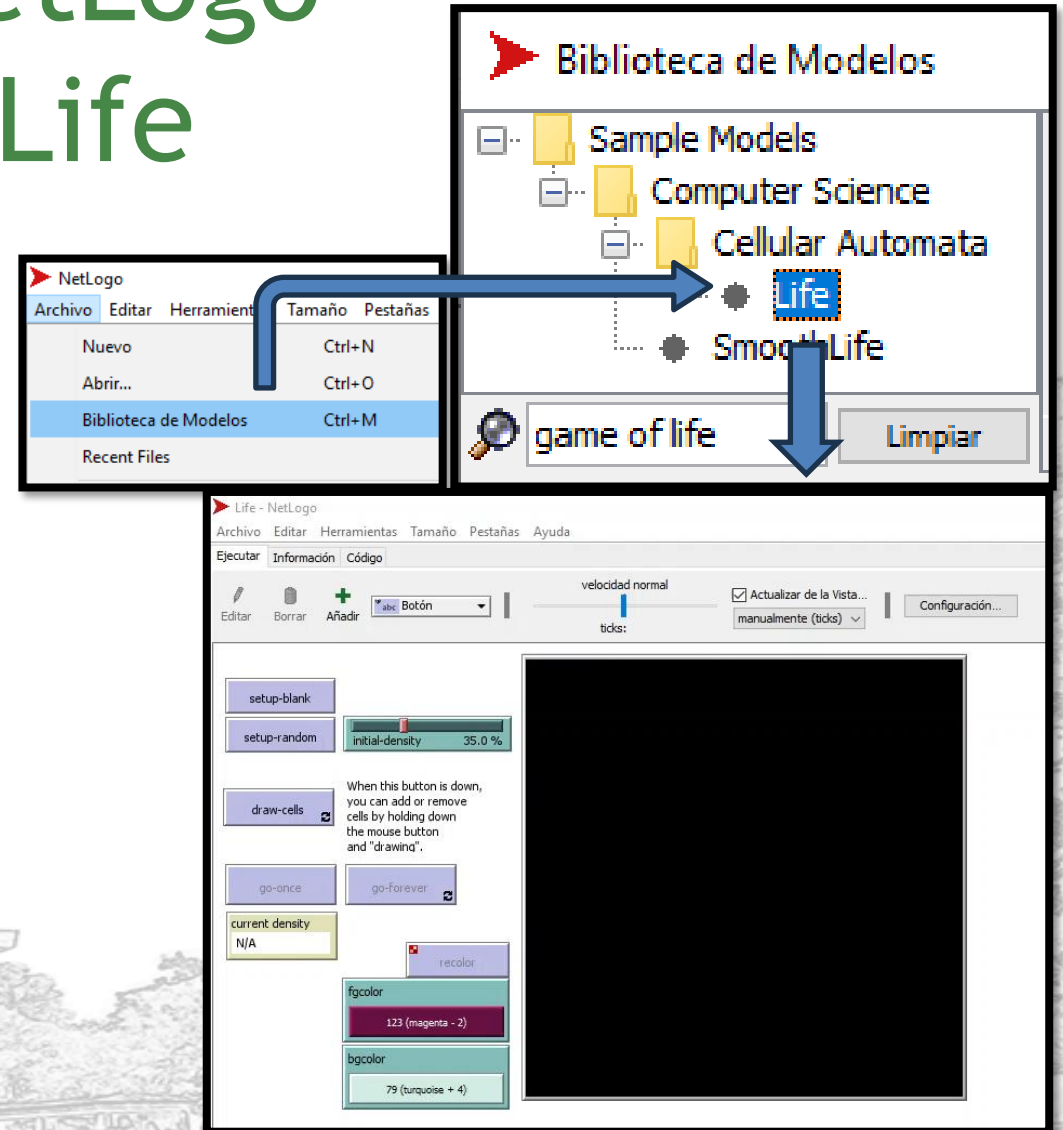
- NetLogo --> Archivo
 - Biblioteca de Modelos
 - En filtro ingresar el valor “90”
 - Clic en abrir
- Repositorio de GitHub: Comentado (Es)
 - [src/celular_automata/CA_1D_Rule90.nlogo](https://github.com/src/celular_automata/CA_1D_Rule90.nlogo)



Ahora en NetLogo

Game of Life

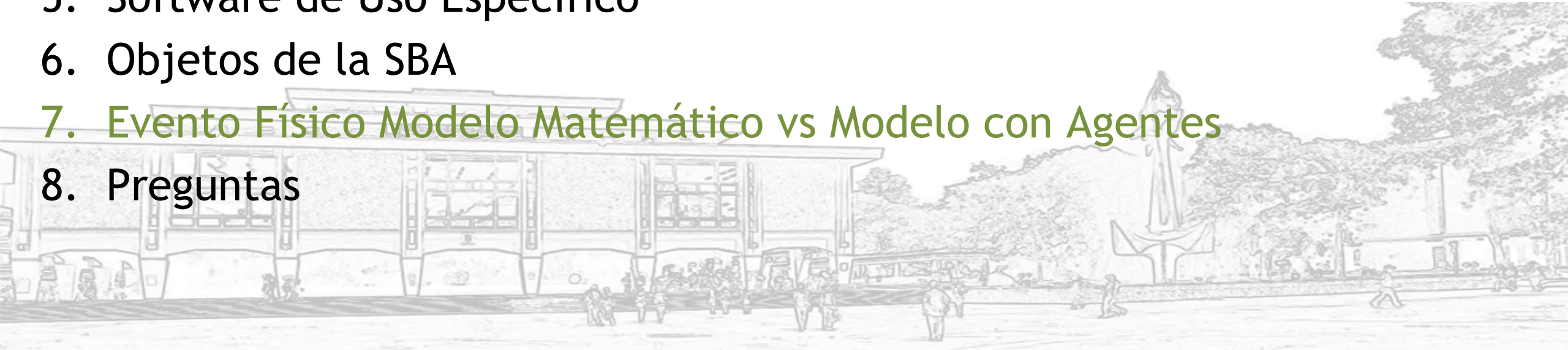
- NetLogo --> Archivo
 - Biblioteca de Modelos
 - En filtro ingresar el valor “game of life”
 - Clic en abrir
- Repositorio de GitHub: Comentado (Es)
 - [src/celular_automata/Life.nlogo](https://github.com/src/celular_automata/Life.nlogo)



Agenda



1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas



Evento Físico: Gota de agua

Ecuación de Onda en 2D

La simulación se fundamenta en la ecuación de onda de dos dimensiones:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

- $u(x, y, t)$ es la altura de la onda en el punto (x, y) y tiempo t
- c es la velocidad de propagación de la onda.
- $\frac{\partial^2 u}{\partial x^2}$, $\frac{\partial^2 u}{\partial y^2}$ representan la curvatura de la onda en las direcciones espaciales x e y .

La condición inicial se establece como una **perturbación gaussiana** en el centro de la malla (x_c, y_c) , simulando la caída de una gota en un fluido:

$$u(x, y, 0) = e^{\left(-\frac{(x-x_c)^2 + (y-y_c)^2}{2\sigma^2} \right)}$$

Esto significa que en el instante inicial $t = 0$, hay una perturbación en el centro de la malla, y su intensidad decrece exponencialmente con la distancia al centro.

Para la **velocidad**, se asume reposo:

$$(x, y, -\Delta t) = u(x, y, 0) \therefore \frac{\partial u}{\partial t} \approx \frac{u^0 - u^{-1}}{\Delta t} = 0$$

Bordes fijos: $u = 0$ en los límites (reflexión total de la onda).



Evento Físico: Gota de agua

Ecuación de Onda en 2D

Diferencias finitas

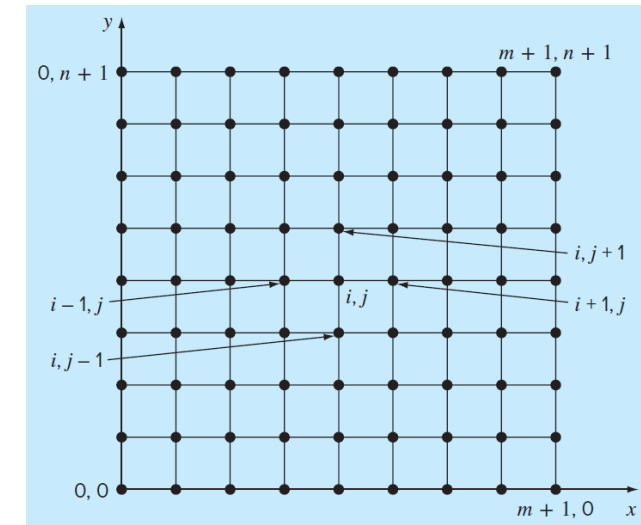
$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2}$$

$$\frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{\Delta y^2}$$

$$\frac{\partial^2 u}{\partial t^2} \approx \frac{u_{i,j}^{n+1} - 2u_{i,j}^n + u_{i,j}^{n-1}}{\Delta t^2}$$

$$u_{i,j}^{n+1} = 2u_{i,j}^n - u_{i,j}^{n-1} + \left(\frac{c\Delta t}{\Delta x} \right)^2 (u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n - 4u_{i,j}^n)$$



- $u(x, y, t)$ es la altura de la onda en el punto (x, y) y tiempo t
- c es la velocidad de propagación de la onda.

Fin

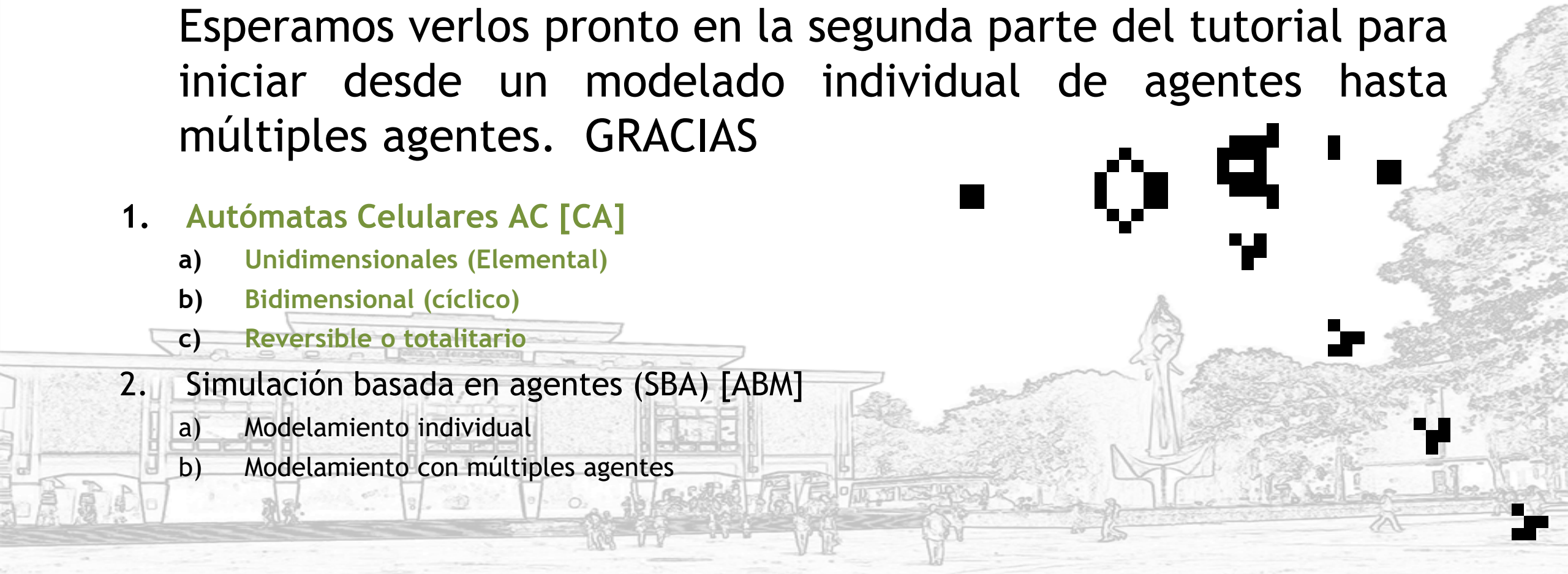
- Con el presente tutorial cubrimos el primer nivel (1). Esperamos verlos pronto en la segunda parte del tutorial para iniciar desde un modelado individual de agentes hasta múltiples agentes. GRACIAS

1. Autómatas Celulares AC [CA]

- a) Unidimensionales (Elemental)
- b) Bidimensional (cíclico)
- c) Reversible o totalitario

2. Simulación basada en agentes (SBA) [ABM]

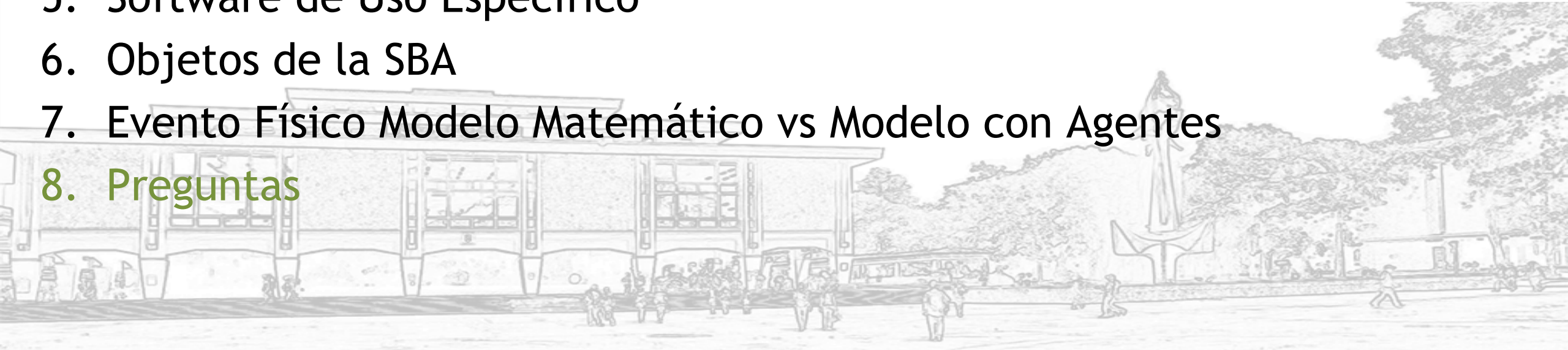
- a) Modelamiento individual
- b) Modelamiento con múltiples agentes



Agenda



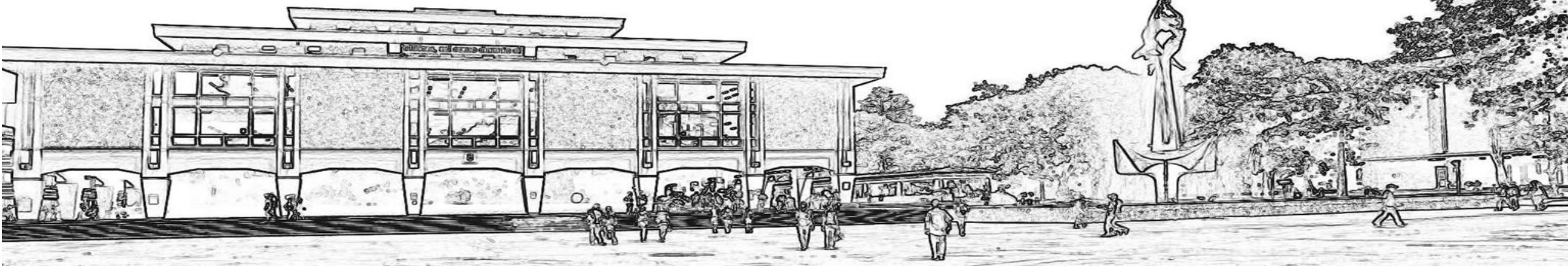
1. ¿Qué es Simulación?
2. Simulación basada en agentes (SBA)
3. Simulación basada en agentes - Historia
4. Autómatas Celulares
5. Software de Uso Específico
6. Objetos de la SBA
7. Evento Físico Modelo Matemático vs Modelo con Agentes
8. Preguntas





UNIVERSIDAD DE ANTIOQUIA

Facultad de Ingeniería



😊 Muchas gracias 😊

Tutorial de Simulación Basada en Agentes
Parte 1: Una aproximación con ejemplos
Espacio para preguntas

Yony Fernando Ceballos, Ph.D
Doctor en Ingeniería - Ingeniero de Sistemas
yony.cebалlos@udea.edu.co

Julián Andrés Castillo G. M.Sc.
Magister en Ingeniería - Ingeniero de Sistemas
Jandres.castillo@udea.edu.co

Departamento de Ingeniería Industrial