FIRMA INVITADA

ENTENDIENDO LA PROGRAMACIÓN VISUAL EN EL DESARROLLO DE VIDEOJUEGOS

Miguel Ángel Roque López
Universidad de Castilla-La Mancha

Este artículo sintetiza la experiencia en programación visual adquirida durante los últimos años por el grupo de investigación IDECA (investigación y desarrollo de contenidos audiovisuales) en el desarrollo de proyectos audiovisuales y videojuegos. Para ello, nos sumergimos en el proceso de creación de un videojuego a través del análisis de dos de los principales entornos de desarrollo basados en programación visual Blender y Unreal Engine. Por medio del estudio del funcionamiento de los Bloques de lógica utilizados en Blender, y del sistema nodular de Blueprints empleado por Unreal Engine, observaremos cómo la programación visual se ha convertido en el referente dentro del desarrollo de videojuegos. Analizaremos y compararemos las similitudes y diferencias entre ambos entornos, con el fin de comprender ante qué proyectos utilizar cada uno de ellos con más garantías de éxito.

This article synthesizes the experience in visual programming acquired during the last years by the research group IDECA (research and development of audiovisual content) in the development of audiovisual projects and videogames. To achieve this goal, we dive into the process of creating a video game through the analysis of two of the main development environments based on visual programming Blender and Unreal Engine. By a detailed study of the methodology and process used on Blender logic bricks and the Unreal Engine nodular system of Blueprints, we will define how visual scripting has become the reference in the development of videogames. To achieve this goal, we will analyze and compare the similarities and differences between both frameworks, in order to understand which framewok is better to guarantee success.

Palabras clave: Videojuego, Blender, Unreal, programación visual.
DOI: https://doi.org/10.4995/caa.2019.11336

El interés que suscitan los videojuegos hace que cada vez más personas quieran dedicarse a este sector en sus apartados más creativos o técnicos. Curiosamente, tal y como señala Casey O'Donnell en su libro *Developer's Dilemma: The Secret World of Videogame Creators* (2014), la labor del diseñador de videojuegos es un trabajo casi invisible y se encuentra eclipsada tras los grandes nombres de los editores y desarrolladores que los distribuyen. O'Donnell aborda en su obra cómo la capacidad de jugar con los sistemas subyacentes, tanto técnicos como conceptuales y sociales, es el núcleo de la práctica creativa y colaborativa, resultando fundamental para la nueva sociedad que está por venir (O'Donnell, 2014: 128). No obstante, el desarrollo de un videojuego es un proceso complejo que requiere de consistencia y un flujo de trabajo sólido. Esto ha propiciado un uso cada vez más extendido de metodologías de programación visual en el desarrollo de videojuegos. A continuación, analizaremos los distintos métodos de programación visual que se emplean en algunos de los principales "framework" para el desarrollo de videojuegos.

Un videojuego es una aplicación interactiva orientada al entretenimiento que permite simular experiencias en una pantalla o dispositivo electrónico. Los videojuegos se diferencian de otras formas de entretenimiento, como son las películas, en que deben ser interactivos y, por lo tanto, los usuarios deben relacionarse con el contenido. Para ello, es necesario utilizar un dispositivo de entrada mediante el cual se envían órdenes al dispositivo principal que reproduce el videojuego, y estas se ven reflejadas en una pantalla con una acción generada por el juego como respuesta.

Los videojuegos pueden ser muy distintos entre sí, tanto en complejidad como en calidad gráfica y en temática. De la misma forma que en el cine y la música, existe una larga y compleja lista de géneros y subgéneros. Los videojuegos forman parte de nuestro entretenimiento desde prácticamente el inicio de la informática. Al principio, eran analógicos y consistían más en juegos adaptados al formato de vídeo que a lo que actualmente conocemos como videojuego. Sin embargo, hoy en día son capaces de competir con las mejores producciones de cine y son plenamente digitales.

La metodología más simple para un diseñador que quiera desarrollar un videojuego es la programación visual, que consiste en la utilización de nodos y la capacidad de dichos nodos de interrelacionarse para definir comportamientos en el motor del juego. Estos nodos tienen como finalidad presentar un entorno de desarrollo visual que no requiera de conocimientos concretos en programación. A continuación, analizaremos los métodos, herramientas y posibilidades que ofrecen los principales entornos de trabajo para el desarrollo de videojuegos por medio de la denominada programación gráfica.

Cuando hablamos del motor de videojuego, nos referimos a una serie de rutinas de programación que permiten el diseño, la creación y la representación de una aplicación interactiva con fines lúdicos. Del mismo modo, existen motores de juegos que operan en diferentes plataformas, como consolas o sistemas operativos. La funcionalidad básica de un motor es proveer al videojuego de un sistema de render para los gráficos, que permita al mismo tiempo la detección de colisiones, sonidos, la ejecución de animaciones, proporcione unas rutinas de inteligencia artificial, facilite el acceso a redes, se encargue de la administración de memoria y

Fig. 1. Fotografía de *Bertie the Brain,* juego de vídeo presentado en la exposición nacional de Canadá de 1950 procedente de la revista *LIFE*.

que además permita hacer todo esto en tiempo real. El proceso de desarrollo de un videojuego puede variar notablemente en función del motor que se utilice, las posibilidades que ofrezca y la metodología que empleemos.

Los videojuegos recrean entornos y situaciones virtuales en los que el jugador puede controlar uno o varios personajes para alcanzar objetivos, por medio de las reglas y mecánicas de juego que cree el diseñador. Estas mecánicas son el elemento más complejo de desarrollar en un videojuego y, para su uso, deberemos interactuar con el juego mediante dispositivos de entrada y salida, siendo entre los primeros los más habituales dispositivos como un teclado, ratón o un gamepad y entre los segundos, dispositivos como el monitor y la televisión, entre otros.

Los videojuegos en otro tiempo considerados juguetes para niños y desarrollados por uno o dos programadores en los sótanos de sus casas han evolucionado hacia formatos que implican tanto a varias de las empresas y distribuidoras más importantes del mundo como a grupos de desarrolladores independientes. Su desarrollo involucra profesionales en campos tan diversos como diseño, informática, música o mercadotecnia. Su importancia es tal que su impacto y evolución se traduce incluso en marcos legislativos que tienen como objetivo dar respuesta a sus propios efectos (González, 2017).

En los videojuegos arte, ciencia y tecnología confluyen, en un conglomerado de habilidades y conocimientos procedentes de distintas disciplinas, desde ciencias formales hasta ciencias sociales que van más allá del típico proyecto de programa, e implican al mismo tiempo la creatividad y la imaginación. Un videojuego combina elementos de narración, música, animación, psicología, estrategia e incluso deporte en un producto, cuyo uso puede volverse tan competitivo como un deporte profesional (Álvarez, 2017). En la era que Dille y Zuur definieron como "primitiva" (2007: 38), cualquier persona podía aprender a programar un videojuego si se disponía de un ordenador personal y conocimientos de programación básicos. Producir un juego en aquel momento no requería de más de dos personas y un par de meses de trabajo. La competencia posterior unida al uso que la sociedad hace de los videojuegos ha derivado en unos equipos de trabajo que actualmente pueden superar el centenar de miembros en un solo videojuego comercial, y que los proyectos pueden durar hasta diez años como en el caso de *The Last Guardian* (Fumito Uheda, 2016).

Pero esto no ha impedido que a su vez se dé un fenómeno sorprendente y es que, aunque la cantidad de recursos y tiempo necesarios para la realización de un videojuego son cada vez mayores, la cantidad de videojuegos producidos no deja de aumentar. Como muestra señalar que plataformas de difusión y venta de videojuegos como Steam publicaron 7.762 juegos en 2017: una media de 21,26 juegos por día durante todo 2017 con un crecimiento del 38% con respecto a al año anterior (Márquez, 2018), esperándose incluso cifras mayores para los próximos años. La causa de este fenómeno seguramente se encuentre en la cada vez más habitual utilización de la denominada como "programación visual" en el desarrollo de videojuegos.

01

La programación visual

Sea cual sea la plataforma utilizada para el uso de la metodología de programación visual, siempre se basará en una propuesta similar centrada en el uso de funciones preconfiguradas que permiten definir procesos complejos. Dichas funciones reciben diferentes nombres en función del programa utilizado siendo algunos ejemplos los "Blueprints" utilizados en Unreal Engine, visual Scripting en Unity o los bloques de lógica usados en Blender Game Engine.

La principal cualidad de estas funciones es que no requieren de ningún tipo de conocimiento previo en programación para su uso, siendo en todo caso mucho más importante un pensamiento lógico y tener claro lo que se quiere lograr en lugar de un conocimiento profundo de programación. Esto los convierte en la forma idónea de aproximación al desarrollo de videojuegos, especialmente desde las áreas de conocimiento más alejadas de la ingeniería y la informática, como puedan ser el arte o las humanidades.

A continuación, analizaremos dos de las plataformas de desarrollo más interesantes que emplean métodos de programación visual para el desarrollo de videojuegos. Por un lado, Blender, software minoritario en el desarrollo de videojuegos empleado en producciones más independientes como *Yo frankie* (Barton, C. et al.; Blender Foundation, 2008) y Unreal Engine, plataforma mucho más implantada a nivel profesional y empleada en grandes éxitos recientes de la industria como *Gears of War* (Epic Games, 2006) o el reciente *Rime* (Tequila Works, 2017).

02

Blender y los bloques de lógica

En Blender denominamos lógica del juego a todos los comportamientos y eventos que suceden dentro del motor de render de videojuegos incluido dentro de Blender. Esta lógica puede ser definida utilizando el lenguaje Python, o por medio de alguno de sus 38 bloques de lógica. Estos bloques representan funciones predefinidas en Python con la finalidad de agilizar el desarrollo de videojuegos (Fig. 2) en Blender existen tres tipos básicos: sensores, controladores y actuadores (Roque, 2015: 48) y pueden ser definidos, ajustados y combinados para crear múltiples tipos distintos de interacciones en un videojuego (Fig. 2).

Los sensores son los bloques de lógica encargados de detectar eventos ("input") como una colisión, una pulsación de tecla o el movimiento del ratón. Por el contrario, los controladores se encargan de procesar los distintos eventos que se producen y determinar qué actuadores deben activarse para responder a una situación concreta. Finalmente, los actuadores son un tipo de bloque de salida y tienen por objetivo realizar acciones.

Los enlaces (Fig. 3) representan la dirección del flujo lógico entre los objetos. Los bloques lógicos sólo se pueden conectar de una forma determinada, siendo esta de sensores a controladores, y de controladores a actuadores. No obstante, los nodos de recepción de cada bloque lógico pueden recibir múltiples enlaces de entrada, obteniendo de esta forma conexiones de tipo uno a muchos, y propiciando de esta forma que una sola acción genere múltiples reacciones en cadena.

03

Unreal Engine 4

Este motor, empleado en algunos de los videojuegos más importantes de la actualidad como *Playerunknown's Battlegrounds* (Bluehole Studio, PUBG Corporation, 2017) o *Fortnite* (Epic Games, 2017) se distribuye de forma pseudo gratuita desde 2015. Aunque en un primer momento estaba orientada a desarrolladores, la flexibilidad de su licencia propició que cualquier usuario pudiese obtenerlo y crear sus propios videojuegos o escenarios, y como consecuencia ha obtenido una gran popularidad dentro de la industria. El único requisito es que los desarrolladores que usen Unreal Engine como motor en sus videojuegos deben abonar a Epic Games, en concepto de desarrollo del programa, un porcentaje de las ganancias que genere. Con este motor gráfico podremos crear

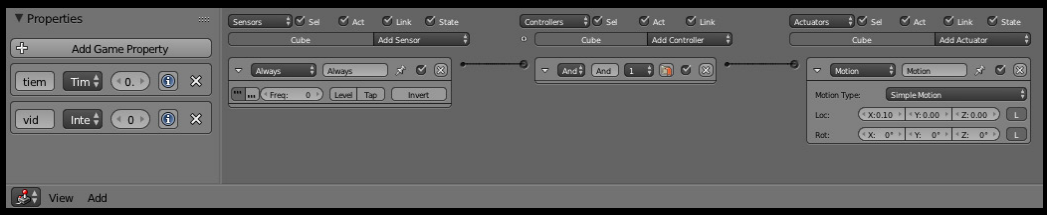

Fig. 2. Ejemplo en Blender de una interacción simple, en este caso un movimiento continuo.

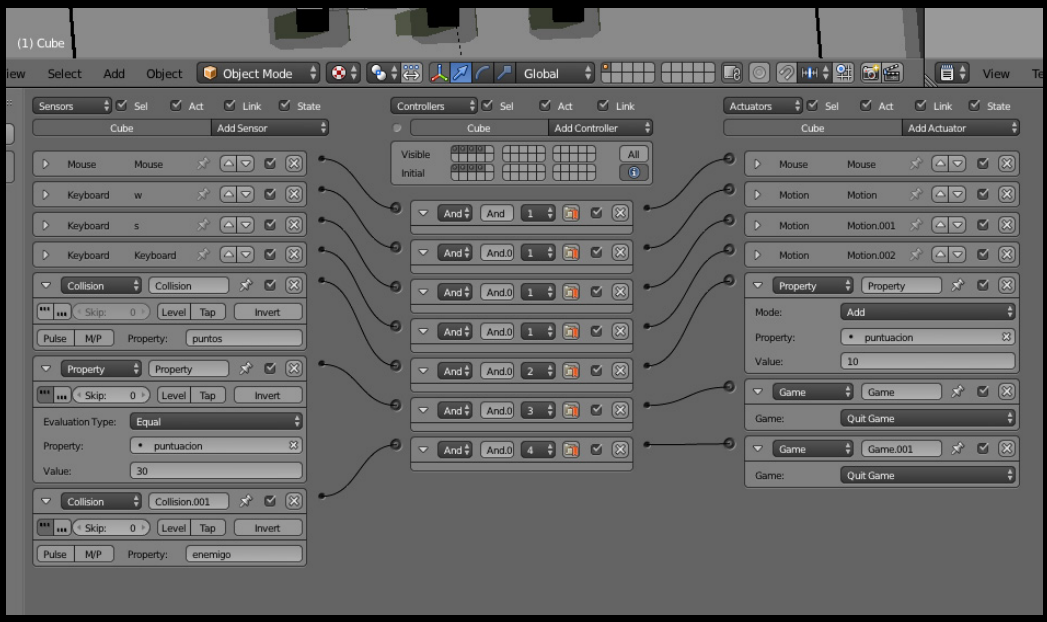

Fig. 3. Ejemplo de conexión estándar entre bloques lógicos.

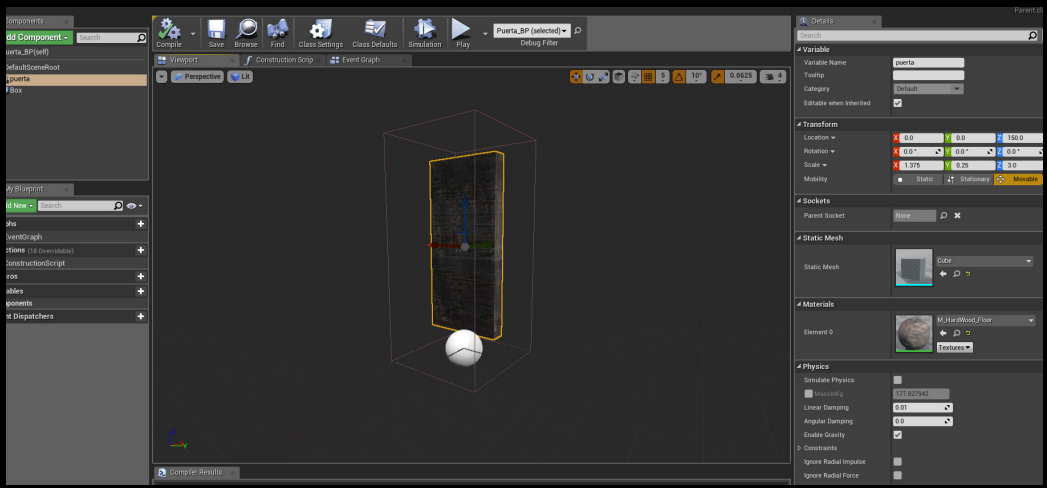

Fig. 4. Vista del contenido –Mesh– de un "Prefab" o "Blueprint", en este caso una puerta.

videojuegos con una gran variedad de niveles de complejidad desde plataformas 2D, juegos AAA o incluso videojuegos para móvil o realidad virtual.

En Unreal Engine, cuando hablamos de lógica del juego, nos referimos a los comportamientos y eventos que suceden dentro del motor durante el juego. Esta lógica puede ser creada utilizando el lenguaje de programación C++ o por medio de "Blueprints". Los "Blueprints" representan plantillas o prefabs de objetos (Fig. 4) similares a los Game kits empleados en otras plataformas (Roque, 2018: 143) y funciones predefinidas en C++, con la finalidad de agilizar el desarrollo de videojuegos en Unreal (Fig. 4).

Unreal tiene cientos de los denominados "Blueprints", que consisten en un sistema de desarrollo simplificado basado en el concepto de utilizar una interfaz basada en nodos (Fig. 5) para crear interacciones entre los elementos del juego. Debido a esto y a su capacidad de replicar comportamientos en objetos diferentes, entre los usuarios de Unreal Engine es frecuente entender los "Blueprints" como plantillas de comportamiento. Este sistema es muy flexible y potente, ya que brinda a los diseñadores la capacidad de utilizar virtualmente toda una gama de conceptos y herramientas que generalmente solo están disponibles para los programadores (Fig. 5).

Al contrario que en Blender, donde los bloques de lógica están vinculados a los objetos, en Unreal Engine los nodos que utilizamos en los "Blueprints" pueden aplicarse a objetos o al nivel activo. En el primero de estos casos, se denominarán "Blueprint Class", una clase de "Blueprint" ideal para hacer objetos interactivos como puertas, interruptores, artículos recolectables y elementos del entorno. Mientras, los denominados "Blueprint Level" se encuentran vinculados al nivel en ejecución en ese momento y se suelen utilizar para interactuar con objetos y otros "Blueprint" de forma personalizada dentro de ese nivel o entorno. Se emplean principalmente en situaciones como la creación de cinemáticas y administrar eventos concretos como el cambio de nivel, los puntos de control y guardado, o en otros procesos relacionados con la interfaz de usuario y la optimización del videojuego.

Conclusiones

Como hemos podido observar a lo largo de este artículo, tanto Blender como Unreal engine incorporan importantes herramientas de la programación visual aplicadas al desarrollo de videojuegos. Podemos concluir que desde un punto de vista metodológico no existe una gran diferencia entre ambas plataformas, limitándose sus diferencias al alcance de las funciones predefinidas en cada motor y a la complejidad de las mismas. En ambos casos, hemos podido observar que si bien el desarrollo de videojuegos está migrando hacia la programación visual, todos los flujos de que emplean esta metodología se apoyan en lenguajes de programación como Python o C++ (Thorn, 2014: 8).

Otra de las observaciones que podemos hacer es que, por su simplicidad, resulta más interesante y rápido desarrollar por medio de bloques de lógica y Blender aquellos proyectos que requieran de un desarrollo básico de funcionalidades, y cuando debamos ceñirnos a planificaciones temporales muy ajustadas. Por el contrario, debido a la gran capacidad de interacción existente entre los diferentes nodos de Unreal Engine, si disponemos de tiempo suficiente o nos enfrentamos a un proyecto que requiera del desarrollo de funcionalidades más avanzadas o específicas, observaremos cómo el desarrollo del videojuego será mucho más ágil en Unreal Engine.

Finalmente, podemos deducir a la vista de estos datos que, aunque todavía no se encuentre a nivel técnico al nivel de soluciones de carácter privativo como Unreal Engine, el hecho de que Blender se distribuye bajo licencia GPL y que permite la explotación comercial sin limitaciones

(Blender Foundation, 2000) de cualquier proyecto, al contrario que los motores comerciales, lo convierten en una alternativa más que viable e interesante para el desarrollo de videojuegos, especialmente en el caso de pequeños estudios que cuenten con presupuestos muy ajustados. Por el contrario, la actual licencia combinada con las capacidades técnicas avanzadas que ofrece Unreal Engine lo convierten en la mejor plataforma actual para el desarrollo de un videojuego empleando programación visual, siendo su sistema basado en "Blueprints" la forma más potente y efectiva de desarrollar un videojuego.

Referencias bibliográficas

ÁLVAREZ Cedena, José, 2017. "Así serán las retransmisiones de Esports de la próxima generación", en *El País*, 12 de septiembre de 2017 (https://elfuturoesapasionante.elpais.com/eve-la-proxima-revolucion-disfrutar-los-esports-major-league-gaming-mlg/ [acceso: junio, 2018]).

BLENDER FOUNDATION. 2000. "Blender Licence", en *Blender* (http://www.blender.org/about/license/ [acceso: junio, 2018]).

DILLE, Flint, ZUUR, John, 2007. "The Ultimate Guide to Video Game Writing and Design", Nueva York: Lone Eagle Publishing.

GONZÁLEZ, Sergio, 2017. "La polémica lotería de los videojuegos", en *El País*, 28 de noviembre de 2017 (https://elpais.com/tecnologia/2017/11/28/actualidad/1511880911_023202.html [acceso: junio, 2018]).

MÁRQUEZ, Raúl, 2018. "El número de lanzamientos en Steam en 2017 es casi el doble de lo lanzado en 2016", en *Vida Extra*, 10 de enero de 2018 (https://www.vidaextra.com/pc/el-numero-de-lanzamientos-en-steam-en-2017-es-casi-el-doble-de-lo-lanzado-en-2016-y-es-insostenible [acceso: abril, 2018]).

O'DONNELL, Casey, 2014. *Developer's Dilemma: The Secret World of Videogame Creators,* Cambridge: The MIT Press.

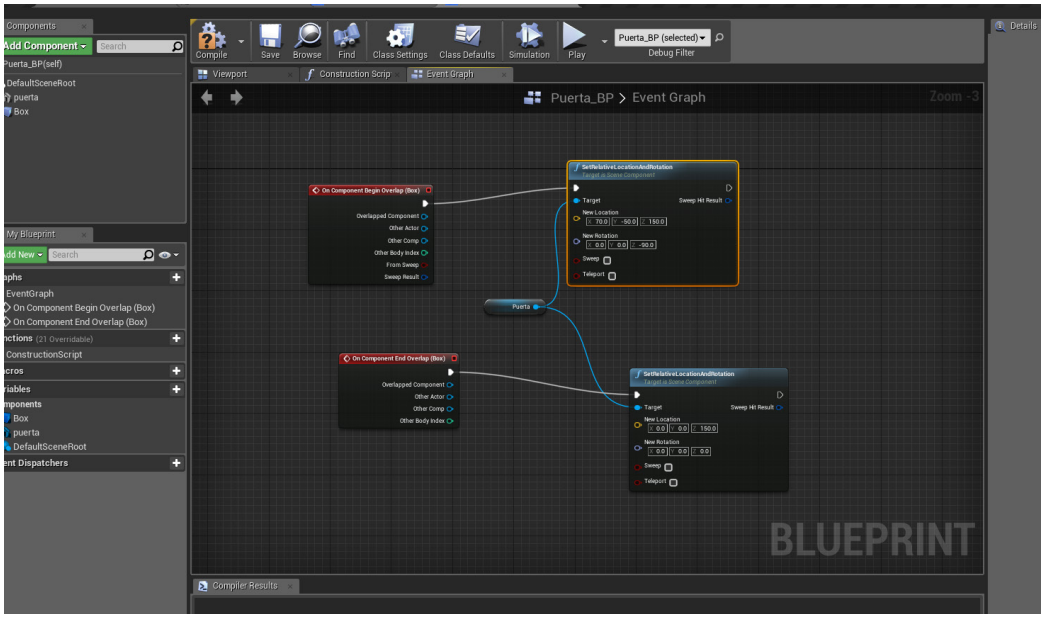

Fig. 5. Vista de los nodos asociados a un "Blueprint", abriendo y cerrando la puerta al acercarnos.

ROQUE, Miguel Ángel, 2015. "Blender: Videogame Development through logic bricks and Python", en Videojuegos: *Desarrollo e industria creativa*, nº3, Madrid: ESNE, pp. 45-56.

ROQUE, Miguel Ángel. 2018. "Desarrollando Porto: un videojuego sobre patrimonio cultural", en *Con A de animación* , nº 8, Valencia: Editorial UPV, Nau Llibres, pp. 136-148.

THORN, Alan, 2014. *Practical Game Development with Unity and Blender*, Boston: Cengage.Learning.

UHEDA, Fumito, 2016. *The Last Guardian*, Japón: Sony Interactive Entertainment.

Biografía

Miguel Ángel Roque es profesor e investigador de la Universidad de Castilla-La Mancha desde el año 2006, cuando entra a formar parte del grupo de investigación IDECA. Doctorado en Bellas Artes con sobresaliente Cum Laude y licenciado en Bellas Artes en la Universidad de Castilla-La Mancha, recibió también clase de diseño en la Accademia di Belle Arti di Bologna (Italia).

E-mail
miguelangel.roque@uclm.es