

## **Estudio y recomendación de Computadoras de Abordo para Cubesat\***

**Roberto Mayer**

**Federico D'Angiolo**

**Guillermo Caporaletti**

**Federico Collado**

**David Contreras**

**Héctor Perez**

**Matías Loiseau**

**Noelia Aparicio**

**Hernán Rodríguez Verahernan**

**Agustina Pelozo\*\***

### **Resumen**

El desarrollo de Cubesats requiere de un gran estudio y desarrollo en varios aspectos técnicos en el área de hardware, software y firmware. En particular, dado el convenio firmado entre la UNDAV y el consorcio de universidades, mediante este documento realizaremos un estudio y recomendación de las posibles Computadoras de Abordo (OBC, On-Board Computer, en inglés) para IoT (Internet of Things, en inglés, o Internet de las Cosas), que se pueden utilizar en el Cubesat. Este análisis tiene en cuenta, además, el firmware y el software necesarios para la gestión de datos con IoT, dentro de cada computadora.

**Palabras clave.** OBC. IoT. Sistemas Embebidos. Software. Firmware.

---

\* Recibido: 14-12-21. Aceptado: 12-09-22.

\*\* Ingeniería en Informática, Departamento de Tecnología y Administración, Universidad Nacional de Avellaneda. Correo electrónico: fdangiolo@undav.edu.ar

## **Abstract**

The development of the Cubesats need a big study and analysis in some themes about hardware, software and firmware. In particular, due to the signed agreement between the UNDAV and the consortium of universities, trough this document we make an study and recomendation about the posibles OBC (On-Board Computer) for IoT (Internet of Things), with use un Cubesats. This analysis have in account the firmware and the software needed for the use of the data in a eah computer.

**Keywords.** OBC. IoT. Embedded Systems. Software. Firmware.

## **Resumo**

O desenvolvimento de Cubesats exige um grande estudo e desenvolvimento em vários aspectos técnicos na área de hardware, software e firmware. Em particular, atendendo ao acordo assinado entre a UNDAV e o consórcio de universidades, através deste documento iremos realizar um estudo e recomendação de possível OBC (On-Board Computer) para IoT (Internet of Things, Internet of Things), que pode ser usado no Cubesat. Essa análise também leva em consideração o firmware e o software necessários para o gerenciamento de dados com IoT, dentro de cada computador.

**Palavras-chave:** OBC. IoT. Sistemas Embarcados. Programas. firmware

## **Introducción**

En los últimos años, el costo que requiere el diseño y construcción de Cubesats ha disminuido como consecuencia del aumento de proyectos en el área, la estandarización de sus partes y la atenuación en el precio de algunos de sus componentes (Zufelt, 2018). Esto trae como beneficio, una mayor exploración en el área de investigación y desarrollo, sobre todo en el área de IoT (*Internet of Things*, en inglés, *o* Internet de las Cosas), la cual es un área que se viene multiplicando en los últimos años (Alen Space, 2021). Por esta razón, el objetivo conjunto que se viene dando entre distintas entidades, siendo la Universidad de Palermo (UP) la originaria de esta idea, es promover un laboratorio espacial destinado a la provisión de servicios de Internet de las Cosas.

En particular, desde la carrera de ingeniería en informática se estará estudiando a las distintas computadoras de abordo que ofrece el mercado de Cubesats<sup>1</sup> para poder embeber herramientas de gestión de datos para el Internet de las Cosas (IoT), tales como *brokers* y base de datos. Para esto, resulta necesario conocer y estudiar a fondo la arquitectura y la capacidad de cada una de las OBC.

Dado que el protocolo utilizado para los servicios de IoT en este Cubesat resulta ser MQTT (Message Queuing Telemetry Transport, en inglés) es importante estudiar cuál broker se puede utilizar dado que el mismo estará embebido en una OBC sobre el mismo satélite. Además, dado que el Cubesat pierde comunicación con las estaciones terrestres durante una parte importante del trayecto de su órbita, resulta necesario almacenar los datos. Por esta última razón, resulta necesario entonces que dentro de la OBC exista una base de datos. Tanto el broker como la base de datos deben consumir la menor cantidad de recursos de la Computadora de Abordo (OBC, en inglés), dado que sobre el satélite existen otras computadoras de abordo como por ejemplo la de telemetría, que también tienen su consumo propio. La suma de todos estos consumos debe ser la menor posible pues, de lo contrario, la potencia desarrollada por el satélite será excesiva y se perderá eficiencia en la misión.

Todos estos análisis serán estudiados mediante un equipo de estudiantes, docentes e investigadores de la Carrera Ingeniería en Informática de la UNDAV.

El estudio que se comenta a continuación se estructura de la siguiente forma: en la sección 1 se describen los fundamentos de Cubesat para luego. En la sección 2 se comenta el hardware y el software adecuado de una OBC que se incorpora a un Cubesat. A continuación, en la sección 3, se comenta la programación que se puede desarrollar en cada computadora de abordo, teniendo en cuenta los conceptos de la sección 2. Luego, en la sección 4, se describe una introducción a IoT en el ámbito de un Cubesat para describir la fundamentación del uso de bases de datos y los posibles brokers para MQTT. Por último, en la sección 4, en base a los estudios comentados en las secciones anteriores, se analizan las distintas OBC que ofrece el mercado y la recomendación para este proyecto.

---

<sup>1</sup> Cubesat es un estandar de diseño de nanosatélites.

## **Sección 1. Fundamentos de Cubesat**

Previamente a describir el análisis y recomendación, resulta importante detallar las nociones básicas sobre Cubesats y el Internet de las Cosas. Un Cubesat es un nanosatélite compuesto por unidades básicas de unos 10x10x10 cm, con una masa de aproximadamente entre 1 y 1,33 kg. Esta unidad es conocida como 1U. El mismo puede ser utilizado para emplearse en diversos objetivos tales como captura de imágenes, medición de propiedades físicas en la ionósfera hasta monitoreo de recursos naturales por detección remota desde el espacio (Alen Space, 2021).

El objetivo principal de éstos es promover la navegación espacial, tanto para aquellos países ajenos a toda la carrera espacial de las potencias mundiales como para investigadores y profesionales del área que buscan diseñar, construir, lanzar y estudiar sus primeros nanosatélites.

Por otro lado, teniendo en cuenta el concepto del Internet de las Cosas (en inglés, IoT), se puede decir que éste se refiere a la conexión de dispositivos físicos que utilizan software embebido y/o sensores junto con internet. De aquí surge que cada uno de estos dispositivos tenga una dirección mediante la cual se pueda acceder a cada uno de ellos para establecer la comunicación. Algunos ejemplos cotidianos pueden ser el de las computadoras, celulares, tablets, televisores, impresoras multifunción, los cuales permiten enviar y recibir información utilizando los propios recursos.

Ante estos conceptos, es entonces importante generar una mayor expansión en IoT mediante nano satélites y, para esto, resulta interesante estudiar todos los elementos de esta infraestructura, las cuales se comentan en las siguientes secciones.

## **Sección 2. Análisis de hardware y software de una OBC**

En el mercado de las Computadoras de Abordo (OBC, en inglés), existe una gran cantidad de dispositivos con diferentes capacidades para distintos objetivos. Por esta razón, resulta importante generar los requisitos mínimos con los que debería cumplir la OBC para IoT. En particular, dado que estas generalmente contienen microcontroladores y módulos FPGA, a continuación, se exponen una serie de requisitos básicos con los que se debería cumplir para poder embeberlos en un Cubesat:

- **hardware**

El hardware de una OBC debe cumplir con los siguientes requisitos:

- a) Exposición a radiación (Kelapure, 2020).
- b) Disponibilidad del módulo de detección y corrección de errores.
- d) Potencia de procesamiento y memoria adecuadas para el mínimo consumo de recursos.
- e) Periféricos como SPI, CAN o I2C para comunicaciones con otros dispositivos.

- **software**

La arquitectura del software debe:

- a) Tener en cuenta un comportamiento en tiempo real (Kelapure, 2020).
- b) Proporcionar una funcionalidad estable para toda la misión (Kelapure, 2020).
- c) Ser modular con interfaces estándares (Kelapure, 2020).

- **programación en alto nivel**

Observando el análisis de las distintas arquitecturas de OBC que se obtienen en el mercado, se puede ver que muchas de ellas ofrecen gestionarlas mediante Bare Metal, RTOS y/o Linux Embebido. Por esta razón, a continuación se realiza un breve análisis de cada una de estas opciones, para poder tener un mejor panorama de cuál OBC elegir.

- **Bare Metal**

Cuando se trata de Bare Metal, la programación se centra en un bucle donde el control de los periféricos conectados se realiza mediante sondeo o interrupciones (Kelapure, 2020). Esto puede ser de gran utilidad cuando se necesita controlar el estado de algún periférico o incluso trabajar con algún tipo de comunicación, pero tiene la desventaja de no ser escalable.

- **RTOS**

Siguiendo con el análisis, RTOS (Sistemas Operativos en Tiempo Real o, en inglés, Real Time Operating System) se puede programar y ejecutar una gran cantidad de tareas para alcanzar un uso óptimo de la CPU (Unidad Central de Procesamiento o, en inglés, Central Processing Unit), y la memoria. Este tipo de programación tiene ciertas ventajas frente a *Bare Metal* en cuanto a la gestión de la CPU, lo cual es un punto importante a la hora de

pensar en el flujo de recursos en un Cubesat, sobre todo por la gestión de la comunicación entre las distintas tareas (Kelapure, 2020).

- **sistemas operativos embebidos**

Cuando se trata de sistemas operativos embebidos, la elección ampliamente elegida suele ser Linux (Linux Embebido) debido a la calidad, disponibilidad de código, amplio soporte de hardware, implementaciones de protocolos de comunicación, interfaces de programación de aplicaciones y herramientas de desarrollo disponibles. Además, una vez ya implementado en un nanosatélite, tiene mayor versatilidad ante futuras mejoras o nuevos desarrollos. Nuestro análisis está destinado a su utilización en computadoras de abordo para un Cubesat para IoT. (Leppinen, 2017).

Un ejemplo de lo comentado anteriormente se da en el trabajo “Design of Attitude Control Systems for CubeSat-Class Nanosatellite” (Li, J., 2013), donde se presenta un diseño de sistema de control de actitud satelital que utiliza hardware y software de bajo costo para un CubeSat 1U. El mercado de las OBC para Cubesat ofrece una gran cantidad de computadoras con esta opción y, en particular, dado que sobre la OBC bajo estudio debe haber un Broker para IoT y una Base de Datos operando, se propone trabajar con computadoras que contengan Linux Embebido como una mejor opción frente a Bare Metal y RTOS. Además, dada la gran cantidad de trabajos donde se aplican estos sistemas (Li, J, 2013), se refuerza que la mejor opción para el Cubesat bajo estudio es usar Linux Embebido.

En base a esta recomendación, a continuación, se fundamentan las sugerencias en cuanto a brokers y bases de datos.

### **Sección 3. IoT en Cubesat**

Retomando la idea comentada anteriormente, se puede definir al Internet de las Cosas (IoT) como una red física de dispositivos integrados con el propósito de conectarse, recopilar e intercambiar datos. Esta tiene la ventaja de poder ofrecer control sobre la telemetría de dispositivos en tiempo real. Dada la flexibilidad que tiene el protocolo MQTT (Message Queuing Telemetry Transport, en inglés) para mensajes de poco tamaño, resulta importante explorar los distintos *brokers* con sus respectivos rendimientos, tarea que desarrolla en la sección 3.1.

Para lograr que el satélite pueda comunicarse con los distintos equipos, es necesario comprender que la infraestructura consta principalmente de dos partes: la primera es la conexión satélite-estación en tierra; y la segunda, la conexión entre la estación en tierra y los softwares para tareas operativas.

Abordando la primera parte de la conexión, debemos considerar que los Cubesat deben tener la capacidad de poder conectarse con distintos dispositivos tanto en tierra como en órbita. Actualmente según las tecnologías existentes, los protocolos más utilizados para cumplir con dicha tarea involucran la transmisión de datos utilizando distintos protocolos de ondas de radio como Ultra Alta Frecuencia, Muy Alta Frecuencia o Banda-S (UHF/VHF/S-Band, en inglés). Estos permiten establecer conexiones de largas distancias utilizándose como la capa de transporte. Si bien la misma es capaz de vincular dos dispositivos mediante una conexión *half-duplex*, la gestión del envío de datos y la coordinación entre dispositivos se da en una capa superior utilizando los protocolos de comunicación establecidos por el Comité Consultivo para Sistema de Datos en el Espacio (CCSDS, en inglés).

Durante esta primera etapa de la comunicación, las estaciones terrestres estiman la posición donde puede llegar a estar el satélite para enviar ondas de radio que intenten comunicarse con él y logran establecer conexión. Los Cubesat solo tienen la capacidad de transmitir datos con una tasa que tiene un rango comprendido entre los 9600 bps y los 3Mbps durante aproximadamente 7 minutos (Zufelt, 2018). A esto, hay que sumarle que el Cubesat pasa por nuestro país, aproximadamente una vez al día. Estos datos fundamentan la utilización de una base de datos dentro de la OBC para guardar la información que se va recibiendo. Además de esto puede darse la posibilidad que, en el lapso de los 7 minutos, por distintos motivos, el satélite no logre comunicarse, con lo cual, resulta importante que los datos se puedan almacenar durante otras 24 horas (aproximadamente).

La segunda parte está encargada de realizar la comunicación entre los dispositivos en tierra y los equipos dedicados al uso y mantenimiento del satélite. Los dispositivos ubicados en esta sección se conectan entre sí utilizando el protocolo TCP/IP de internet y MQTT. Este último se utiliza debido a las múltiples ventajas que este posee. El mismo es liviano, rápido, óptimo en el manejo de datos y energéticamente eficiente por la forma en que trabaja. Su tecnología funciona utilizando un sistema de suscripciones entre

dispositivos, algo que le brinda flexibilidad al momento de conectarse con las estaciones terrestres y otros dispositivos. Otro punto a favor es que el paquete de datos utilizado por el protocolo MQTT permite transferir perfectamente los paquetes del protocolo CCSDS.

Es importante remarcar que actualmente está surgiendo una propuesta para integrar distintas estaciones de tierra con los diversos equipos de trabajo. El propósito es que todos puedan acceder a cualquier estación de tierra que sea compatible con su Cubesat para poder transmitir datos. Esto permite hacer mejor uso de las estaciones en tierra organizándolas para lograr una mayor cantidad de conexiones con los satélites.

Al analizar el modo en que funciona IoT en el campo de los Cubesat, se desprenden dos pautas que son convenientes estudiar para poder realizar las consideraciones de recomendación para la OBC.

La primera es que para poder utilizar correctamente la infraestructura de comunicación entre tierra y Cubesat, es necesario poder contar un software dentro de la Computadora de Abordo (On Board Computer, en inglés) que tenga la capacidad de poder almacenar datos. Estos datos guardados podrán ser enviados a través de paquetes siempre que se los solicite en los distintos puntos de conexión. En base esto, se considera importante contar con una base de datos dentro de la OBC que permita almacenar dichos datos en memoria durante el tiempo que sea necesario.

Como segundo punto, la infraestructura de comunicación en la Tierra no es una limitante para la cantidad de comunicaciones que se pueden establecer con el satélite. Esta cantidad no quedará delimitada por la conexión, sino por los proyectos a lo que se someta el Cubesat y los gastos energéticos que estos requieran.

### **Sección 3.1. Rendimiento de *brokers* y consumo de recursos**

En base a lo comentado anteriormente sobre IoT en Cubesat, este apartado se dispone a sugerir el broker MQTT más apropiado, dentro de los más usados, para ser implementado en la computadora de abordo (OBC) del Cubesat. Dicha elección tiene el objetivo de seleccionar el broker MQTT que brinde el mejor rendimiento y utilice la menor cantidad de recursos de la OBC. Estas consideraciones se tienen en cuenta debido a que el Cubesat cuenta con recursos limitados al igual que la OBC. Es importante destacar que el Cubesat estará compuesto por varias OBC, cada una destinada a una funcionalidad específica. Por lo tanto, la selección aquí comentada se refiere a la que

implementa el broker MQTT y la base de datos. El consumo de energía total del Cubesat debe ser mínimo (en la sección 4 se explican mayores detalles). Por tal motivo, cada OBC deberá consumir la menor energía posible siendo, por lo tanto, de suma importancia una utilización óptima de los recursos destinados al broker MQTT y a la base de datos.

Existen muchos protocolos de transferencia de datos M2M (intercambio de información entre dos máquinas) para sistemas IoT, como por ejemplo: CoAP, AMQP, HTTP, MQTT, entre otros. Considerando el consumo de energía, la confiabilidad, el ancho de banda, etc. MQTT se destaca frente al resto, al tiempo que tiene muy buen rendimiento en dispositivos que tengan capacidades de procesamiento y memoria limitadas. Por estas consideraciones MQTT es el protocolo adecuado para el desarrollo del Cubesat que es el propósito de nuestra intervención.

La selección del broker MQTT a ser sugerido está basado en un ensayo desarrollado en el documento titulado “Performance Evaluation of MQTT Broker Servers” (Biswajeeban, 2018). En éste se compara el desempeño de varios brokers y para ello se divide el experimento en dos partes: por un lado utilizando los *brokers* disponibles públicamente y por el otro implementados en forma local. Las comparaciones se realizan sobre el rendimiento de suscripción evaluando cuánto tiempo tarda un broker en enviar un paquete de datos al cliente. Estas se desarrollan en los distintos niveles de calidad de servicio (QoS) de cada broker MQTT, los cuales gestionan la robustez de los mensajes ante errores en la transmisión. Estos se dividen en tres niveles:

- a) QoS 0: El mensaje se envía por única vez. Ante errores puede que alguno no se entregue.
- b) QoS 1: El mensaje se envía hasta que se garantiza la entrega. Pueden recibirse mensajes duplicados en el transcurso de confirmación.
- c) QoS 2: Se garantiza la entrega y por única vez.

Para las pruebas se implementaron los brokers ActiveMQ, Bevywise, HiveMQ, Mosquitto y RabbitMQ, con el objetivo de poder evaluarlos según el QoS comentado anteriormente (Biswajeeban, 2018). De estos estudios se puede observar que en los niveles de calidad de servicio QoS0 y QoS2 la tasa de carga de Mosquitto fue más alta que los otros brokers en estudio. Por otro lado, también se observa que en los niveles de

calidad QoS0 y QoS2 Mosquito fue más rápido mientras que en QoS1 se obtiene una relativa paridad.

Dadas las implementaciones comentadas anteriormente, la que se puede asemejar más a la comunicación entre Cubesat y dispositivos en tierra es la implementación pública ya que cumple con el objetivo de tomar datos de sensores (en la descripción son simulados) y enviarlos al broker. La implementación local puede darse en el caso de las estaciones en tierra ya que, una vez recibidos los datos desde el Cubesat, se pueden utilizar brokers para la distribución de estos datos. Ante esta situación y los resultados obtenidos, se puede observar que Mosquitto es una opción recomendable. Para avalar lo comentado, se pueden observar los siguientes gráficos:

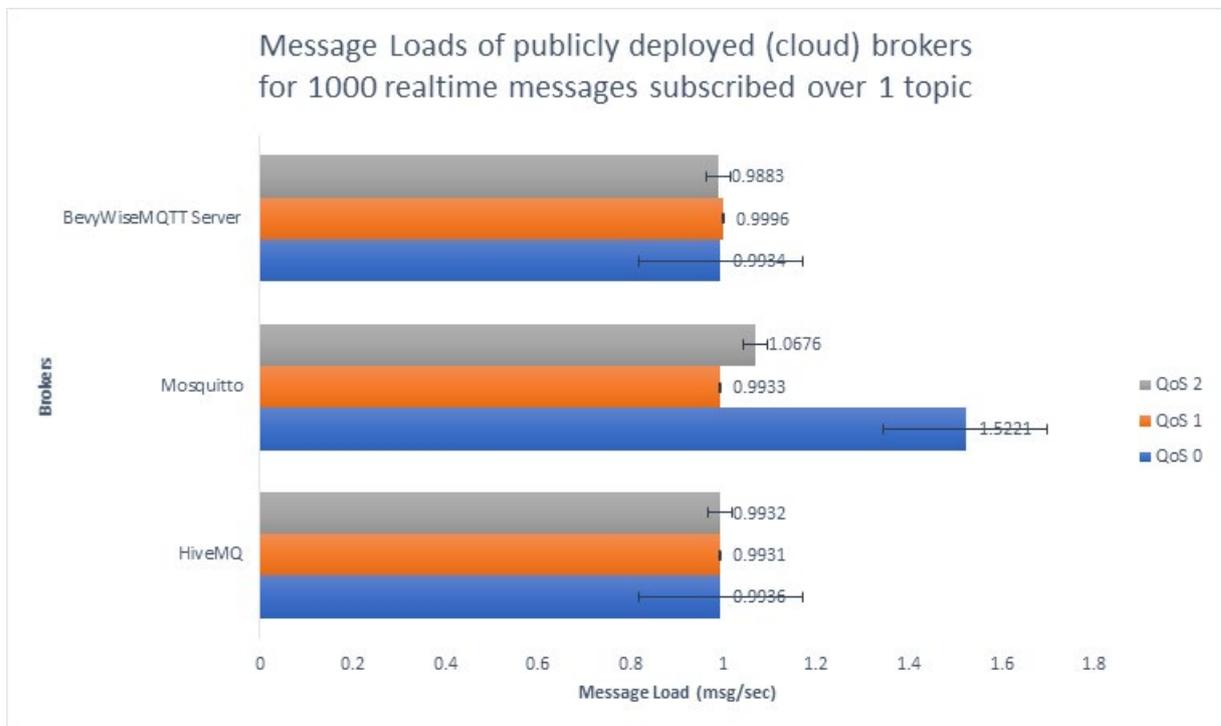


Figura 1. Tasa de carga de mensajes de brokers públicos. Fuente: Biswajeeban (2018)

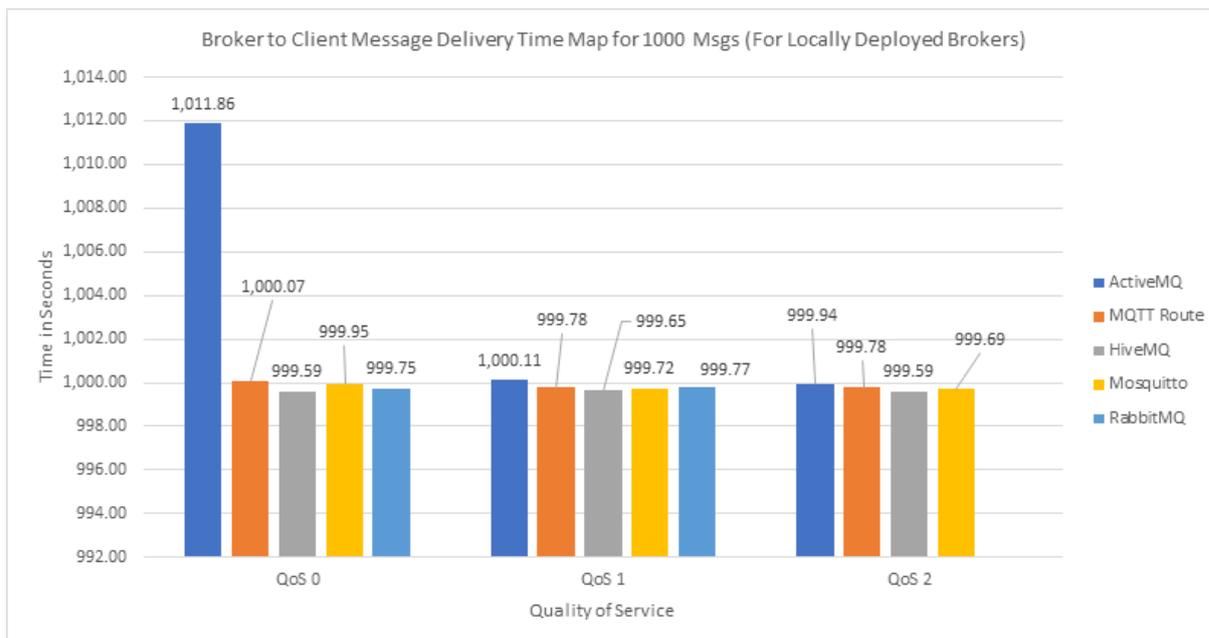


Figura 2. Mapa de tiempo de entrega de mensajes de brokers a cliente - brokers locales. Fuente: Biswajeeban (2018)

### Sección 3.2. Análisis de la base de datos

Para el análisis de este punto, hay que tener en cuenta que el mismo requiere de gran investigación dado que resulta importante conocer puntualmente los requerimientos mínimos de hardware. Sin embargo, para tener un acercamiento a este tema, se propone el estudio de bases de datos que sean de código abierto y escalables dada la gran aplicación que tienen en Cubesat. Tomando esta idea como punto de partida, se propone entonces bases de datos como: MongoDB y PostgreSQL. La primera es una base de datos relacional y además es de gran utilización en minería de datos dada la gran flexibilidad que ofrece para trabajar con distintos tipos de datos y tamaños. Por otro lado, SQL también es una base de datos relacional de gran aplicación (Zufelt, 2018).

### Sección 4. Recomendación de OBC y conclusiones

Además de lo expuesto hasta aquí en lo que respecta *al broker* y a la base de datos sobre la OBC, también se debe tener en cuenta la disponibilidad de potencia y el Sistema de Energía (EPS). Por esto, previamente a la recomendación de las distintas OBC con sus características, se comenta brevemente el sistema EPS.

El sistema de energía está compuesto por la generación, almacenamiento y distribución de energía, que representa un tercio de la masa y del volumen del Cubesat.

El método de generación de energía se realiza con paneles fotovoltaicos, lo cual agrega la complejidad de los lapsos de tiempo con interrupciones debido a los eclipses. Para subsanar este problema se cuenta con baterías que acumulan la carga y que puedan ser utilizadas cuando no se dispone de luz solar (Nasa, 2021).

En base a lo comentado, es importante ver que la cantidad de energía disponible limita el diseño de un nanosatélite y éste determina el tamaño. Ante esto último, se pueden agregar paneles solares en la superficie externa, aunque a veces esto no es posible ya que no todas las caras se encuentran disponibles pues algunas cuentan con instrumentos, antenas y cámaras, lo cual reduce la superficie total disponible.

Los paneles solares están compuestos por dos celdas para generar aproximadamente 2W con una eficiencia del 28% (Grau, 2019). En base a este análisis, a continuación se muestra una tabla con las características de los diferentes modelos disponibles en el mercado teniendo en cuenta la masa, el volumen y la potencia disponible:

Marca	Nombre	Satélite	Carga útil		
			Volumen	Volumen	Masa
		U	U	kg	W
Clyde Space		1	0,2		2
GomSpace	Basic	1	0,25	0,17	1,3
Space Inventor		2	1,25	0,3	20
BCT	XB3	3	2	4,5	60
Clyde Space		3	1,6		12
GomSpace	Advanced	3	2		3,5
ISIS	Basic	3	2	4	2
ISIS	Advanced	3	1,5	3	3,5
SSTL	Cube-X	3	2	2	6
Tyvak	Endeavour	3	2		65

*Tabla 1.* Características Mecánicas y eléctricas de plataformas Cubesat integradas. Fuente: Grau, 2019.

En base a la información provista en la Tabla 1, se puede ver que un Cubesat de tamaño U1 no puede superar los 2W de carga útil (*payload*, en inglés). Para potencias superiores, se puede avanzar a U2 o U3. Por esta razón, la computadora de abordo debe acotarse a estos límites de potencia. La electrónica utilizada en este tipo de Cubesat es de bajo nivel, con un grado de dificultad alto-medio para el desarrollo software y no cuenta con la flexibilidad del software de alto nivel (Grau, 2019).

OBC	MCU	Memoria incluida	Interfaces	Firmware / Sistema Operativo	Extras	Consumo
BeagleBone Black Board	ARM Cortex A8 1000 MHz	512 B EEPROM 512 MB DDR3 RAM 4GB eMMC Flash SD Card	I2C USB CAN SPI UART GPIO	Linux Debian Android Ubuntu Cloud9 IDE	RTC LAN PWM	2 W
Cube Computer	ARM Cortex M3 4-48 Mhz	1 MB SRAM 4 MB Flash	I2C CAN SPI UART		RTC ADC	
Endurosat OBC	ARM Cortex M7 Hasta 216 MHz	512 KB SRAM 2 MB Flash 2 x Micro SD	I2C USB RS485/422 SPI UART		RTC Magnetómetro y acelerómetro 3X	
GomSpace NanoMind Z7000	Dual ARM Cortex A9 Hasta 800 MHz	1 GB DDR3 RAM 32 GB storage	I2C CAN CSP	Linux	Módulo FPGA	2,3 W máx.
ISIS OBC	ARM9 400 MHz	512 KB FRAM 1 MB NOR Flash 64 MB SDRAM 4 GB SD card	I2C USB UART GPIO	FreeRTOS KubOS Linux	RTC Control de potencia	0,4 W prom.
Q7 Processor	ARM Cortex A9M Hasta 766 MHz	512 MB + 256 MB 2 x SD slots	USB RS232/422/485 CAN	Linux 4.14 LTS ROS	RTC	2 W

Tabla 2. Comparación de computadoras de a bordo (OBC). Elaboración propia en base a datos de proveedores.

Basados en todos los puntos comentados anteriormente, la recomendación de la OBC se centra en la OBC: “ISIS OBC”. Esta OBC brinda la posibilidad de usar KubOS Linux, contiene una memoria volátil de 64MB SDRAM, una memoria de almacenamiento a prueba de fallos de 2 x 2GB SD card y tiene un consumo de 400 mW promedio.

Existen otras alternativas como por ejemplo, “BeagleBone Black Board” (BBB). La misma también permite la utilización de linux, contiene una memoria volátil de 512MB y 4GB de almacenamiento SD card pero tiene como desventaja un consumo de 2W, la cual es mayor comparada con la ISIS OBC. Es importante destacar que, esta OBC, necesita un MotherBoard para su funcionamiento dentro del satélite, por eso es que dentro de esta recomendación, se debe incluir “Motherboard Module 2”.

Las recomendaciones están basadas en diversos artículos de investigaciones y de proveedores. El trabajo a futuro consta en realizar ensayos reales midiendo consumo de potencia, utilización de memoria y capacidad de procesamiento principalmente sobre el producto recomendado (ISIS OBC), siendo posible también, experimentar sobre otras placas de similares características, para tener un mayor punto de comparación.

Esto último es de relevante importancia dado que este trabajo se basa exclusivamente en un análisis exploratorio y de aproximación, para la toma de decisiones acerca del tema propuesto.

## **Agradecimientos**

Un especial agradecimiento a la Universidad Nacional de Avellaneda (UNDAV), ya que gracias a la universidad se pudo realizar este trabajo de investigación en la temática. También a la Universidad de Palermo (UP) por la gestión, como así también a todo el consorcio de universidades por la gran colaboración y participación.

## **Referencias**

- Alen Space. Guía básica de nanosatélites. Recuperado de: <https://alen.space/es/guia-basica-nanosatelites/> (visitado el 1/11/2021).
- Biswajeeban, M. (2018). Performance Evaluation of MQTT Broker Servers. En O. Gervasi et al. (eds.), *Computational Science and Its Applications – ICCSA 2018: Part IV* (pp.599-609), ISBN 978-3-319-95171-3. DOI 10.1007/978-3-319-95171-3\_47. Recuperado de: [https://www.researchgate.net/publication/326167412\\_Performance\\_Evaluation\\_of\\_MQTT\\_Broker\\_Servers](https://www.researchgate.net/publication/326167412_Performance_Evaluation_of_MQTT_Broker_Servers) (visitado el 1/11/2021).

- European Cooperation for Space Standardization. (2011). ECSS Q80-04 - Software metrication handbook. Recuperado de: <https://ecss.nl/hbstms/ecss-q-hb-80-04a-software-metrication-handbook/>
- Gonzalez Llorente, J. Puerto Leguizamón, G. (2014). Estimación de la cantidad de potencia suministrada por las celdas fotovoltaicas de un cubesat. Revista Tecnura, Nº18. Recuperado de: [https://www.researchgate.net/publication/276713968\\_Estimacion\\_de\\_la\\_cantidad\\_de\\_potencia\\_suministrada\\_por\\_las\\_celdas\\_fotovoltaicas\\_de\\_un\\_CubeSat](https://www.researchgate.net/publication/276713968_Estimacion_de_la_cantidad_de_potencia_suministrada_por_las_celdas_fotovoltaicas_de_un_CubeSat)
- Grau, S. (2019). Contributions to the Advance of the Integration Density of CubeSats. Technische Universität Berlin, Institute of Aeronautics and Astronautics: Scientific Series, ISBN 978-3-7983-3027-6. DOI 10.14279/depositonce-7293. Recuperado de: <https://depositonce.tu-berlin.de/handle/11303/8132> (consultado 1/11/2021).
- International Organization for Standardization. (2014). Estándar internacional para la evaluación de la calidad del software. ISO 9126.
- Kelapure, S. (2020). Design of a Software Architecture for Supervisor System in a Satellite (Master Thesis).
- Leppinen, H. (2017). Current use of Linux in spacecraft flight software, IEEE Aerospace and Electronic Systems Magazine, Nro. 32, pp. 4-13. DOI 10.1109/MAES.2017.160182. Recuperado de: [https://www.researchgate.net/publication/321788741\\_Current\\_use\\_of\\_linux\\_in\\_spacecraft\\_flight\\_software](https://www.researchgate.net/publication/321788741_Current_use_of_linux_in_spacecraft_flight_software) (consultado 1/11/2021).
- Li, J. Mark, P. Thomas, W. y Lee, R. (2013). Design of Attitude Control Systems for CubeSat-Class Nanosatellite, Hindawi Publishing Corporation Journal of Control Science and Engineering, Volumen 2013, Article ID 657182, 15 págs. DOI 10.1155/2013/657182. Recuperado de: <https://www.hindawi.com/journals/jcse/2013/657182/> (consultado el 1/11/2021).
- Narayanasamy, A. (2017). Nanosatellites constellation as an IoT communication platform for near equatorial countries, IOP Conference Series: Materials Science and Engineering. DOI 10.1088/1757-899X/260/1/012028. Recuperado de: <https://iopscience.iop.org/article/10.1088/1757-899X/260/1/012028> (consultado el 11/11/2021).
- Nasa. (2021). State-of-the-Art of Small Spacecraft Technology. Recuperado de: <https://www.nasa.gov/smallsat-institute/sst-soa-2020/power> (consultado el 1/11/2021).
- Sanny, O. Guglielmo, D. Di Mauro, G. Martin, T. y Bevilacqua, R. (2018). CubeSat Mission to Demonstrate Aerodynamically Controlled Re-Entry using the Drag De-Orbit Device (D3), University of Florida. Recuperado de: <https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=4147&context=smallsat> (consultado el 1/11/2021).
- Zufelt, B. (2018). CloudSat: IoT Approach to Small Satellite Ground Infrastructure, University of New Mexico. Recuperado de: [https://digitalrepository.unm.edu/ece\\_etds/450](https://digitalrepository.unm.edu/ece_etds/450) (consultado el 1/11/2021)