



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

Departamento:

Departamento de Ingeniería e Investigaciones Tecnológicas

Programa de acreditación:

CyTMA2

Programa de Investigación¹:

Código del Proyecto:

C2-ING-056

Título del proyecto

Dispositivo de asistencia de personas mediante monitoreo y análisis de datos en la nube

PIDC:

Elija un elemento.

PII:

Elija un elemento.

Director:

Lic. De Luca, Graciela Elisabeth

Codirector:

Ing. Valiente, Waldo Adolfo

Integrantes:

Graduado UNLaM:

Ing. García, Gerardo; Ing. Volker, Mariano

Alumnos de grado:

Sr. Villca, Raúl; Sr. Adagio, Matias

Alumnos de posgrado:

Ing. Carnuccio, Esteban

Resolución Rectoral de acreditación: N°

118/19

Fecha de inicio:

01/01/2018

Fecha de finalización:

31/12/2019

¹ Los Programas de Investigación de la UNLaM están acreditados con resolución rectoral, según lo indica la Resolución HCS N° 014/15 sobre **Lineamientos generales para el establecimiento, desarrollo y gestión de Programas de Investigación a desarrollarse en la Universidad Nacional de La Matanza**. Consultar en el departamento académico correspondiente la inscripción del proyecto en un Programa acreditado.



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

A. Desarrollo del proyecto (adjuntar el protocolo)

A.1. Grado de ejecución de los objetivos inicialmente planteados, modificaciones o ampliaciones u obstáculos encontrados para su realización (desarrolle en no más de dos (2) páginas)

A continuación se exponen los objetivos planteados inicialmente para el proyecto de investigación, junto con su porcentaje de completitud en cada uno:

Con respecto al objetivo general, se logró cumplir con el 90%. Ya que se desarrolló una solución IoT que permite la localización, detección de caídas, consulta de información y envío de alertas. El punto faltante del objetivo general, está en la obtención de datos biométricos. El cual había sido planteado utilizando un dispositivo externo tipo *Smart Band* que se iban a comunicar con el sistema embebido. Este punto no pudo cumplirse principalmente por el retraso en la disponibilidad presupuestaria y en el alza de los costos. Ya que cuando se presupuestó el proyecto, se estimó la compra de dos unidades por \$2.712, pero en el momento en que el grupo de investigación dispuso de los recursos, una sola unidad costaba alrededor de \$5.000. Por lo antes dicho, se decidió reacomodar los recursos para la construcción un prototipo funcional y que este posea mayor seguridad.

Con respecto a los objetivos específicos, se plantearon con distinta complejidad. En donde cada uno de ellos corresponde a una etapa o parte de la solución.

En el primero de los objetivos específicos (cumplido al 100%), se planteó que el dispositivo final sea de tamaño reducido, en alrededor de cincuenta por ciento o más con respecto del prototipo del proyecto anterior (Carnuccio, y otros, 2017). Como consecuencia de que este presentaba un tamaño de 100 x 130 x 200 mm, dando un volumen total de 2.600.000 mm³ (Fig. 1), con esas proporciones dificultaba su uso diario. Por ese motivo, para este nuevo proyecto, se diseñó el producto final utilizando la herramienta de modelado *Solid Edge*², con las medidas 95,7 x 57,8 x 39,6 mm, generando un volumen de 529.360 mm³ (Fig. 2). Por lo que el producto diseñado es de tan solo del 20%, comparándolo con el volumen total del prototipo del proyecto anterior.



Fig. 1 - Prototipo proyecto anterior.

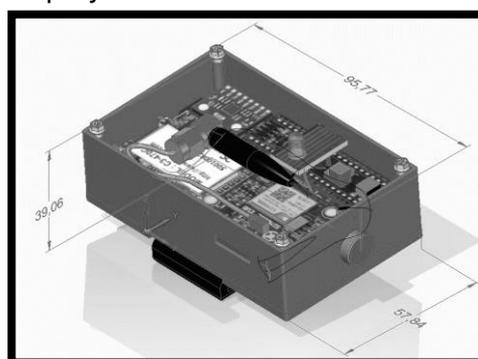


Fig. 2 - Diseño del nuevo dispositivo.

El segundo objetivo específico se enfocó en los métodos de comunicación y recolección de información del dispositivo embebido (cumplido al 95%). Como forma de comunicación se utiliza un módulo GSM con chip telefónico por red 2G, debido a que la tecnología 3G estaba fuera de presupuesto. Otra forma utilizada de comunicación es por Bluetooth versión 4.0. Se eligió esa versión porque permite el ahorro energético. Con respecto a la recolección de información se utilizó un acelerómetro para detectar caídas y un GPS para el almacenamiento de la localización de las personas monitoreadas. Además inicialmente se planteó la obtención de datos biométricos, pero por lo antes comentado con respecto a la *Smart Band* no se pudo desarrollar esta funcionalidad. Por

² URL: <https://solidedge.siemens.com/es/>



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

último utilizando los servicios expuestos en la nube se consigue dar aviso sobre la ocurrencia de distintos eventos e información. Para esto se utiliza un servidor programado con *Laravel*³ y servicios *FireBase*⁴, que permite enviar los distintos mensajes de alerta sobre la ocurrencia de los sucesos. El tercer objetivo (cumplido al 100%) requirió adaptar el algoritmo de caídas, desarrollado en el lenguaje interpretado *NodeJS* por el grupo de investigación, para que pueda funcionar en la nueva arquitectura. Esto demandó que el algoritmo sea reescrito y adaptado empleando el lenguaje C++. El gran problema fue pasar los tipos de datos dinámicos de *NodeJS* a estructurados de C++. Otro punto fue la calibración del algoritmo de caídas, que requirió de tener el prototipo funcional listo para realizar las caídas y permitir así realizar los ajustes del algoritmo para que pueda funcionar correctamente.

El cuarto objetivo específico (cumplido al 100%), solicitó que el dispositivo requiera mínimo de esfuerzo para su uso, de manera que sea fácil de utilizar. En consecuencia, el dispositivo obtenido no requiere esfuerzo en su utilización, debido a que todos los datos son recolectados y enviados al servidor automáticamente.

El quinto objetivo específico (cumplido al 95%), se centró en el ahorro de consumo energético del dispositivo. Para ello se utilizaron baterías recargables de 2.200 m *Amperes*. No obstante no se llegó a cumplir con la meta, de que funcione ininterrumpidamente 24hs sin recargar. Esto se debió a que módulo GSM (SIM800L) utilizado requiere de 2 *Amperes* para su funcionamiento, como consecuencia de la tecnología antigua que utiliza (2G). Los modelos de GSM con tecnología actual (4G), como el SIM7600CE-T, presentan un modo de reposo que permite el bajo consumo de tan solo 20 m *Amperes*. Sin embargo su precio se encontraba fuera del presupuesto y no se podían conseguir en el país.

El sexto objetivo específico no se desarrolló ya que requería el desarrollo de una conexión estándar y adaptable para distintos dispositivos de datos biométricos. Por lo antes dicho con respecto al *Smart Band*, no se realizó este punto.

El séptimo objetivo específico buscó incorporar mecanismos de seguridad en las comunicaciones del dispositivo (cumplido al 100%). En este sentido para la comunicación entre dispositivo-Servidor se creó un protocolo propio de encriptación. Esto fue por el bajo poder de procesamiento para encriptar en el dispositivo.



El mensaje cifrado posee dos secciones. La primera está formado por la dirección MAC del dispositivo, concatenado a un número interno que identifica el evento ocurrido que se desea notificar. Esos dos campos son encriptados utilizando MD5. La segunda parte posee las coordenadas que fueron obtenidas por el GPS en donde sucedió el evento y estas van encriptadas utilizando el algoritmo de XOR. Para este último método la clave utilizada se encuentra en la primer parte del mensaje, que esta encriptado en MD5. La cual no se puede desencriptar, por lo que el texto MD5 se encuentra indexado en el servidor. Utilizando ese índice el servidor sabe cuál es el dispositivo que viene el mensaje y como desencriptar el mensaje con XOR. Por otro lado la comunicación Android-Servidor se encripta utilizando el protocolo AES⁵ en ambos lados. Además, en la aplicación móvil se agregó seguridad al ingreso. La autenticación ahora se realiza utilizando servicios de *Facebook*. Toda la información de los mensajes es enviada en formato JSON encriptado por la biblioteca.

³ Entorno de desarrollo para servidor Web. URL: <https://laravel.com/>

⁴ Plataforma de productos Web. URL: <https://firebase.google.com/>

⁵ Cifrado estándar adoptado por el gobierno de los Estados Unidos.



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

Finalmente el Octavo objetivo específico planteaba que el dispositivo móvil (Android) posea un módulo de consulta y reportes (cumplido al 100%). Ya que con los datos del GPS muestra la ubicación de la persona monitoreada en cualquier momento y permite la visualización de su estado e información pertinente al mismo.

B. Principales resultados de la investigación

B.1. Publicaciones en revistas (informar cada producción por separado)

Sin publicaciones en Revistas.

B.2. Libros

Sin publicaciones de libros.

B.3. Capítulos de libros

Sin publicar capítulos de libros.

B.4. Trabajos presentados a congresos y/o seminarios

Primer artículo	
Autores	<i>Valiente, Waldo & Carnuccio, Esteban & Volker, Mariano & De Luca, Graciela & García, Gerardo & Guilianelli, Daniel & Villca, Raúl & Vittorio, Marcos</i>
Título	<i>Dispositivo de asistencia de personas mediante monitoreo IoT</i>
Año	2018
Evento	<i>XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018)</i>
Lugar de realización	<i>Universidad Nacional del Nordeste</i>
Fecha de presentación de la ponencia	<i>26 de Abril</i>
Entidad que organiza	<i>Red de Universidades con Carreras en Informática</i>
URL de descarga del trabajo (especificar solo si es la descarga del trabajo; formatos pdf, e-pub, etc.)	http://se-dici.unlp.edu.ar/bitstream/handle/10915/68034/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Segundo artículo	
Autores	<i>Valiente, Waldo & Carnuccio, Esteban & Volker, Mariano & De Luca, Graciela & Villca, Raúl & Lorenz, Germán</i>
Título	<i>Uso colaborativo del procesador en sistemas embebidos para múltiples interfaces</i>
Año	2019



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

Evento	<i>XXI Workshop de Investigadores en Ciencias de la Computación (WICC 2019)</i>
Lugar de realización	<i>Universidad Nacional de San Juan</i>
Fecha de presentación de la ponencia	<i>25 de Abril</i>
Entidad que organiza	<i>Red de Universidades con Carreras en Informática</i>
URL de descarga del trabajo (especificar solo si es la descarga del trabajo; formatos pdf, e-pub, etc.)	http://se-dici.unlp.edu.ar/bitstream/handle/10915/76970/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Tercer artículo	
Autores	<i>Valiente, Waldo & Carnuccio, Esteban & Volker, Mariano & De Luca, Graciela & Gerardo, García & Villca, Raúl & Lorenz, Germán & Matias Adagio</i>
Título	<i>Adaptación de Driver Serial para la placa STM32F103C8T6 y su utilización en controladores de GSM – Bluetooth</i>
Año	<i>2019</i>
Evento	<i>XXI Workshop de Investigadores en Ciencias de la Computación (WICC 2019)</i>
Lugar de realización	<i>Universidad Nacional de San Juan</i>
Fecha de presentación de la ponencia	<i>25 de Abril</i>
Entidad que organiza	<i>Red de Universidades con Carreras en Informática</i>
URL de descarga del trabajo (especificar solo si es la descarga del trabajo; formatos pdf, e-pub, etc.)	http://se-dici.unlp.edu.ar/bitstream/handle/10915/76951/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

B.5. Otras publicaciones

Sin otras publicaciones.

C. Otros resultados. Indicar aquellos resultados pasibles de ser protegidos a través de instrumentos de propiedad intelectual, como patentes, derechos de autor, derechos de obtentor, etc. y desarrollos que no pueden ser protegidos por instrumentos de propiedad intelectual, como las tecnologías organizacionales y otros. Complete un cuadro por cada uno de estos dos tipos de productos.

C.1. Títulos de propiedad intelectual. Indicar: Tipo (marcas, patentes, modelos y diseños, la transferencia tecnológica) de desarrollo o producto, Titular, Fecha de solicitud, Fecha de otorgamiento



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

No fue requerido tramitar títulos de propiedad intelectual.

C.2. Otros desarrollos no pasibles de ser protegidos por títulos de propiedad intelectual. Indicar: Producto y Descripción.

No fue requerido tramitar títulos de propiedad intelectual.

D. Formación de recursos humanos. Trabajos finales de graduación, tesis de grado y posgrado. Completar un cuadro por cada uno de los trabajos generados en el marco del proyecto.

D.1. Tesis de grado

Bajo esta investigación no se realizaron tesis de grado.

D.2 Trabajo Final de Especialización

Bajo esta investigación no se realizaron trabajos de especialización.

D.2. Tesis de posgrado: Maestría

Bajo esta investigación no se realizaron tesis de posgrado.

D.3. Tesis de posgrado: Doctorado

Bajo esta investigación no se realizaron tesis de doctorado.

D.4. Trabajos de Posdoctorado

Bajo esta investigación no se realizaron trabajos de posdoctorado.

E. Otros recursos humanos en formación: estudiantes/ investigadores (grado/posgrado/ posdoctorado)

Apellido y nombre del Recurso Humano	Tipo	Institución	Período (desde/hasta)	Actividad asignada ⁶
Villca, Raúl David	Grado	UNLaM	01-2018 al 12-2019	El alumno realizo tareas con respecto a la búsqueda bibliográfica y construcción de la aplicación para el móvil en <i>Android</i> .
Lorenz Vieta, German	Grado	UNLaM	01-2018 al 12-2019	El alumno colaboró en la búsqueda de información sobre dispositivos <i>GPS</i> , así como también en la programación del <i>driver</i> .

⁶ Descripción de la/s actividad/es a cargo (máximo 30 palabras)



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

Adagio, Matias	Grado	UNLaM	01-2018 al 12-2019	El alumno se introdujo en el tema rápidamente y colaboró en la búsqueda de información, así como también en la programación.
Vittorio, Marcos León	Grado	UNLaM	01-2018 al 12-2019	El alumno colaboró principalmente con la búsqueda bibliográfica para la confección de los <i>papers</i> e informe de avance en los temas de sistemas embebidos, Android

F. Vinculación⁷: Indicar conformación de redes, intercambio científico, etc. con otros grupos de investigación; con el ámbito productivo o con entidades públicas. Desarrolle en no más de dos (2) páginas.

G. Otra información. Incluir toda otra información que se considere pertinente.

--

H. Cuerpo de anexos:

- Anexo I: Copia de cada uno de los trabajos mencionados en los puntos B, C y D, y certificaciones cuando corresponda.⁸
- Anexo II:
 - FPI-013: Evaluación de alumnos integrantes. (si corresponde)
 - FPI-014: Comprobante de liquidación y rendición de viáticos. (si corresponde)
 - FPI-015: Rendición de gastos del proyecto de investigación acompañado de las hojas foliadas con los comprobantes de gastos.
 - FPI-035: Formulario de reasignación de fondos en Presupuesto.
- Anexo III: Alta patrimonial de los bienes adquiridos con presupuesto del proyecto (FPI 017)
- Nota justificando baja de integrantes del equipo de investigación.

⁷ Entendemos por acciones de “vinculación” aquellas que tienen por objetivo dar respuesta a problemas, generando la creación de productos o servicios innovadores y confeccionados “a medida” de sus contrapartes.

⁸ En caso de libros, podrá presentarse una fotocopia de la primera hoja significativa o su equivalente y el índice.



Código	FPI-009
Objeto	Guía de elaboración de Informe final de proyecto
Usuario	Director de proyecto de investigación
Autor	Secretaría de Ciencia y Tecnología de la UNLaM
Versión	5
Vigencia	03/9/2019

Firma y aclaración
del director del proyecto.

Lugar y fecha :.....

- Presentar una copia impresa firmada del presente documento junto con los Anexos, y enviar todo en archivo PDF por correo electrónico a la Secretaría de Investigación Departamental. **Límite de entrega: 28 de febrero de 2020**



ANEXO I: Publicaciones realizadas durante el proyecto de investigación.

En total se realizaron tres publicaciones una durante 2018 y las restantes dos en 2019.

1. WICC 2018 – POSTER DEL PRIMER ARTÍCULO.....	2
2. WICC 2018 - PAPER DEL PRIMER ARTÍCULO.	3
3. WICC 2019 – POSTER DEL SEGUNDO ARTÍCULO.....	8
4. WICC 2019 – PAPER DEL SEGUNDO ARTÍCULO.	9
5. WICC 2019 – POSTER DEL TERCER ARTÍCULO.	14
6. WICC 2019 – PAPER DEL TERCER ARTÍCULO.....	15



1. Wicc 2018 – Poster del primer artículo.



Dispositivo de asistencia de personas mediante monitoreo IoT

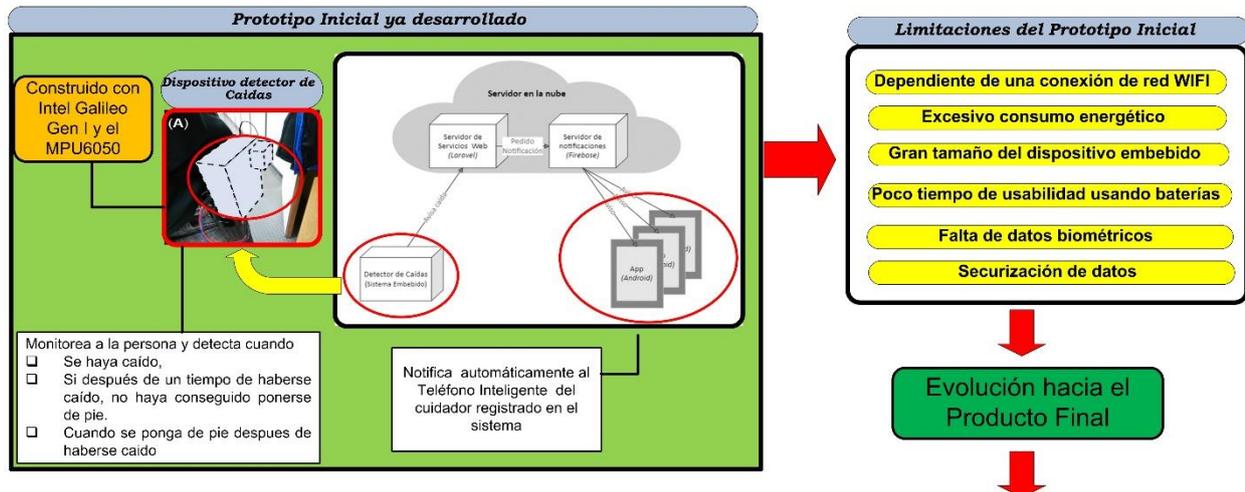
Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca, Gerardo Garcia, Daniel Giulianielli, Raúl Vilca, Marcos Vittorio

ISS
Innovación en Sistemas de Software

- Objetivos**
- Desarrollar un dispositivo simple, amigable y de bajo costo para el cuidado de las personas mediante IoT
 - Únicamente realizará funciones de monitoreo y del estado de las personas.
 - De fácil uso por parte del monitoreado y sus cuidadores
 - Diseñar circuitos impresos del dispositivos y su software a medida para una posterior producción en serie
 - Contará con el sistema detector de caídas desarrollado por el equipo de investigación, adaptándolo a un dispositivo pequeño y con baterías de larga duración.

- Contexto**
- Esta línea de investigación forma parte del proyecto denominado " Dispositivo de asistencia de personas mediante monitoreo y análisis de datos en la nube", dependiente del Departamento de Ingeniería de la Universidad Nacional de La Matanza

Investigación



Mejoras necesaria en el Producto final

Mejoras a realizar en el dispositivo	Mejoras a realizar en el servidor en la nube	Mejoras a realizar en la aplicación móvil
<ul style="list-style-type: none"> <input type="checkbox"/> Reducir su tamaño y consumo energético migrando a otra arquitectura (ejemplo Cortex-M3 o ARM Mono) <input type="checkbox"/> Utilizar comunicación con la nube por medio red GSM, utilizando solo paquetes de datos 3g y 4G. <input type="checkbox"/> Diseñar un circuito impreso del dispositivo para su producción en serie 	<ul style="list-style-type: none"> <input type="checkbox"/> Optimizar el sitio donde se almacenan los datos <input type="checkbox"/> Encriptar o cifrar la transferencia de datos 	<ul style="list-style-type: none"> <input type="checkbox"/> Permitirle al cuidador realizar distintas consultas sobre los datos biométricos del monitoreado <input type="checkbox"/> Permitir generar gráficos estadísticos <input type="checkbox"/> Agregar securización de los datos utilizados

Resultados esperados

- Desarrollar un nuevo prototipo mejorado IoT, dando la base para la elaboración de un producto final que incluya las mejoras
- Diseñar un circuito impreso con todos los componentes, un servidor web securizado y aplicaciones Android masivas.

Líneas de investigación

- Adaptar el sistema detector de caídas desarrollado anteriormente a la nueva arquitectura
- Implementar las mejoras en el dispositivo, en el servidor en la nube y en la aplicación móvil

Formación de Recursos Humanos

- La presente línea de investigación forma parte del trabajo que se encuentra realizando un integrante del grupo de investigación para su maestría.
- Grupo de trabajo conformado por dos investigadores inicial y dos alumnos que se encuentran finalizando sus carreras de grado



2. Wicc 2018 - Paper del primer artículo.

Dispositivo de asistencia de personas mediante monitoreo IoT

*Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca,
Gerardo García, Daniel Giulianelli, Raúl Villca, Marcos Vittorio
Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza*

*Dirección: Florencio Varela 1703 – CP 1754 – {wvaliente, ecarnuccio, mvolker,
gdeluca, ggarcia, dgiulian}@unlam.edu.ar, raul.villcasd@gmail.com,
marcosvittorio@hotmail.com*

RESUMEN

Los objetos y las personas, gracias a la tecnología actual, pueden ser comunicados y utilizados para mejorar las condiciones de vida. Este trabajo intenta hacer llegar la tecnología para el cuidado de las personas, realizando un aporte social, ya que se trata de desarrollar un dispositivo simple, amigable y de bajo costo. Será simple debido a que solo realizará funciones para el monitoreo y comunicación del estado de la persona. Amigable, ya que será fácil de usar para el monitoreo y contará con una aplicación Android en un dispositivo del cuidador. Los datos obtenidos durante el monitoreo serán almacenados en un servidor en la nube pudiendo ser procesados y utilizados posteriormente de ser necesario. Contará con el sistema de caídas desarrollado por este equipo de investigación, adaptándolo a un dispositivo pequeño y con baterías de larga duración. De bajo costo, debido a una producción en serie mediante circuitos y software a medida.

Palabras clave: *Computación en la nube, securización de datos, monitoreo de personas, geolocalización, IoT, Android, giroscopio, acelerómetro.*

CONTEXTO

Nuestra Línea de Investigación es parte del proyecto *Dispositivo de asistencia de personas mediante monitoreo y análisis de datos en la nube*, dependiente de la Unidad Académica del *Departamento de Ingeniería e Investigaciones Tecnológicas*, perteneciente al programa de Investigaciones PROINCE de la Universidad Nacional de La Matanza, el cual es formado por docentes, investigadores y alumnos de las carreras de ingeniería en informática e ingeniería en electrónica. Este proyecto es continuación de los trabajos que viene realizando el grupo de investigación, en sistemas operativos, computación de alto rendimiento, entre otros, en el área de Internet de las cosas (IoT).

1. INTRODUCCIÓN

Los grandes avances que ha sufrido la

tecnología en los últimos años han influido considerablemente en la vida cotidiana de las personas de cualquier edad. Hoy en día, niños, adolescentes y adultos emplean distintos elementos tecnológicos para estar interconectados entre sí, compartiendo información involuntariamente. Aprovechando esta característica evolucionó el concepto de Internet de las Cosas, generando relaciones entre los objetos utilizados en la vida diaria que antes era impensados realizar, produciendo una gran cantidad de datos que se puede analizar a través de Big Data [1]. Por consiguiente, existen diferentes disciplinas en las que se puede implementar sistemas que hagan uso de IoT. En este sentido es de suma importancia la aplicación de la tecnología para el cuidado de niños, personas mayores, enfermos, o con capacidades diferentes. Siendo de vital importancia la asistencia rápida y oportuna, ante determinadas situaciones que puedan sufrir esas personas. Tratando de esta forma de prevenir problemas mayores de salud. En ese contexto, en la investigación anterior [2], se implementó el prototipo de un sistema para detectar las caídas de personas mayores de edad, utilizando las placas Intel® Galileo Generación I y el sensor MPU6050. El sistema de detección de caídas está compuesto por distintos programas, que se muestran en la Fig. 1, donde cada uno de ellos realiza una funcionalidad específica.

El detector de caídas en el sistema embebido [2], ejecuta un programa que detecta caídas de la persona mayor que lo esté utilizando. Su programación emplea un algoritmo que recolecta y analiza la información motriz diaria de la persona adulta, con el fin de detectar la ocurrencia de cualquiera de los siguientes eventos. Estos sucesos son una mejora de los propuestos por [3] y [4] en sus trabajos.

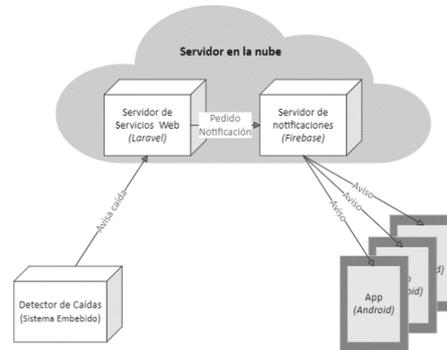


Fig. 1-Componentes del sistema detector de caídas.

El primero de ellos ocurre en el momento que se haya producido una caída, el segundo al transcurrir un determinado tiempo de haberse caído la persona y no haya conseguido ponerse de pie. Finalmente, un tercero cuando se reincorpore. Cuando el programa detecte algunos de estos eventos, automáticamente enviará mensajes de alertas a una aplicación Android, que estará instalada en el teléfono inteligente de la persona a cargo del mayor. Para realizar este cometido, se utiliza un servidor en la nube como intermediario para la transferencia de datos.

Con la experiencia lograda en la investigación anterior [2], se advierte la necesidad de evolucionar el dispositivo detector de caídas. Como anexo se agregarán capacidades que permitirán obtener parámetros biométricos simples y de ubicación, para una rápida asistencia. Con la evolución de los teléfonos inteligentes, sus sensores y la facilidad para conectarse, desarrollar una solución con este tipo de tecnología parece tentadora. Sin embargo en la investigación se considera que esta opción no resulta viable por tres factores. El primero por el marco de nuestra economía, los equipos móviles tienen un alto costo. El segundo algunas personas no suelen estar familiarizadas con esta tecnología y lo ven



como algo extraño e intrusivo. El tercer factor requiere constante intervención humana, no siendo un dispositivo dedicado para tal fin. Por eso se pretende desarrollar un dispositivo a medida con tecnologías asentadas que son utilizadas en nuestro país. Esto permite que sea de bajo costo y simple de utilizar tanto [5].

2. LINEAS DE INVESTIGACIÓN Y DESARROLLO

El apartado anterior da cuenta de la obtención de un producto estable que cumple con las funcionalidades propuestas, incluso superando a proyectos en una línea de investigación similar. Sin embargo al plantearse la viabilidad de convertirlo en un producto de uso diario se detectaron ciertas falencias. En este sentido las limitaciones que se destacan en el prototipo desarrollado, son el consumo de batería, el tamaño, su necesidad de depender de conexión de Wifi y la falta de datos biométricos. Por otro lado resulta necesario que el servidor permita emplear mecanismos de securización en los datos intercambiados hacia la nube, para mayor seguridad de la información privada de los usuarios. Además es necesario que la aplicación del teléfono móvil muestre mayor información detallada y crezca en funcionalidades en concordancia a lo anteriormente mencionado.

Mejoras al prototipo

Con respecto al consumo de batería resultante, según la especificación del producto [6], las placas Intel® Galileo GEN 1 requieren un voltaje de 5 volts y tiene un consumo de 800mA (Estos consumos no incluyen los periféricos como sensores y Wifi). No obstante las baterías recargables y portables que pueden producir esos niveles de tensión requieren de un tamaño superior a la de las pilas AA. La carga útil de estas baterías apenas logran cubrir el uso de pruebas con el dispositivo, ya que requieren ser

recargadas constantemente, cada dos horas aproximadamente. Por consiguiente lo que se busca reducir es el consumo del dispositivo, cambiando la arquitectura. Por ese motivo se están investigando varios modelos, por ejemplo los dispositivos de la familia Cortex-M3®, que requieren de un voltaje que puede llegar a los 3.6 v y el consumo puede rondar entre los 80 a 150 mA [7] [8]. Otras tecnologías como ARM nano® tiene voltajes de referencia de 3.3v y un consumo de 50mA [9].

La primera versión del prototipo presentaba elevadas dimensiones y no resultaba cómodo para ser utilizado de modo diario, debido a que es necesario llevarlo en la cintura (Fig. 2 - A). En consecuencia, al migrar de arquitectura se busca conseguir un menor tamaño del producto final. La ubicación del dispositivo está condicionada al diseño del algoritmo de caídas, donde se espera que éste se encuentre en una posición fija y vertical en el cuerpo del usuario, ya sea en su cintura o pecho, descartando así lugares como muñeca o tobillos. Esta ubicación, en el algoritmo, da un eje fijo de referencia ya que siempre se encuentra afectado por la gravedad. Por consiguiente el nuevo prototipo debería estar en la cintura. A modo de ejemplo la (Fig. 2 - B) muestra las nuevas proporciones esperadas para la siguiente generación del producto. Reduciendo así el tamaño a menos del 20% del prototipo actual.

Cabe aclarar que más adelante en la etapa final del proyecto, se va a desarrollar un circuito impreso, diseñado a medida, que miniaturice aún más el producto final.

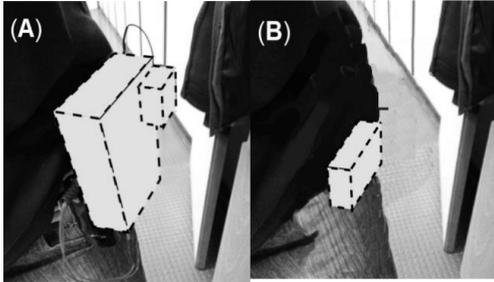


Fig. 2 Dimensiones del prototipo actual (A).
Dimensiones esperadas del nuevo prototipo (B).

Con respecto a la forma de conectarse a la nube, se planteó la necesidad de que el dispositivo esté conectado más allá de Wifi, que es de uso interno. Por esto se utilizará conexión por red GSM, utilizando solo los paquetes de datos 3G/4G.

Como datos biométricos se están evaluando la implementación de sensores que permitan el monitoreo del estado de una persona. Como por ejemplo pulsometro, de temperatura, de ritmo cardiaco, entre otros. Junto con datos de la localización de la caída obtenida por medio de un GPS.

Mejoras al Servidor web

El dispositivo embebido al recolectar mayor cantidad de información y al ser datos sensibles de la persona monitoreada, surgen dos requisitos a agregar al servidor web. El primero con respecto al almacenamiento de la información, solo con fines de una posterior revisión. Mientras que el segundo, es en cuanto a la encriptación o cifrado en la transferencia de los datos.

Mejoras a la aplicación móvil

La aplicación móvil actualmente permite realizar la gestión de personas a monitorear y la recepción de los sucesos reportados por el dispositivo que realiza el monitoreo. La nueva

solución, al recolectar datos biométricos, será necesario adaptar la aplicación móvil para que realice distintas consultas y generar gráficos estadísticos. Además será necesario que la información consultada sea transferida con algún tipo de securización confiable para mayor seguridad.

3. RESULTADOS OBTENIDOS/ESPERADOS

Las mejoras que se proponen realizar en el presente trabajo de investigación, tiene como objetivo desarrollar un nuevo prototipo mejorado IoT. Dando la base para la elaboración de un producto final, que incluya las mejoras mencionadas en los párrafos previos. De esta forma se pretende diseñar un circuito impreso con todos los componentes, un servidor web securizado y aplicaciones Android masivas.

4. FORMACIÓN DE RECURSOS HUMANOS

La presente línea de investigación dentro del departamento de Ingeniería e Investigaciones Tecnológicas forma parte del trabajo que uno de los investigadores se encuentra realizando para su maestría.

Completan el grupo de investigación dos investigadores en formación y dos alumnos que se encuentran finalizando sus carreras de grado.

5. BIBLIOGRAFÍA

- [1] Bonilla-Fabela, Morales-Escobar y Guajardo-Muñoz, «IOT, El Internet de las Cosas y la Innovación de sus Aplciaciones,» 2 ed., 2016, pp. 2313-2340.
- [2] Esteban Carnuccio, Waldo Valiente, Mariano Volker, Graciela De Luca, Gerardo García, Daniel Giulianelli, Sebastián Barillaro, «Desarrollo de un Prototipo



- detector de caídas utilizando la placa Intel Galileo Generación I y el sensor MPU6050.» de *IX Workshop Innovación en Sistemas de Software (WISS)*., La Plata, 2017, pp. 954-964.
- [3] R. Blanco, «Sistema de detección de caída en personas de la tercera edad para uso en centro geriátricos,» Bogota, 2010.
- [4] E. Oporto Díaz, «Diseño de un sistema inalámbrico de detección de caídas aplicado a personas de la tercera edad basado en acelerómetro y teléfono móvil,» Lima-Peru, 214.
- [5] J. Aceros, M. Cavalcante y M. Doménech, «Identidad de lugar en usuarios de teleasistencia un análisis conversacional,» Bellaterra, España, 2013.
- [6] Intel(R), «Galileo Datasheet,» de *Intel Galileo*, 2013, p. 13.
- [7] STMicroelectronics, «Datasheet - production data Ultra-low-power 32-bit MCU ARM®-based Cortex®-M3,» de *STM32L151x6/8/B STM32L152x6/8/B*, STMicroelectronics, 2016.
- [8] STMicroelectronics, «Datasheet - production data- Medium-density performance line ARM®,» de *STM32F103x8*, STMicroelectronics, 2015.
- [9] «ARM-N8LD,» de *Narrow band ISM Transceiver 868MHz / 27 Narrow band ISM Transceiver 868MHz / 27dBm*, ATIM, 2015.

3. Wicc 2019 – Poster del segundo artículo.

Uso colaborativo del procesador en sistemas embebidos para múltiples interfaces

Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca, Raúl Villca, Germán Lorenz

Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza

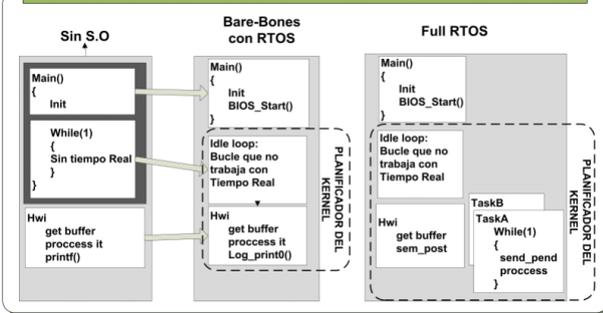
ARSO

Arquitectura, Redes y
Sistemas Operativos

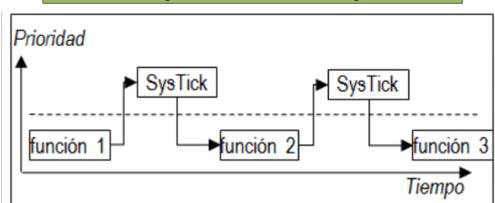
Resumen

Las soluciones IoT requieren de sistemas embebidos cada vez más complejos, como es el caso de nuestra investigación para el monitoreo de personas, asistencia ante caídas, signos vitales anómalos y seguridad. Donde muchas interfaces entre los sensores y los canales de comunicación se deben atender junto con la lógica del programa principal. En el presente trabajo se analizan técnicas de programación alternativas y funcionalidades provistas por los **SISTEMAS OPERATIVOS DE TIEMPO REAL**, que brindan la posibilidad de alternancia de tareas sobre un único procesador. Se proponen soluciones para compartir el procesador evitando en lo posible el uso de funciones provistas por los sistemas operativos de tiempo real, ya que estos consumen una cantidad de recursos considerables para estas arquitecturas. Se propone solucionar esta dificultad mediante la implementación de una **MÁQUINA DE ESTADOS** a través del uso de **INTERRUPCIONES**.

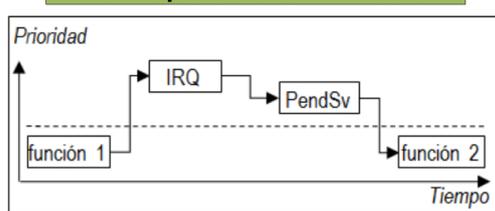
Clasificación de Implementación en Sistemas Embebidos



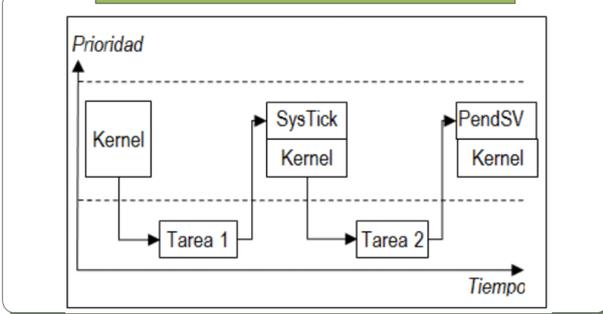
Interrupción Hardware SysTick



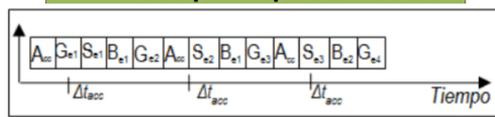
Interrupción Hardware PendSV



Cambio de Contexto Utilizando RTOS



Alternancia por Máquina de Estados



Líneas de investigación

- Análisis de como gestionar los recursos para múltiples interfaces sobre un único hilo de ejecución.
- La arquitectura ARM Cortex-M3 permite los métodos:
 - o Cambios de contexto por hardware (Interrupciones *SysTick* y interrupciones *PendSV*).
 - o Cambios de contexto por software, utilizando bibliotecas *RTOS* (alguna de ellas *FreeRTOS*, *mbedOS*, *NuttX*, *RTEMS*, *ChibiOS*).
- Se compara con un modelo de máquina de estados finito.

Formación de Recursos Humanos

- La presente línea de investigación forma parte del trabajo que se encuentra realizando dos integrantes del grupo de investigación para su maestría.
- Completan el grupo de investigación cuatro alumnos de Ingeniería en Informática que se encuentran finalizando su formación de grado se encuentran comenzando su carrera como investigador.

Resultados Obtenidos / Esperados

- Se puede obviar la utilización de un RTOS siempre que el programa no utilice funciones bloqueantes.
- En la investigación en curso se realiza la prueba de una solución híbrida:
 - o La lógica de las interfaces de GSM, GPRS y Bluetooth, menos prioritarias, se desarrollarán siguiendo el modelo de máquina de estado finito.
 - o Para la interfaz de acelerómetro, que es pieza clave en la detección de caídas, se estará utilizando interrupciones por Systick.



4. WICC 2019 – Paper del segundo artículo.

Uso colaborativo del procesador en sistemas embebidos para múltiples interfaces

*Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca,
Raúl Villca, Germán Lorenz*

*Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza
Dirección: Florencio Varela 1703 – CP 1754 –
{wvaliente, ecarnuccio, mvolker, gdeluca}@unlam.edu.ar,
{raul.villcasd, germangelv}@gmail.com*

RESUMEN

Las soluciones IoT requieren de sistemas embebidos cada vez más complejos, como es el caso de nuestra investigación para el monitoreo de personas, asistencia ante caídas, signos vitales anómalos y seguridad. Donde muchas interfaces entre los sensores y los canales de comunicación se deben atender junto con la lógica del programa principal. Estos a su vez se construyen sobre pequeñas arquitecturas con procesadores de rendimiento moderado impulsados por el bajo consumo energético que se busca. En el presente trabajo se analizan técnicas de programación alternativas y funcionalidades provistas por los Sistemas Operativos de Tiempo Real, que brindan la posibilidad de alternancia de tareas sobre un único procesador. También al mismo tiempo se evalúan las necesidades de recursos, que en estos tipos de procesadores son reducidas. Se proponen soluciones para compartir el procesador evitando en lo posible el uso de funciones provistas por los sistemas operativos de tiempo real, ya que estos consumen una cantidad de recursos considerables para estas arquitecturas. Se propone solucionar esta dificultad mediante la implementación de una

máquina de estados a través del uso de interrupciones, como única solución o una combinación de ambas.

Palabras clave: *RTOS, interrupciones, monitoreo de personas, máquina de estado, funciones bloqueantes.*

CONTEXTO

Nuestra Línea de Investigación es parte del proyecto “Dispositivo de asistencia de personas mediante monitoreo y análisis de datos en la nube”, dependiente de la Unidad Académica del Departamento de Ingeniería e Investigaciones Tecnológicas, perteneciente al programa de Investigaciones CYTMA2 de la Universidad Nacional de La Matanza, el cual está formado por docentes investigadores y alumnos de las carreras de ingeniería en informática e ingeniería en electrónica. Este proyecto es continuación de los trabajos que viene realizando el grupo de investigación, en sistemas operativos, computación de alto rendimiento e Internet de las cosas (IoT), entre otros.

1. INTRODUCCIÓN

Los sistemas embebidos que forman parte de un sistema complejo de IoT [1], se construyen sobre arquitecturas simples y colaborativas como el procesador ARM de la familia Cortex-M3 [2] [3]. Estos sistemas tienen la complejidad de atender y gestionar los recursos para múltiples interfaces sobre un único hilo de ejecución. La dificultad radica en cómo repartir el recurso escaso, en este caso el procesador, brindando atención a cada interfaz, siempre manteniendo los tiempos de respuesta requeridos en todos los dispositivos que se utilizan. En nuestra investigación [4] se evidencia que cuando se utilizan interfaces complejas se requiere realizar una serie de pasos o procedimientos compuestos para lograr dar respuesta a una solicitud. Como puede ser el caso de conexión a Internet por GSM, conexión local por Bluetooth y obtener datos desde el GPS, junto a otros sensores que requieren un muestreo periódico. Tal es el caso del acelerómetro, donde se evalúan las mediciones para verificar si la persona que lleva el dispositivo se encuentra en una caída o no.

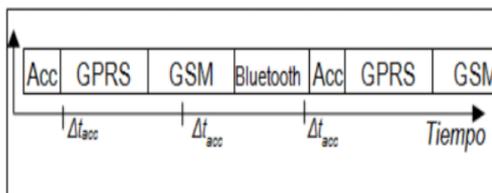


Figura 1 – Múltiples funciones sobre un único hilo de ejecución.

Se puede observar en la Figura 1 que sobre un único procesador ejecutan varias funciones. La función que requiere ejecutar regularmente y la del acelerómetro (Acc), no siempre logra cumplir con su periodicidad (Δt_{acc}) requerida para el correcto funcionamiento del algoritmo de caídas.

2. LINEAS DE INVESTIGACIÓN Y DESARROLLO

A continuación se plantean distintos métodos para lograr repartir el uso del único procesador entre varias funciones que se necesitan ejecutar del sistema embebido y reconociendo aquellas que son críticas.

Las implementaciones en sistemas embebidos podrían clasificarse en non-os, bare-bones con RTOS¹ y Full RTOS. En el caso de una implementación sin S.O se programa un binario con todo el programa en sí, que realiza la inicialización-ejecución principal y finalización. En el caso de bare-bones con RTOS es un mix entre ambos métodos. Por último en el caso del uso de Full RTOS, la funcionalidad necesaria se implementa como una aplicación convencional en un sistema operativo de tiempo real, Figura 2.

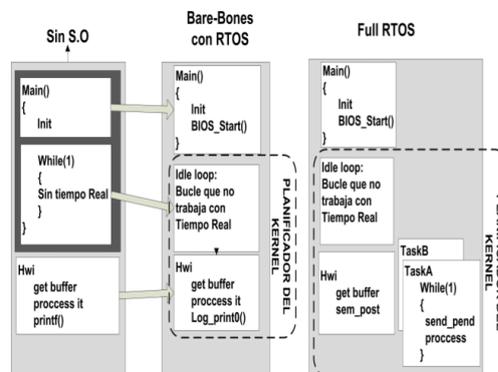


Figura 2 – Clasificación de implementación en sistemas embebidos.

Cambios de contexto por hardware

La arquitectura ARM Cortex-M3 [5] facilita dos mecanismos para interrumpir el programa en ejecución. El primero *SysTick* es un reloj de 24 bits que puede interrumpir al procesador cada vez que su contador llegue a cero. Este

¹ RTOS Real Time Operating System

mecanismo es utilizado para realizar cambios de contexto al estilo *round-robin*, Figura 3.

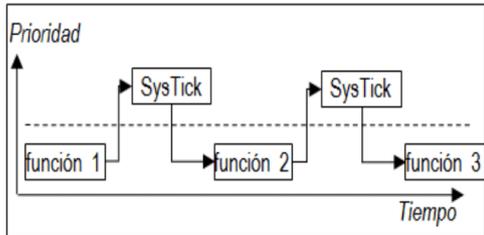


Figura 3 - Interrupción SysTick.

En el segundo mecanismo *PendSV* el programa permite ceder el control del CPU al manejador, cuando este se encuentre inactivo. Se puede utilizar con cambios de contexto al estilo de FIFO Figura 4.

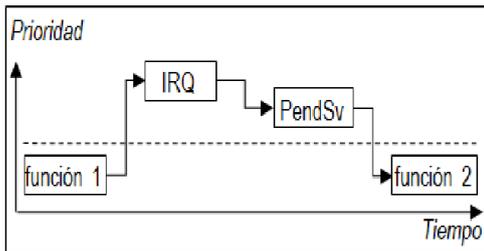


Figura 4 - Interrupción PendSV.

La ventaja de este método es que el programador tiene el control y puede definir el comportamiento que necesite para su implementación. La desventaja se tiene, cuando debe desarrollarse la lógica del planificador en forma más robusta, esta tarea suele ser engorrosa y difícil de probar la integridad del mismo.

Cambios de contexto por software

Los cambios de contexto se realizan agregando bibliotecas de terceros en el programa. Estas poseen tal nivel de integración brindando servicios directamente a nivel de Sistema

Operativo de Tiempo Real (RTOS) [6] en reducido tamaño, que facilitan su programación e implementación. En el mercado existen abundante cantidad de desarrollos para la arquitectura en cuestión, a continuación se enumeran los más destacadas:

La versión de *mbedOS* [7], entre sus principales características se destaca la seguridad y la conexión de dispositivos, mientras que *FreeRTOS* [8], facilita la programación, implementación e integración con servicios en la nube de Amazon™. El caso de *NuttX* [9], pensado para dispositivos desde 8 a 32bits basado en el estándar de APIs Unix. El pequeño kernel de *μ-velOSity* [10], es diseñado para ser eficiente y ocupar poca memoria. *RTEMS* [11], es soportado en gran cantidad de arquitecturas. *ChibiOS* [12], es un pequeño kernel que permite la integración con otros componentes *OpenSource* para brindar más servicios.

Los RTOS facilitan la implementación ya que los cambios de contexto, junto con otras funciones de comunicación de tareas, ya se encuentran integradas en el kernel, el desarrollador solo se limita a desarrollar la lógica de la aplicación para las tareas que ejecutan ver Figura 5.

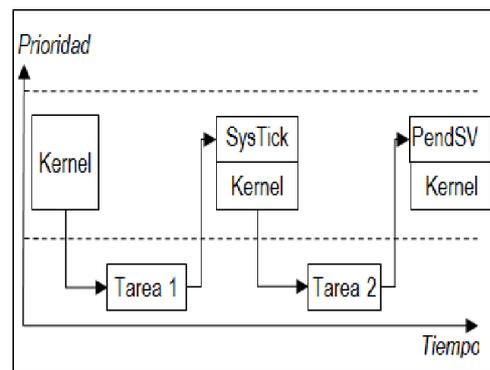


Figura 5 – Cambio de contexto utilizando RTOS

La ventaja de este mecanismo es que soluciona los problemas planteados con bibliotecas probadas por la comunidad y acelera el tiempo de desarrollo. La desventaja que es dificultosa la coordinación entre tareas fuertemente acopladas y la utilización de software de terceros amplía el espacio que ocupara el programa en el reducido espacio de memoria que posee la arquitectura embebida.

Modelo máquina de estado finito

La alternativa que plantea nuestro grupo de investigación, permite distribuir el uso del único procesador entre varias funciones. La misma radica en descomponer la lógica de cada función en sub-partes no bloqueantes. Así el programa principal, compagina su ejecución entre las distintas sub-partes de cada función.

El método encontrado consiste en organizar esta división utilizando el modelo de máquina de estado finito. Así las funciones de las interfaces tienen un comportamiento distinto, según el estado que se encuentre. En cada interacción del programa se evalúa el estado en el que se encuentra la aplicación, ejecutando la pequeña sub-parte que le corresponde. A continuación, se evalúa la siguiente porción de código de la función siguiente.

En la Figura 6, se observa que la función del acelerómetro (Acc) logra cumplir con la periodicidad requerida. Mientras se altera la ejecución de las sub-partes del GPRS ($G_{e1}, G_{e2}, G_{e3}, G_{e4}$), para el GSM (S_{e1}, S_{e2}, S_{e3}) y Bluetooth (B_{e1}, B_{e2}).

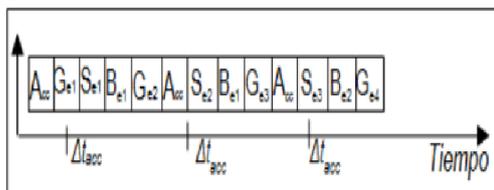


Figura 6 – Alternancia por estado de funciones.

La ventaja de esta solución es que no ocupa mucho lugar en memoria y no requiere de grandes cambios de implementación, comparándolo con la migración del programa al utilizar RTOS. Como desventaja, sin un buen diseño de la transición de estados, puede haber partes de funciones que nunca se ejecuten. Otra desventaja es que no se diferencian prioridades entre las funciones de interfaces.

3. RESULTADOS OBTENIDOS/ESPERADOS

Luego del análisis de las alternativas se confirma, que se puede obviar la utilización de un RTOS siempre que el programa no utilice funciones bloqueantes. Los RTOS son una solución compleja que brinda beneficios en casos que el código sea bloqueante, sin poder desarrollarlo de otra forma. La propuesta de máquina de estado es equitativa en el uso del procesador, pero no satisface estrictamente el cumplimiento de la periodicidad necesaria en la función crítica. En la investigación en curso se realiza la prueba de una solución híbrida. La lógica de las interfaces de GSM, GPRS y Bluetooth, menos prioritarias, se desarrollarán siguiendo el modelo de máquina de estado finito, que permite el uso del procesador colaborativamente.

Para la interfaz de acelerómetro, que es pieza clave en la detección de caídas. Se realizara un pequeño módulo que permita cambios de contexto utilizando interrupciones *Systick*. Luego de tener las mediciones periódicamente, se evaluará el algoritmo de caídas junto con las demás interfaces antes mencionadas.

4. FORMACIÓN DE RECURSOS HUMANOS

La presente línea de investigación dentro del departamento de Ingeniería e Investigaciones



Tecnológicas forma parte del trabajo que dos investigadores se encuentran realizando su tesis de maestrías. Completan el grupo de investigación cuatro alumnos de Ingeniería en Informática que se encuentran finalizando su formación de grado y realizan su iniciación a la investigación.

5. BIBLIOGRAFÍA

- [1] Bonilla-Fabela, Morales-Escobar y Guajardo-Muñoz, «IoT, El Internet de las Cosas y la Innovación de sus Aplicaciones,» 2 ed., 2016, pp. 2313-2340.
- [2] STMicroelectronics, «Referece Manual - STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced Arm®-based 32-bit MCUs,» STMicroelectronics, 2018.
- [3] STMicroelectronics, «Datasheet - production data Ultra-low-power 32-bit MCU ARM®-based Cortex®-M3,» de *STM32L151x6/8/B STM32L152x6/8/B*, STMicroelectronics, 2016.
- [4] E. C. M. V. G. D. L. G. G. D. G. R. V. M. V. Waldo Valiente, «Dispositivo de asistencia de personas mediante monitoreo IoT,» SEDICI, San Luis, 2018.
- [5] T. Martin, «The Designer's Guide to the Cortex-M Processor Family,» India, elseiver, 2016, pp. 71-130.
- [6] L. P. L. P. M. T. Fei Guan, «Open source FreeRTOS as a case study in real-time operating system evolution,» de *Journal of Systems and Software*, elseiver, 2016, pp. 19-35.
- [7] A. Mbed, «Mbed OS,» 2018. [En línea]. Available: <https://www.mbed.com/en/platform/mbed-os/>.
- [8] Amazon Web Services, «Amazon FreeRTOS Sistema operativo compatible con IoT para microcontroladores,» 2019. [En línea]. Available: <https://aws.amazon.com/es/freertos/>.
- [9] G. Nutt, «NuttX Real-Time Operating System,» 2019. [En línea]. Available: <https://nuttx.org/>.
- [10] Green Hills Software, «μ-velOSity real-time operating system (RTOS),» 2019. [En línea]. Available: https://www.ghs.com/products/micro_velocity.html.
- [11] On-Line Applications Research (OAR) Corporation, «RTEMS - An Open Real-Time Operating System,» 2019. [En línea]. Available: <http://rtems.com/>.
- [12] G. D. Sirio, «ChibiOS free embedded RTOs - ChibiOS Homepage,» 2017. [En línea]. Available: <http://www.chibios.org/dokuwiki/doku.php>.
- [13] STMicroelectronics, «Datasheet - production data- Medium-density performance line ARM®,» de *STM32F103x8*, STMicroelectronics, 2015.



5. WICC 2019 – Poster del tercer artículo.

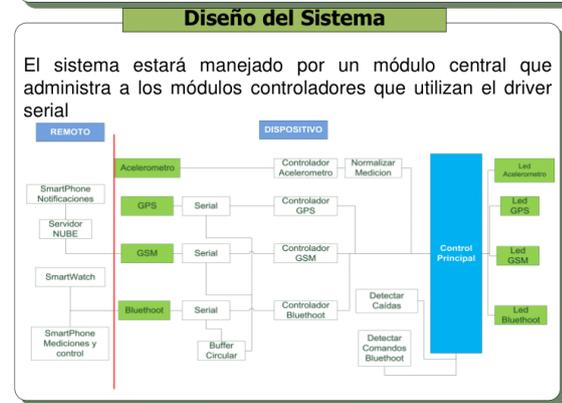
Adaptación de Driver Serial para la placa STM32F103C8T6 y su utilización en controladores de GSM – Bluetooth

Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca, Gerardo Gracia, Raúl Villca, Germán Lorenz, Matías Adagio
Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza

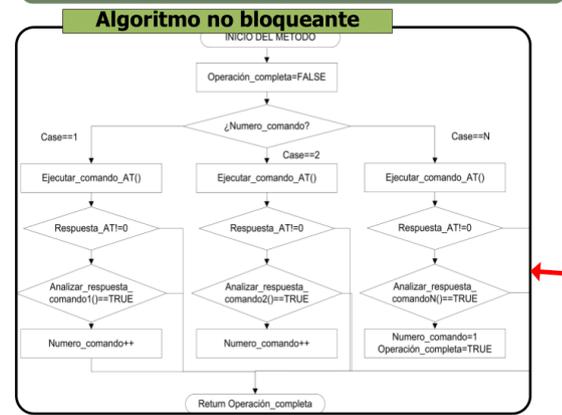
ARSO
Arquitectura, Redes y Sistemas Operativos

- Objetivos**
- Adaptar e implementar un driver serial que permita el intercambio de datos entre la placa STM32F103C8T6 y distintos dispositivos mediante comunicación serie.
 - A través del driver serial, se desarrollará controladores que permitan la utilización de módulos GPS y GSM/GPRS junto con esta plataforma.
 - Los controladores y el driver serial ha desarrollar permitirán construir un dispositivo que ayude a realizar tareas de asistencia remota empleando el concepto de Internet de las Cosas

- Metodología de Programación**
- Se emplean bibliotecas HAL y LL, a través de código en lenguaje C, para la configuración inicial de la placa.
 - Utilizando la programación orientada a objetos mediante C++, se desarrolló el driver serial y los controladores
 - Fue necesario adaptar los parámetros del compilador ARM-NONE-EABI.
- PROGRAMACION HIBRIDA**



- Adaptación del driver serial**
- Se desarrolló un driver genérico, a través de una biblioteca que contiene una clase denominada Serial, que hace uso de tres técnicas de comunicación: por polling, interrupciones y DMA. El driver es configurable para poder seleccionar un mecanismo de comunicación.
 - Se debieron readaptar las bibliotecas HAL y LL para acceder a los registros RDR y TDR de las USARTS, realizando operaciones de lectura y escritura en forma ininterrumpida.
 - Se emplearon buffers circulares para amortizar la velocidad de envío y recepción de datos de los registros de las USARTS.
 - Se crearon las APIs send (que apila datos en un buffer circular y posteriormente envía su contenido por el puerto serie mediante interrupciones) y read (recibe los datos por un puerto serie y los almacena en un buffer circular para su posterior procesamiento).
 - El driver puede funcionar de forma no bloqueante, utilizando mecanismos de sincronización inhabilitando interrupciones en determinados momentos. Todo el programa se ejecuta en un único hilo de ejecución dado que no se esta utilizando multiprogramación



- Utilización del driver serial**
- La clase Control Principal utiliza tres instancias de la clase Serial: uno para Bluetooth, otro para GSM/GPRS y otro para el GPS.
 - El controlador Bluetooth permite controlar y monitorear el embebedo desde un Smartphone
 - El controlador GSM realiza operaciones en el módulo SIM800L, ejecutando una serie de comandos AT, cada uno de ellos realiza una patrón ejecución-respuesta.
 - Estas operaciones se ejecutan mediante un algoritmo no bloqueante, sin usar HAL_delay, a diferencia de otras implementaciones que utilizan S.O de tiempo real para ello.

- Líneas de investigación**
- Emplear el driver serial para la utilización del modulo GPS a través de su controlador.
 - Integrar el dispositivo embebedo para funcionar en conjunto con un servidor en la nube por medio del controlador GSM
 - Integrar los distintos controladores de sistema para que funcionen coordinadamente

- Formacion de Recursos Humanos**
- La presente línea de investigación forma parte del trabajo que se encuentra realizando dos integrantes del grupo de investigación para su maestría.
 - Completan el grupo de investigación cuatro alumnos de Ingeniería en Informática que se encuentran finalizando su formación de grado se encuentran comenzando su carrera como investigador.

- Resultados Obtenidos / Esperados**
- Se ha conseguido desarrollar un módulo que permite realizar una comunicación serial con distintos dispositivos que emplean dicho protocolo, adaptando las bibliotecas HAL y LL
 - Se han desarrollado algoritmos que permiten controlar y monitorear la ejecución del dispositivo IoT, durante la etapa de desarrollo en forma remota por medio del Bluetooth
 - Se han creado APIs que permiten ejecutar funcionalidades del módulo GSM como: Realizar llamadas telefónicas, enviar SMS y mensajes POST a un servidor, de forma no bloqueante en un único hilo de ejecución.



6. WICC 2019 – Paper del tercer artículo.

Adaptación de Driver Serial para la placa STM32F103C8T6 y su utilización en controladores de GSM – Bluetooth

Waldo Valiente, Esteban Carnuccio, Mariano Volker, Graciela De Luca, Gerardo García, Raúl Villca, Germán Lorenz, Matias Adagio

*Departamento de Ingeniería e Investigaciones Tecnológicas
Universidad Nacional de La Matanza*

*Dirección: Florencio Varela 1703 – CP 1754 –
{wvaliente, ecarnuccio, mvolker, gdeluca, ggarcia} @unlam.edu.ar,
{raul.villcasd, germangelv, mati.adagio} @gmail.com*

RESUMEN

En la actualidad la gran mayoría de los proyectos orientados a Internet de las Cosas, utilizan sistemas embebidos en donde resulta necesario emplear comunicación serial para comunicar la plataforma seleccionada con los distintos componentes de hardware que conforman el dispositivo IoT. Existen diversas cantidades de sensores, actuadores y mecanismos de comunicación que emplean el protocolo serial para su correcto funcionamiento. Las bases del sistema embebido que se está realizando en esta investigación se sustentan en este principio. En consecuencia, el presente trabajo expone de qué manera se desarrolló un módulo serial que permitiese utilizar la plataforma STM32F103C8T6 con dispositivos de este tipo. A partir de la utilización de este módulo, se explica de qué forma se generó un Controlador GSM, que permite manejar el SIM800L a través de comandos AT, de forma no bloqueante en un único hilo de ejecución. De la misma manera se detalla cómo se desarrolló un Controlador Bluetooth, que otorga la posibilidad de monitorear y utilizar en forma remota el dispositivo IoT durante la etapa de desarrollo. Por ese motivo en el presente trabajo se explican los métodos empleados en dicha

construcción, detallando el algoritmo utilizado.

Palabras clave: *STM32, bibliotecas HAL, Driver Serial, SIM800L, HM-10.*

CONTEXTO

Nuestra Línea de Investigación es parte del proyecto *Dispositivo de asistencia de personas mediante monitoreo y análisis de datos en la nube*, dependiente de la Unidad Académica del *Departamento de Ingeniería e Investigaciones Tecnológicas*, perteneciente al programa de Investigaciones CYTMA2 de la Universidad Nacional de La Matanza, el cual está formado por docentes, investigadores y alumnos de las carreras de ingeniería en informática e ingeniería en electrónica. Este proyecto es continuación de los trabajos que viene realizando el grupo de investigación, en sistemas operativos, computación de alto rendimiento y en Internet de las cosas.

1. INTRODUCCIÓN

Esta investigación pretende desarrollar un dispositivo que ayude a realizar esta tarea de asistencia remota empleando el concepto de Internet de las Cosas, de acuerdo a lo expuesto en [1]. Implementando un dispositivo IoT que permita la asistencia de personas mediante monitoreo remoto. Debido a que el sistema embebido IoT, debe ser fácilmente portable para



que el usuario lo pueda utilizar en su vida cotidiana, el tamaño del dispositivo y su consumo energético son factores preponderantes. En consecuencia a partir de un extenso estudio para poder cumplimentar estos requisitos de hardware adecuadamente, se ha seleccionado la arquitectura Cortex-M3, a través de la utilización de la placa STM32F10C8T6, también conocida como Blue Pill. [2]. El dispositivo IoT que se está desarrollando utilizará distintos sensores, actuadores y mecanismos de comunicación. Entre los principales sensores y actuadores que se están utilizando se encuentran: el acelerómetro MPU6050 y el GPS C3-470. No obstante una de las partes elementales del dispositivo IoT es la parte comunicacional con componentes externos; tales como un servidor en la nube, computadoras y otros dispositivos Bluetooth. Para ello se está utilizando el SIM800L V2, para la comunicación GSM/GPRS con un servidor, además del Bluetooth 4.0 HM-10. La característica destacable de estos componentes es que son dispositivos seriales, por lo tanto las comunicaciones deben ser realizadas a través del protocolo serial. En consecuencia el software, que se está desarrollando para funcionar en el embebido, deberá emplear un driver que entienda dicha comunicación.

2. LINEAS DE INVESTIGACIÓN Y DESARROLLO

La placa Blue Pill puede ser programada de dos formas distintas: la primera utilizando programas Arduino, empleando el lenguaje Wiring. Este lenguaje anexa una compleja capa de abstracción que afecta la performance del sistema. Por ese motivo se determinó que era conveniente emplear la segunda metodología, que consiste en realizar la programación de estas placas utilizando el lenguaje C y C++, y emplear las bibliotecas HAL y LL [3], que

ofrecen la familia de los microcontroladores STM32, las cuales permiten un mayor control de los recursos del hardware y son autogeneradas por la herramienta STM32Cube-MX, que permite configurar la Blue Pill fácilmente. También el código fuente que autogenera el STM32Cube-MX, es creado en lenguaje C únicamente, por lo que por defecto, los proyectos deben ser construidos utilizando ese lenguaje de programación. No obstante, debido a los beneficios que ofrece la programación orientada a objetos con C++ [4] [5], se determinó conveniente programar utilizando también este último lenguaje. Por consiguiente el código del programa embebido está siendo desarrollado en forma híbrida, parte de la configuración inicial de la placa es realizada en C, a través de las bibliotecas HAL, y por otro lado el resto del programa embebido es desarrollado en C++. Adaptando el código fuente que autogenera el STM32Cube-MX y las opciones del compilador arm-none-eabi, exclusivo para la arquitectura ARM Cortex [6], para que el código fuente pueda ser desarrollado utilizando C y C++.

Diseño del Sistema

El diseño de los módulos del programa que se ejecutará en el dispositivo IoT como se muestra en la Fig. 1. El sistema es dirigido por un Controlador Principal, el cuál determina las acciones que deberán realizar los distintos componentes de hardware que estarán conectados a la Blue Pill. A su vez como una acción de un componente hardware conlleva distintas tareas complejas, ejecutando un conjunto de instrucciones, se determinó conveniente que estas sean realizadas por los demás Controladores para mayor simplicidad. Para el manejo de los dispositivos seriales se creó el módulo *Serial* encargado de dicha tarea.

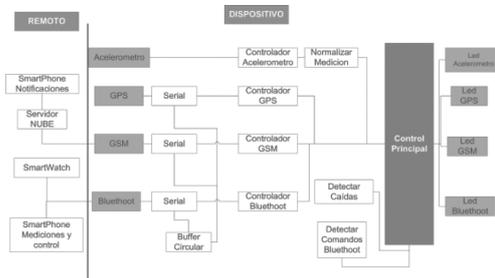


Fig. 2 Diseño de los módulos que conforman el dispositivo IoT

Adaptación de bibliotecas para el Driver Serial

El desarrollo del software necesario para poder comunicar de manera eficiente el Bluetooth, GPS y GSM/GPRS con la placa STM32., presenta tres mecanismos distintos para leer o escribir datos. Estas operaciones se realizan utilizando determinados registros de acuerdo al tipo de periférico con que se esté trabajando. El primer mecanismo consiste en la lectura y escritura de forma bloqueante por medio de la técnica de polling, el segundo en el uso de Interrupciones y tercer método utiliza transferencia de datos a través de DMA [7]. Haciendo uso de estas técnicas se desarrolló un driver genérico, que permite realizar la transferencia de datos de forma serial. Esto se efectuó de esta forma debido a que tanto el módulo de bluetooth, como el de GPS y el de GSM/GPRS realizan la transferencia de datos en forma serial mediante los puertos USART¹. El dispositivo STM32F103C8T6 presenta tres puertos USART independientes, que son accesibles a través de sus pines de conexión. Para poder manipular los registros “Receive Data Register” (RDR) y “Transmit Data Register” (TDR) de la USART [7], se utilizaron parte de las bibliotecas HAL y LL configuradas con el software STM32Cube-MX, que debieron ser adaptadas en su funcionamiento,

¹ Universal Receptor Transmisor Sincrónico/Asincrónico

complementándolas por medio de la creación del módulo Serial (Fig. 2) para poder realizar la lectura y escritura continua de datos a través de la USART. Esto se debió a que las funciones de HAL y LL no están adaptadas para realizar dichas operaciones en forma ininterrumpida con la Blue Pill. Por lo que no se lograba completar la totalidad de las funcionalidades que estas otorgan. El módulo serial fue creado de manera que pueda ser configurado dinámicamente, permitiendo seleccionar el mecanismo de operación con el que se desea trabajar los puertos series: ya sea mediante polling, interrupciones o DMA. Una vez escogido el tipo de operación el driver realiza la transferencia de datos desde la USART al dispositivo a través de instrucciones que envían datos al registro TDR del puerto serie. Dado el tamaño limitado del registro, no es inmediato el envío de los bytes de las tramas al dispositivo serial destino. Por ese motivo fue necesario crear un buffer circular de transmisión que amortiguase los datos que se envían al dispositivo serial, hasta que el TDR se encuentre disponible para poder enviar esos bytes de información y otro buffer circular de recepción, con la finalidad de evitar pérdida de información en los datos que envían los dispositivos seriales a la Blue Pill. Estos datos son recibidos en su registro RDR.

El módulo Serial desarrollado está compuesto por una biblioteca que contiene una Clase denominada *Serial*, cuyos métodos pueden ser invocados desde cualquier parte del código fuente del programa embebido con tan solo crear un objeto de dicha clase. Por consiguiente, al iniciar el Controlador Principal se debieron instanciar tres objetos Seriales: el primero de ellos para emplear el GSM, el segundo para el Bluetooth y el tercero para poder acceder al GPS. Luego estos tres objetos son utilizados internamente por los módulos Controladores para acceder a los respectivos dispositivos. Los



métodos más importantes de la Clase Serial son: *read* y *send*. De acuerdo al método de operación con el cual haya sido creado el objeto, el método *read* recibe los bytes que son enviados al puerto serial correspondiente y los retorna a su invocador. A su vez este método puede ser ejecutado de forma bloqueante o no, leyendo los bytes que fueron almacenados en el buffer circular de recepción correspondiente a dicho a objeto. Por otro lado el método *send*, permite enviar cadenas de bytes al puerto serial correspondiente, también de forma bloqueante o no bloqueante. En este último caso, cada vez que sea invocado el método *send*, se irán apilando en un buffer circular las cadenas que se desean enviar, que serán luego transmitidas cuando se libere el registro TDR por medio de un manejador de interrupciones. Para poder enviar varios parámetros al puerto serial, el método *send* utiliza las macros *va_start* y *va_end*, que son utilizadas en ANSI C para acceder a múltiples parámetros. Además para evitar problemas de sincronismo con los buffers circulares, fue necesario deshabilitar y habilitar las interrupciones en ciertas secciones del código fuente del programa. Esto se debe a que la programación del sistema embebido está siendo realizada sin multiprogramación, ejecutándose todo el programa en un único hilo de ejecución.

Utilización del Driver Serial

Los módulos de hardware GSM SIM800L y el Bluetooth HM-10 son dispositivos que deben emplear el módulo serial. Dentro del código fuente de sus Controladores se realiza los envíos de datos y comandos al componente, invocando para ello el método *send* del objeto que se encuentra asociado al mismo. Mientras que por otro lado la recepción de datos y comandos recibidos por estos componentes, es realizado a través del método *read*, para posteriormente

poder analizar la información recibida y en base a ella actuar en consecuencia. Los comandos recibidos y enviados por el Controlador Bluetooth, permiten controlar y monitorear de forma remota el dispositivo IoT a través de un Smartphone, durante la etapa de desarrollo. En las etapas subsiguientes se pretende adaptar este Controlador para que pueda conectarse a otros sensores Bluetooth. Por otro lado el SIM800L funciona mediante el envío y ejecución de secuencias de comandos AT. Este componente, actúa con un patrón de ejecución-respuesta, el SIM800L ejecuta un comando AT y retorna una respuesta de confirmación de ejecución correcta. Por ese motivo la ejecución del Controlador GSM no puede ser bloqueante, aplicando las funciones *HAL_Delay*, debido a que se produciría una espera activa que afectaría el desempeño del sistema. Para resolver esto, muchas aplicaciones de STM32 hacen uso de paralelismo empleando un S.O. de Tiempo Real [8] [9]. No obstante estas soluciones no son óptimas en cuanto a rendimiento, debido a que ocupan demasiado espacio en memoria, entre otros factores. Por lo cual se desarrolló una solución no bloqueante, que se pueda ejecutar en un único hilo de ejecución, aplicando retardos no bloqueantes a través de contadores de tiempos de reloj, por medio de la función *HAL_GetTicks*. Aplicando dicho concepto, se desarrolló el funcionamiento del algoritmo que ejecutan los distintos comandos en los distintos Controladores. Para esto, se implementaron la codificación de diagramas de estados para la ejecución de los distintos comandos que conforman una operación (Fig. 3). Un ejemplo de ello es el caso del ControladorGSM, que para poder enviar un SMS a un número telefónico, debe ejecutar una de secuencia de tres comandos AT. En la figura Fig. 3 se muestra el algoritmo básico que realiza dicha operación. Al ser no bloqueante, este método debe ser

ejecutado constantemente dentro del bucle del Controlador Principal.

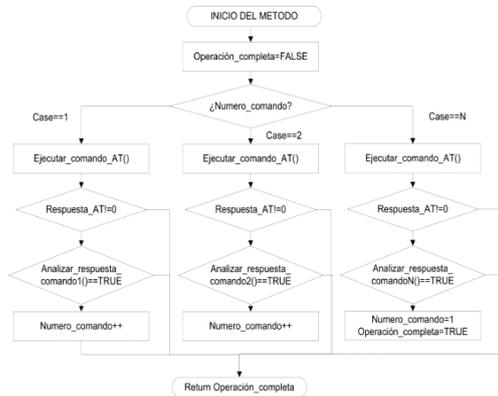


Fig. 3 Diagrama de flujo del Algoritmo básico para ejecutar un comando AT en forma no bloqueante

Un aspecto importante, es que dentro de la función `Ejecutar_comando_AT` se invoca al método `send`, para enviarle al SIM800L el comando AT que deberá ejecutar. Posteriormente en dicha sección de código, se realiza una espera sin detener el transcurso normal del programa, para poder recibir la respuesta del comando AT previamente ejecutado, utilizando el método `read`.

3. RESULTADOS OBTENIDOS/ESPERADOS

Se ha conseguido desarrollar un módulo que permite realizar una comunicación serial con distintos dispositivos que emplean dicho protocolo, adaptando las bibliotecas HAL y LL, que genera el STM32Cube-MX. De esta forma se han desarrollado algoritmos que permiten controlar y monitorear la ejecución del dispositivo IoT, durante la etapa de desarrollo en forma remota por medio del Bluetooth. Al mismo tiempo se han creado APIS que permiten ejecutar funcionalidades del módulo GSM como: Realizar llamadas telefónicas, enviar SMS y mensajes POST a un servidor, de forma no bloqueante en un único hilo de ejecución.

4. FORMACIÓN DE RECURSOS HUMANOS

La presente línea de investigación dentro del Departamento de Ingeniería e Investigaciones Tecnológicas forma parte del trabajo que dos investigadores se encuentran realizando para su tesis de maestrías. Completan el grupo de investigación cuatro alumnos de Ingeniería que se encuentran finalizando su formación de grado y realizan su iniciación a la investigación.

5. BIBLIOGRAFÍA

- [1] G. De Luca, E. Carnuccio, G. García, D. Guilianelli, M. Volker, W. Valiente, V. Raul y M. Vittorio, «Dispositivo de asistencia de personas mediante monitoreo IoT,» de *Libro de Actas Wicc 2018*, Abril 2018, pp. 854-858.
- [2] STMicroelectronics, «Specifications STM32F103C8,» [En línea]. Available: <https://www.st.com/en/microcontrollers/stm32f103c8.html>.
- [3] STMicroelectronics, Description of STM32F1 HAL and Low-layer drivers, 2017.
- [4] S. Dan, «C++ for Embedded C Programers,» 2013. [En línea]. Available: <http://www.dansaks.com/talks/ESC-205.pdf>.
- [5] upwork, 2017. [En línea]. Available: <https://www.upwork.com/hiring/development/c-vs-c-plus-plus/>.
- [6] ARM, ARM Compiler User Guide, vol. Version 6.7, Cambridge, 2017.
- [7] STMicroelectronics, de *STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced Arm®-based 32-bit MCUs*, 2018.
- [8] N. Askari, 2018. [En línea]. Available: https://github.com/nimaltd/Sim800_V2.
- [9] T. Majerle, 2018. [En línea]. Available: https://github.com/MaJerle/GSM_AT_Lib.