Universidad ORT Uruguay

Facultad de Ingeniería

Especificación y verificación de marcapasos: un método de modelado utilizando grafos temporizados

Entregado como requisito para la obtención del título de
Master en Ingeniería (por investigación)

Pablo Damonte - 163012

Tutor: Álvaro Tasistro

2015

Declaración de Autoría

Yo, Pablo Damonte, declaro que el trabajo que se presenta en esa obra es de mi propia mano. Puedo asegurar que:

- La obra fue producida en su totalidad mientras realizaba el Proyecto de Tesis de la Maestría en Ingeniería (por investigación);

- Cuando he consultado el trabajo publicado por otros, lo he atribuido con claridad;

- Cuando he citado obras de otros, he indicado las fuentes. Con excepción de estas citas, la obra es enteramente mía;

- En la obra, he acusado recibo de las ayudas recibidas;

- Cuando la obra se basa en trabajo realizado conjuntamente con otros, he explicado claramente qué fue contribuido por otros, y qué fue contribuido por mi;

- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Pablo Damonte
12/11/2014

Agradecimientos

Quisiera agradecer a Álvaro Tasistro, por su infinita paciencia, constante apoyo y motivación; a Cristina Cornes, por estar siempre dispuesta a ayudarme a entender cómo funcionan los marcapasos y por el interés que demostró en mi trabajo. En ambos casos, mi sincero agradecimiento, ya que hicieron posible la realización de este trabajo.

Un agradecimiento muy especial para mi esposa Fiorella, ya que gracias a su apoyo y sacrificio, hizo posible que dispusiera del tiempo necesario para realizar la maestría. Finalmente, agradezco a quienes perdieron muchos momentos con su padre, a quienes son lo más importante, a mis hijos Emanuel y Julieta.

Resumen

Los sistemas de tiempo real en general requieren que su funcionamiento carezca de fallas; en particular en el caso de controladores de marcapasos resulta algo vital. En este sentido, y en el marco del desafío propuesto por el fabricante de marcapasos Boston Scientific, en este trabajo se plantea una metodología lo más didáctica posible que permita obtener una correcta especificación de los sistemas y detectar fallas en su diseño en las etapas más tempranas. Para que un sistema no falle, es necesario tener certeza de que su especificación es correcta, para luego poder decir lo mismo de su diseño y finalmente de su implementación. De ahí que luego de analizar diversos trabajos que intentan resolver este problema, en particular los presentados en [1], se propone una notación gráfica, extensión de grafos temporizados con facilidades de abstracción, para la representación de los diferentes modos de operación de un marcapasos y se presenta una metodología con un enfoque pedagógico para la utilización de la misma. Se propone entonces un modelo de corazón que permite reflejar diferentes anomalías de su funcionamiento real, analizándose su interacción con los diferentes modos de operación de marcapasos modelados. Los modelos se representan mediante el sistema UPPAAL [2]. Posteriormente, usando el chequeador de modelos de UPPAAL se verifica un número de propiedades consideradas básicas para el correcto funcionamiento de los modelos, e.g. el sistema no se bloquea, no se detectan pulsos en los períodos refractarios, no transcurre más de un tiempo prefijado entre dos estimulaciones o pulsos intrínsecos consecutivos, etc. Adicionalmente se comparan los resultados obtenidos con los presentados por Jiang et al. [19] en un trabajo anterior, concluyendo que el enfoque propuesto en este trabajo resulta ser más completo y permite su comprensión por parte de un público no tan entendido en el área. Finalmente se considera la posibilidad de continuar la línea de investigación y llegar a la generación del código del marcapasos o ampliar la línea de investigación y generalizar el método propuesto para contemplar otro tipo de dispositivos médicos implantables.

Palabras Clave

UPPAAL, Chequeador de Modelos, Especificación Formal, Marcapasos, Boston Scientific

Índice

1 Introducción

Boston Scientific, una empresa dedicada a la fabricación de marcapasos, ha lanzado al dominio público la especificación de un sistema de marcapasos de una generación anterior con el fin de que sirva como base para un desafío para la comunidad de métodos formales. En dicho contexto, el Laboratorio de Investigación en Calidad del Software (SQRL) del Departamento de Computación y Software de la Universidad McMaster, Canadá, oficia de anfitrión y administrador del desafío "PACEMAKER Formal Methods Challenge". El desafío tiene múltiples dimensiones y niveles. Se puede optar por presentar una versión completa del software de marcapasos, diseñado para ejecutarse en un microcontrolador PIC específico (arquitectura RISC), presentar sólo los documentos de requerimientos formales, o algo intermedio. El marcapasos es un ejemplo de sistema crítico, un sistema del cual se requieren fuertes garantías de conformidad con sus requerimientos.

En febrero de 2014 en Dagstuhl, Alemanía se celebró un seminario [1] donde se abordaron temas relativos al desafío del marcapasos en particular, como al desarrollo certificable de dispositivos médicos en general. En el informe correspondiente se presenta un resumen del estado del arte en lo que respecta al desafío y se remarca su importancia en cuanto a estimular la investigación en el área y a los aportes que éste ha generado. El hecho de que muy pocos trabajos pueden aplicarse a la industria es considerado como uno de los principales problemas que aquejan al área. Esto se atribuye en algunos casos a lo poco práctico de los enfoques utilizados, y en otros, a las dificultades que se encuentran a la hora de transferir el conocimiento técnico correspondiente, provocando que las personas que finalmente trabajan en el diseño de los dispositivos desconozcan los métodos y herramientas que pueden ser de mayor utilidad para el diseño e implementación de los mismos. Finalmente, se analiza cómo los métodos de desarrollo basados en modelos pueden ayudar a la construcción y certificación de dispositivos médicos implantables en forma segura.

1.1 Motivación y objetivos

Se plantean varios problemas interesantes, por un lado el hecho de tratarse de un problema real, combinando métodos formales con la medicina y el desarrollo de software. Además, sirve como ejemplo práctico de casos en los que la tolerancia a fallos es muy baja (o nula) y es vital garantizar dicha cualidad, mostrando cómo los métodos formales pueden ser de mucha utilidad para dicho fin.

Por otro lado y según lo que se desprende del seminario de Dagstuhl, existe una gran necesidad de poder transferir el conocimiento del área, de forma tal que las personas que diseñan y/o desarrollan los dispositivos (ejemplo, ingenieros de software) sean capaces de llevar a la práctica los nuevos métodos o la utilización de nuevas herramientas. En este sentido se intenta impulsar la utilización del desarrollo basado en modelos, lo que facilitaría lo antes mencionado.

Sería de esperar que el presente trabajo contribuya a la profundización en el estudio y aplicación de métodos formales para especificar problemas, lo que permitiría a posteriori demostrar que su implementación está libre de errores. Adicionalmente podría generar nuevos lazos de comunicación con la industria, ya sea para la formalización de especificaciones, o para la aplicación de técnicas conducentes a demostrar que un programa cumple con su especificación.

En conclusión, en el presente trabajo se realiza un estudio completo del estado del arte en lo que a especificación formal del software de un marcapasos se refiere, se presenta una especificación formal del marcapasos lo más realista posible (algo que resulta de utilidad para la industria) dejando por fuera conceptos que se resuelven a nivel de micro hardware (cómo se detecta un pulso, cómo se realiza una estimulación, etc.). Finalmente se desarrolla una metodología de una forma didáctica que permite a los diseñadores y/o desarrolladores de marcapasos modelar los mismos utilizando grafos temporizados (lo que favorece su transferencia). Mediante la utilización de una notación basada en la de UPPAAL [2] y utilizando una técnica de refinamientos sucesivos se realizan los modelos, los cuales son posteriormente llevados a UPPAAL para verificar mediante el chequeo de modelos que satisfacen ciertas propiedades, como ser por ejemplo:

- no transcurre más de un tiempo preestablecido entre dos pulsos consecutivos (ya sea intrínsecos o artificiales),
- el tiempo entre dos pulsos artificiales es menor o igual al tiempo preestablecido para dicho fin,
- el marcapasos no se bloquea.

1.2 Corazón y Marcapasos

Con el objetivo de tener una idea precisa del problema que se intenta resolver es necesaria una breve introducción a ciertos conceptos considerados básicos para la eficaz comprensión del mismo. Para ello es necesario comprender cómo funciona el corazón humano, cuáles son algunas de sus afecciones y qué papel juega en su tratamiento la utilización de un marcapasos.

1.2.1 Corazón

Como se muestra en la Figura 1-1, el corazón humano se encuentra dividido en cuatro cámaras, las aurículas izquierda y derecha y los ventrículos izquierdo y derecho. El corazón tiene un marcapasos natural, el Nodo Sinusal, que genera los pulsos eléctricos (éstos, luego de un lapso de tiempo son amplificados por el Nodo Auriculoventricular) .

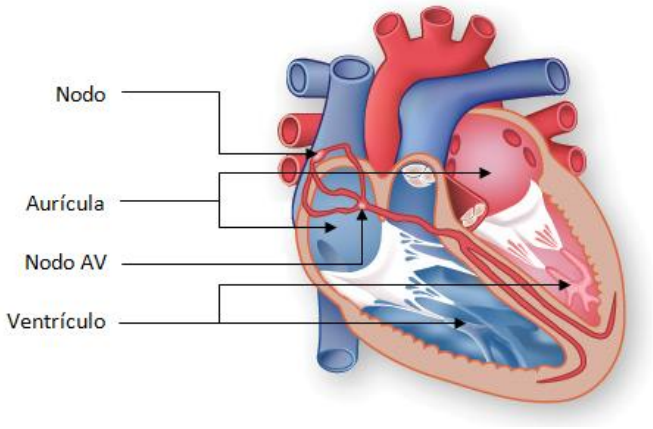

Figura 1-1

La bradicardia [3], del griego "corazón lento", es una afección cardíaca que implica, por ejemplo, que el nodo sinusal emita menos de 60 pulsaciones por minuto (ppm) o directamente su falta de funcionamiento, o que emita pulsos muy débiles, los cuales no llegan a estimular al nodo auriculoventricular. Cualquiera sea la causa de la bradicardia, es necesaria la implantación de un marcapasos artificial que ayude o sustituya el funcionamiento del marcapasos natural.

1.2.2 Marcapasos

Un marcapasos es un pequeño dispositivo electrónico capaz de compensar deficiencias del corazón humano. Se implanta en el interior del pecho mediante cirugía y se conecta al corazón mediante uno o dos conductores. Éstos permiten estimular y detectar los impulsos del corazón, pudiendo estar conectados al ventrículo, aurícula o a ambos (ver Figura 1-2), en función de las necesidades del paciente.

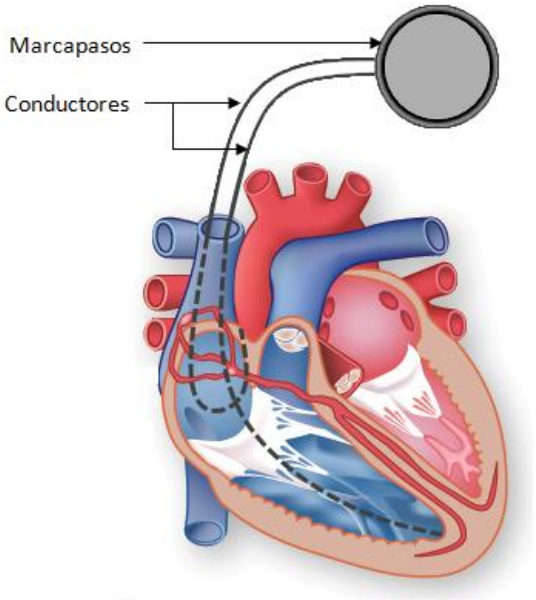

Figura 1-2

El marcapasos tiene básicamente dos modos de operación:

- Modo permanente. Su operación principal es la terapia bradicardia, detectando pulsos cardíacos intrínsecos y/o estimulando con pulsos artificiales.

- Modo temporal. Permite probar las funcionalidades del marcapasos y emitir reportes durante la terapia de bradicardia.

El marcapasos es capaz de entregar pulsos de acuerdo a un conjunto de parámetros programados por el cardiólogo durante el implante. Estos parámetros se relacionan con la frecuencia de los impulsos, el voltaje y el tipo de respuesta a la detección de los latidos. Entre estos parámetros se encuentra el modo de operación de bradicardia (bo mode) que describe cómo la terapia se comportará respecto a:

- la cámara a estimular, pudiendo ser: la aurícula, el ventrículo o ambas.

- la cámara a monitorear, pudiendo ser también en este caso: la aurícula, el ventrículo o ambas.

- el comportamiento ante la detección de pulsos intrínsecos al corazón (auriculares o ventriculares), indicando si se debe o no reforzar dicho pulso con la generación de uno artificial.

- si el marcapasos hace uso o no de un acelerómetro para incrementar la frecuencia en la entrega de impulsos de acuerdo a la intensidad del movimiento del cuerpo.

La NASPE (North American Society of Pacing and Electrophysiology) y el BPEG (British Pacing and Electrophysiology Group) utilizan un sistema de códigos para describir los diferentes modos de operación de un marcapasos. A continuación se presenta una tabla describiendo dicha nomenclatura:

Posición	I	II	III	IV
Categoría	Cámara Estimulada	Cámara Censada	Respuesta a la detección	Modulación de frecuencia (opcional)
Letras Usadas	O – Ninguna A – Aurícula V – Ventrículo D – Dual (A+V)	O – Ninguna A – Aurícula V – Ventrículo D – Dual (A+V)	O – Ninguna T – Activada I – Inhibida D – (T+I)	O – Ninguna R – Modulación de frecuencia

Tabla 1-1

Ejemplos:

- modo VOO es un modo de operación de bradicardia, en el que los pulsos son enviados al ventrículo (lo cual es simbolizado por la letra 'V'), los pulsos no son detectados por el marcapasos (lo cual es simbolizado por la primer 'O') y no hay respuesta a los eventos de detección (lo cual es simbolizado por el segundo 'O').

  En dicho modo el marcapasos puede enviar pulsos al corazón con una periodicidad programada por el cardiólogo, independientemente de cualquier actividad eléctrica del corazón.

- modo VVT, al igual que en el ejemplo anterior, estimula al ventrículo (lo cual es simbolizado por la primer letra 'V'), se censa al ventrículo (lo cual es simbolizado por la segunda letra 'V') y cuando se detecta un pulso ventricular realiza un estímulo al ventrículo (lo cual es simbolizado por la letra 'T').

Entre los modos de funcionamiento temporal del marcapasos, se encuentran unos de particular importancia, los modos magneto. El marcapasos dispone de un detector de magnetos y cuando éste detecta un magneto se cambia el modo de funcionamiento original del marcapasos a un nuevo modo con el objetivo informar al médico cuál es el estado de batería del marcapasos. La elección de este nuevo modo se toma en función del modo original del marcapasos con las siguientes consideraciones:

- si el modo de operación original es VXXX se pasa a un modo VOO magneto,
- si el modo de operación original es AXXX se pasa a un modo AOO magneto,
- finalmente, si el modo original es DXXX se pasa a un modo DOO magneto.

Los modos XOO magneto funcionan en esencia de la misma forma que los modos XOO estándares. Se diferencian en la frecuencia con la que estimulan: en lugar de ser fija como en los originales, en los modos magneto la frecuencia varía en función del nivel de batería del marcapasos. Luego, al dejar de detectarse la presencia de un magneto se vuelve al modo de funcionamiento original. De esta forma, y sin necesidad de artefactos especiales, el médico puede determinar el nivel de batería del marcapasos simplemente acercando un magneto al mismo y contando la cantidad de pulsos cardíacos en un determinado período de tiempo.

1.3 Organización del documento

El presente documento se estructura en seis capítulos además del presente. En el capítulo 2 se presenta el estudio del estado del arte en el desafío del marcapasos, mientras que en el capítulo 3 se presenta la especificación técnica informal del marcapasos de Boston Scientific y se definen algunos términos técnicos. Posteriormente, en el capítulo 4 se presentan los grafos temporizados y chequeadores de

modelos, mientras que los modelos propuestos se introducen en detalle en el capítulo 5. Luego en el capítulo 6 se presenta tanto el modelo en UPPAAL como las propiedades que se verificaron, para seguidamente en el capítulo 7 presentar las conclusiones y trabajos futuros.

2 El estado del arte del desafío del marcapasos

Existen diversos trabajos que de una u otra forma intentan resolver, o por lo menos dar un acercamiento, al problema en cuestión. Debido a la heterogeneidad y complejidad de los diferentes enfoques resulta imposible poder dar una comparación detallada de éstos o analizarlos en profundidad, ya que en algunos casos ello requiere conocimientos muy específicos y experiencia más amplia de la que se tiene al momento. Por esto, a continuación se presenta una búsqueda en amplitud de los diferentes trabajos relacionados.

Gomes y de Oliveira presentan trabajos [4] [5] que se basan en una especificación informal de un modelo viejo de marcapasos de Boston Scientific. Se usa el lenguaje Z para dar una especificación formal del mismo y se ve al marcapasos como un generador de pulsos. Presentan únicamente la especificación del modo de funcionamiento VOO, quedando algunos aspectos de la misma poco claros. Además se discute cómo se utilizó el demostrador de teoremas ProofPower-Z para probar que la especificación del sistema es consistente, aunque no se demuestra que se verifiquen propiedades que aseguren el correcto funcionamiento del marcapasos. Ante la imposibilidad de adquirir el hardware que permite ejecutar el código generado para el marcapasos, luego de evaluar diferentes alternativas, se muestra una traducción del modelo hecho en Z a Perfect Developer. Usando esta herramienta, es posible verificar el código traducido de forma completamente automática, refinando la especificación verificada en lenguajes de programación como Java, C#, C++ o ADA. En este trabajo se indica que la tarea de realizar una especificación formal del marcapasos es compleja: las dificultades van desde el hecho de que la especificación informal no da suficiente información (incluso algunas funcionalidades no pueden modelarse por falta de información, como ser el algoritmo de adaptación de frecuencia), hasta que el equipo de desarrollo del sistema de marcapasos carece de conocimientos previos en el área.

En el trabajo de Méry y Singh [6] se presenta el modelado basado en eventos de un marcapasos unicameral con todos sus modos posibles de funcionamiento, lo cual es una simplificación del problema ya que no se consideran modos complejos como los bicamerales. Se comienza con un modelo abstracto del sistema y luego mediante una serie de refinamientos sucesivos se incluye información más detallada acerca de los diferentes modos de funcionamiento del marcapasos, como ser: propiedades funcionales, especificaciones relativas al transcurso del tiempo, eventos más concretos, etc. Cada modelo es expresado utilizando el lenguaje de modelado Event-B y mediante la herramienta de demostración RODIN se verifica que los diferentes refinamientos son comportamientos del modelo abstracto, lo cual brinda en forma semiautomática una demostración formal de su especificación. Sin embargo, debido a que se manejan una serie de suposiciones respecto al funcionamiento del marcapasos, se requiere validar en forma separada las mismas con el fin de garantizar un funcionamiento seguro. Finalmente, mediante el uso de la herramienta ProB se prueba el funcionamiento de los diferentes modos de operación del marcapasos respecto a restricciones de tiempo real de los mismos en diferentes situaciones. Por ejemplo: adaptación de la frecuencia de la estimulación o histéresis de frecuencia, lo que consiste en permitir frecuencias

intrínsecas por debajo de la frecuencia mínima programada con el objetivo de maximizar los beneficios al paciente.

Larson, Chalin y Hatcliff [7] toman como casos de estudio sólo algunos modos de operación del marcapasos. Parten del lenguaje de análisis y diseño de arquitectura AADL, el cual es ampliamente utilizado en la industria para modelar el software y hardware de sistemas embebidos de tiempo real. AADL es un lenguaje de modelado orientado a componentes donde se hace énfasis principalmente en las interfaces de los diferentes componentes; no proporciona una semántica, ni algoritmos de verificación básicos que permitan razonar cómo es la composición de los mismos o si se comportan en forma correcta. Para resolver este problema se define un nuevo lenguaje sobre el comportamiento del software de sistemas embebidos (BLESS por sus siglas en inglés), el cual extiende AADL mediante una serie de notaciones que permiten describir el comportamiento del software. Finalmente, se implementa un demostrador BLESS que en forma semiautomática permite afirmar el correcto comportamiento de los diferentes componentes.

Sun et al. [8] propone un lenguaje CSP con estado temporizado y un enfoque automatizado para la verificación de los modelos en dicho lenguaje. Este lenguaje se basa en CSP temporizado [9] y es capaz de especificar sistemas jerárquicos de tiempo real. Para esto se hace uso de zonas dinámicas de abstracción, las cuales permiten generar en forma automática zonas de grafos de estado finito a partir de modelos expresados en el lenguaje propuesto. El trabajo se centra principalmente en la presentación del lenguaje, dando sólo una especificación del modo de funcionamiento AAT a modo de ejemplo, no dando mucho detalle sobre cómo se llega a la misma. Por último se extiende el chequeador de modelo PAT (Process Analysis Toolkit) para soportar el modelado del sistema y su verificación utilizando el lenguaje CSP con estado temporizado.

Tuan y ChunZheng [10] debido a la falta de precisión de la especificación que brinda el desafío, hacen foco principalmente en la estimulación y detección de pulsos cardíacos, no considerando comportamientos más complejos, como ser, por ejemplo, el algoritmo de adaptación de frecuencia. A partir de la especificación informal se obtiene un modelo formal utilizando el álgebra de procesos; con este fin es que se modela al marcapasos como diferentes procesos paralelos, donde cada uno de estos se corresponde con diferentes componentes del marcapasos o el corazón. Este modelado se realiza utilizando RTS (Sistema de Tiempo Real), el cual es en esencia CSP temporizado con extensiones para el manejo de variables, eventos, operaciones, etc. Las diferentes propiedades críticas que debe cumplir el marcapasos fueron expresadas utilizando fórmulas de la lógica temporal lineal (LTL) y refinamientos de tiempo, las cuales finalmente fueron verificadas mediante la utilización de una versión modificada del chequeador de modelos PAT que permite la verificación de refinamientos de tiempo.

En cambio, Bessling y Huhn [11] proponen la utilización de SCADE Suite [12] para el modelado de marcapasos, Dicha herramienta es un ambiente de desarrollo basado en modelos dedicado especialmente al software embebido de sistemas críticos. Para esto se especifica la línea de producción y sus productos utilizando el Lenguaje Común de

Variabilidad (CVL por sus siglas en inglés [13]). Éste separa el modelado del producto de la variabilidad en la modelización utilizados para diseñar los diferentes elementos base del mismo. Implica una estricta estructuración del modelado de productos con el fin de reflejar los conceptos de sustitución utilizados para describir la variabilidad. A partir del modelado y mediante el uso de SCADE Design Verifier se verifican algunas propiedades críticas del marcapasos realizando una búsqueda de todas las posibles combinaciones de valores de entrada y caminos posibles en las cuales la propiedad puede satisfacerse. Adicionalmente, mediante la utilización de un generador de código C certificado que posee SCADE Suite se genera el código que controla la lógica del marcapasos.

Dos Santos Macedo et al. [14] estudian la posibilidad de utilizar metodologías de desarrollo ágiles y rigurosas para sistemas distribuidos de tiempo real. Es en este contexto que se analiza la viabilidad de utilizar el Método de Desarrollo de Viena (VDM) [15] para especificar y analizar el modelo en etapas tempranas del diseño, tomándose como caso de estudio el problema del Marcapasos, contemplando solo siete modos de funcionamiento del mismo. En este trabajo, se genera un modelo resumido del marcapasos (VDM-SL) donde queda claro cómo se llega a cada componente y qué función cumple cada uno. Este modelo es utilizado inicialmente para comprender el problema y posteriormente sirve de base para modelos donde se presentan diferentes alternativas para el modelado del tiempo, como ser: VDM++ secuencial, en el cual mediante un mecanismo de pasos se incrementa el tiempo y realizan las diferentes operaciones; VDM++ concurrente, donde mediante la utilización de eventos y el patrón Espera-Notificación se maneja el transcurso del tiempo y disparan las operaciones; DR-T, el cual es un modelo de tiempo real distribuido, modelando al tiempo mediante la herramienta VICE. Finalmente mediante la utilización de la herramienta VDMTools se testea el modelo concurrente en diferentes escenarios.

La Manna, Motta y Bonanno [16] proponen la utilización del lenguaje de especificación de sistemas TRIO+ (extensión de TRIO [17] con conceptos de clases) para expresar una especificación formal del marcapasos, presentando como casos de estudio los modos de funcionamiento AAI, VVT y DDD. TRIO (Tempo Reale ImplicitO) es un lenguaje formal y un método para la especificación, análisis y verificación de sistemas de tiempo real críticos y es el resultado de extender a la lógica temporal de primer orden con nociones de tiempo lineal y características típicas de la orientación a objetos. Se parte de una especificación formal inicial, y mediante la introducción de axiomas se refina dicha especificación hasta llegar a una definitiva. En cada paso de este proceso cíclico y para verificar que la especificación continúa siendo correcta, se utilizan el chequeador de modelos ZOT y el demostrador de teoremas PVS. La especificación presentada utiliza un enfoque modular donde se ve marcapasos como un generador de pulsos, dejando dentro de dicho modulo especificaciones asociadas a los tres modos de funcionamiento presentados, algo que dificulta analizar cada modo de funcionamiento por separado, por ejemplo, cuando se quiere transmitir cómo funciona cada modo. Con esta especificación formal se procederá posteriormente a diseñar e implementar el sistema y se generaran los diferentes casos de prueba del mismo.

Como en la actualidad no existe una metodología formal o plataforma experimental que sea abierta para validar y verificar el correcto funcionamiento del software de

dispositivos médicos implantables, Jiang et al. [18] plantean un modelo de corazón virtual (VHM por sus siglas en inglés). Éste modela el funcionamiento electrofisiológico del corazón, tanto en los casos en que funciona correctamente (es decir, durante el ritmo sinusal normal), como en los casos de mal funcionamiento (durante arritmias). Se presenta para ello una metodología que permite extraer propiedades temporizadas del corazón y construir un modelo de autómatas temporizados. Posteriormente se demuestra que el VHM es capaz de generar una respuesta clínicamente relevante a señales intrínsecas (es decir, estímulos prematuros) y externas (es decir, marcapasos artificial) para una variedad de arritmias comunes. Finalmente, se prueba que conectando el VHM con un modelo de marcapasos se es capaz de estimular y sincronizar al corazón durante el inicio de ritmos irregulares del corazón.

Finalmente, en un trabajo presentado por Jiang et al. [19] se modela el funcionamiento del modo DDD de un marcapasos en UPPAAL y se formalizan diferentes propiedades de seguridad básica que debe cumplir el marcapasos. En un trabajo posterior, Chen, Diciolla, Kwiatkowska y Mereacre [20] basándose en el trabajo de Jiang et al. desarrollan una metodología para la verificación cuantitativa automatizada de modelos de software de marcapasos. Por tratarse de los trabajos más similares al que se está presentando, los mismos se describen con mayor detalle al comienzo de capítulo 5.

3 Sobre la especificación técnica informal y términos básicos

El marcapasos de Boston Scientific, como se puede ver en su especificación [21], consta de diferentes modos de operación y funcionalidades. La especificación que se brinda, además de ser muy informal, resulta ser muy vaga en cuanto a cómo funcionan sus diferentes modos de operación. Su principal énfasis está en la especificación de los diferentes parámetros que deben configurarse, sus rangos de valores posibles, así como también los diferentes valores estadísticos que registra. Presenta un nivel de detalle que parece estar orientado a médicos o técnicos que trabajan normalmente con este tipo de dispositivos, dando muchos comportamientos o restricciones por sobreentendidas u obvias, como ser, por ejemplo, el funcionamiento del corazón.

Dado que con este nivel de detalle se hace imposible realizar una especificación formal, y mucho menos una validación de la misma, es que se consultó con entendidos en el área para lograr tener un acercamiento al problema que se intenta modelar. Al mismo tiempo se realizaron innumerables búsquedas de información en la Web, comprendiendo: sitios especializados en medicina, fabricantes de marcapasos, congresos médicos, etc.

De todo esto se pudo concluir que existen variadas implementaciones de marcapasos y que difieren esencialmente en la eficiencia energética, el tamaño de los mismos, la implementación de algoritmos avanzados que permiten detectar y prevenir en forma anticipada tanto fallas del propio marcapasos como diferentes afecciones del paciente. Pero en sus funciones más estandarizadas el funcionamiento de todos ellos es prácticamente igual, por lo que es posible completar varias zonas grises de la especificación brindada por Boston Scientific con la información extra recabada. En cambio no se pudo conocer de qué forma y en qué momento son almacenados los datos registrados por el marcapasos, cómo es su posterior procesamiento y cómo son implementados los diferentes algoritmos por Boston Scientific. Como ser, por ejemplo, el algoritmo de adaptación de frecuencia utilizado en los modos XXXR, donde se desconoce cómo se calcula la frecuencia y cuáles son todos los parámetros de entrada que necesita.

Existen además diferentes intervalos o períodos de tiempo que resultan básicos para el funcionamiento del marcapasos. Algunos de ellos son:

- Intervalo de Frecuencia Inferior (LRI - Low Rate Interval). Es el intervalo de tiempo más largo que puede existir entre un pulso ventricular (detectado o estimulado) y el siguiente pulso ventricular estimulado. Esto garantiza una frecuencia cardiaca mínima al paciente.

- Intervalo de Frecuencia Superior (URI - Upper Rate Interval). Es el intervalo de tiempo más pequeño que puede existir entre dos pulsos ventriculares consecutivos.

- Período Refractario Ventricular (PVR - Post Ventricular Period). Intervalo de tiempo que comienza cuando se detecta un pulso ventricular (detectado o estimulado) y durante el cual no se detectan pulsos para evitar la detección del propio estímulo o la despolarización del ventrículo.

- Período Refractario Auricular (PAR - Post Auricular Period). Intervalo de tiempo que comienza cuando se detecta un pulso auricular (detectado o estimulado) y durante el cual no se detectan pulsos para evitar la detección del propio estímulo o la despolarización de la aurícula.

- Intervalo Auriculoventricular (AVI - Atrioventricular Interval). Intervalo de tiempo comprendido entre un pulso auricular (detectado o estimulado) y una estimulación ventricular.

- Período Refractario Auricular Post Ventricular (PVARP - Post Ventricular Atrial Refractory Period). Es el intervalo de tiempo que comienza luego de un pulso ventricular y durante el cual se evita la detección de pulsos auriculares.

- Período Refractario Auricular Total (TARP - Total Atrial Refractory Period). Es la suma del AVI y el PVARP, constituye un intervalo básico para limitar la máxima frecuencia de estimulación del marcapasos.

A medida que transcurre el tiempo de uso de un marcapasos el nivel de batería del mismo decrece. Según este nivel, es posible que se desactiven algunas funcionalidades del marcapasos o se cambie el modo de funcionamiento del mismo con el fin de prolongar la vida de la batería y asegurar la salud del paciente. Para hacer esto posible se manejan diferentes estados relativos al nivel de batería, que se detallan a continuación:

- Inicio De Vida (BOL - Beginning Of Life). Representa el estado esperado de la batería en el momento del implante.

- Cerca del Reemplazo Electivo (ERN - Elective Replacement Near). Representa el estado de la batería cuando resta aproximadamente menos de un año para coordinar el reemplazo de la batería.

- Tiempo de Reemplazo Electivo (ERT - Elective Replacement Time). Estado de la batería en el cuál debe programarse el reemplazo del marcapasos.

- Fin De Vida (EOL - End Of Life). Representa el estado de la batería aproximadamente 3 meses después del ERT; a partir de este momento no se puede garantizar el correcto funcionamiento del marcapasos.

En la especificación técnica brindada por Boston Scientific podemos encontrar una serie de parámetros programables que pueden clasificarse en dos grandes tipos:

- Parámetros asociados a la interacción del marcapasos con el medio exterior que son resueltos por medio de micro hardware, como ser por ejemplo el voltaje, duración o intensidad de la estimulación, el voltaje que debe medirse para considerar que se detectó un pulso, etc.

- Parámetros que indican cómo debe ser el comportamiento del marcapasos, como ser: el tiempo que debe transcurrir entre dos estimulaciones, cuál es el período ventricular refractario, etc.

En este trabajo no se analizan los conceptos asociados al primer grupo de parámetros, considerándose por ejemplo sólo el hecho de enviar un estímulo al ventrículo, sin considerar otros criterios, como por ejemplo, su voltaje, intensidad o duración. Por consiguiente, en la Tabla 3-1 pueden observarse cuáles son los parámetros a tomar en cuenta, cuál es su rango de valores válidos y cuál es su valor nominal.

Parámetro	Valor Programable	Valor Nominal
Modo	DDD, VDD, DDI, DOO, AOO, AAI, VOO, VVI, AAT, VVT, DDDR, VDDR, DDIR, DOOR, AOOR, AAIR, VOOR, VVIR	DDD
Intervalo AV Fijo	70-300 ms	150 ms
Período Refractario Ventricular	150-500 ms	320 ms
Período Refractario Auricular	150-500 ms	250 ms
Período Refractario Auricular Post Ventricular	150-500 ms	250 ms
Intervalo de Frecuencia Superior	50-175 ppm	120 ppm
Intervalo de Frecuencia Inferior	30-175 ppm	60 ppm

Tabla 3-1

4 Grafos Temporizados y Chequeadores de Modelos

4.1 Grafos Temporizados

En los años 90, Alur y Dill [22] desarrollan el concepto de autómatas temporizados. Un grafo o autómata temporizado es una máquina de estados finita, extendida con variables de tipo reloj. En la teoría usa un modelo de tiempo donde las variables de relojes toman valores de los números reales y donde todos los relojes progresan sincrónicamente. En este contexto, un sistema es modelado como una red de varios autómatas temporizados en paralelo. El modelo es extendido con variables discretas acotadas, las cuales son parte del estado; estas variables pueden ser usadas como en los lenguajes de programación, i.e. pueden leerse, escribirse y pueden aplicárseles operaciones comunes de la aritmética.

En este trabajo se formula y utiliza una versión extendida de los grafos temporizados que permite reutilizar grafos preexistentes en el diseño de nuevos grafos (macros), lo que permite proceder por refinamientos sucesivos con una representación gráfica simple que facilita su comprensión.

Todo autómata tiene aristas o transiciones, las cuales pueden actuar separadamente o en forma sincronizada con otros autómatas y conducen a nuevos estados. Cada arista se conforma de una guarda, un canal de sincronización y una acción, pudiendo cualquiera de ellas omitirse en caso de no ser requerida. Donde:

- **Guarda**. Es una condición necesaria (no suficiente) para realizar la transición. Una guarda es una expresión particular que satisface la siguiente semántica: debe cumplirse que es libre de efectos laterales; evalúa a un booleano; solo pueden referenciarse relojes, variables enteras y constantes (o arreglos de dichos tipos); relojes y diferencias de relojes pueden compararse solo con expresiones enteras; guardas sobre relojes son esencialmente conjunciones (disyunciones son permitidas sobre condiciones de enteros). Una guarda puede llamar a una función libre de efectos laterales que retorne un booleano, a pesar de que las restricciones sobre relojes no son soportadas en dichas funciones.

- **Sincronizaciones**. Para que diferentes subsistemas logren comunicarse entre sí se utilizan canales de sincronización. Una etiqueta de sincronización es de la forma: Expresión!, Expresión? o es una etiqueta vacía. La expresión debe ser libre de efectos laterales, evalúa a un canal y solo hace referencia a enteros, constantes o canales. Un canal puede ser de alguno de los siguiente tres tipos:

    - **Canales de sincronización binaria**, donde una arista etiquetada con c! sincroniza con otra etiquetada con c?, produciéndose comunicación. En el caso que varias combinaciones se encuentren habilitadas, la pareja que efectivamente sincroniza es elegida de forma no determinística. Este tipo de canales puede llevar al sistema a un estado inválido en caso de que no

haya una posible pareja con quien sincronizar. Es decir, un subsistema no puede transicionar por una arista etiquetada con un canal c?, si no existe otro subsistema en el que sea posible transicionar por una arista etiquetada con un canal c! en forma simultánea.

- **Canales urgentes**. Se diferencian de los anteriores en el hecho de que no es posible mantenerse en un estado si en éste existe una sincronización a través de un canal urgente. Además, su utilización no es posible en conjunto con condiciones sobre relojes en la guarda de la transición. Para diferenciarlos de los canales binarios, el nombre del canal debe terminar en "_U".

- **Canales de difusión**. Son una extensión de los binarios, donde un emisor c! puede sincronizarse con un número arbitrario de receptores c?. Todo receptor que pueda sincronizarse en su estado actual lo hará y en caso de que no los haya, entonces el emisor puede continuar ejecutando la acción c!. Por esta razón este tipo de canales nunca lleva al sistema a un estado inválido. Para diferenciarlos gráficamente de los otros dos tipos de canales, el nombre de los mismos debe terminar en "_D".

- **Actualización**. Una etiqueta de actualización es una lista separada por "," de expresiones libre de efectos laterales, dichas expresiones solo puede referenciar relojes, variables enteras, y constantes y solo asigna valores enteros a los relojes. Pueden llamar también a funciones.

Una ubicación (o nodo del grafo) puede contar con un nombre y un invariante. Un invariante es una expresión que debe satisfacerse siempre que se esté en la ubicación y puede implicar que no sea posible llegar a dicha ubicación o que se deba salir de la misma si el invariante deja de cumplirse. Adicionalmente un invariante satisface las siguiente condiciones: es libre de efectos laterales; solo relojes, variables enteras, y constantes son referenciadas. Es una conjunción de condiciones de la forma x<e o x<=e donde x es una referencia de reloj y e evalúa a un entero. Un invariante puede llamar a una función libre de efectos laterales que retorna un booleano, a pesar de que las restricciones sobre relojes no son soportadas en dichas funciones. A su vez, hay tres tipos especiales de ubicación:

- **Ubicaciones iniciales.** Cada grafo debe tener exactamente una ubicación inicial. La ubicación inicial es marcada con un doble círculo.

- **Ubicaciones urgentes**. En este tipo de ubicaciones no se puede permanecer, ya que no se permite que el tiempo transcurra mientras un proceso está en una ubicación urgente. Se marcan con una letra U y semánticamente son equivalentes a:

  - añadir un reloj extra x, que se reinicia en cada transición entrante, y

  - añadir un invariante x<=0 a la ubicación.

- **Ubicaciones comprometidas.** Al igual que las ubicaciones urgentes no permiten que el tiempo transcurra y se las marca con la letra C. Por otra parte, si cualquier proceso se encuentra en un lugar comprometido, la siguiente transición debe implicar la salida de una de las ubicaciones comprometidas. Son útiles para la codificación de sincronización entre más de dos componentes o para la creación de secuencias atómicas. Tener en cuenta que, si varios procesos se encuentran en una ubicación comprometida al mismo tiempo, entonces va a tomarse una transición de salida de una de esas ubicaciones comprometidas (elegida no determinísticamente).

  En la Figura 4-1 puede observarse un ejemplo donde el sistema E debe comunicarse con los sistemas R1 y R2 en forma simultánea y utilizando para dicho fin los canales m1 y m2 respectivamente. La particularidad de la ubicación E2 es que ha sido declarada como comprometida, lo que implica que en la misma no se permite el transcurso del tiempo, por lo que desde un punto de vista temporal es "como" si m1 y m2 fueran simultáneos.

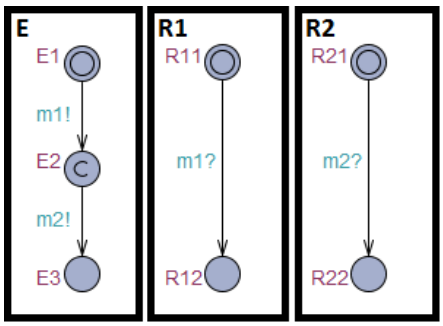

Figura 4-1

Para facilitar la comprensión de la representación gráfica utilizada y con el objetivo de evitar malas interpretaciones o lecturas, se utilizan diferentes colores para representar a las guardas, actualizaciones, canales de sincronización, invariantes, etc., la cual puede verse en la Figura 4-2.

Figura 4-2

A continuación se pueden encontrar ejemplos de especificaciones de sistemas que permiten clarificar los conceptos antes mencionados.

Ejemplo 1:

Una lámpara con tres estados: apagada, encendida y brillante. Si el usuario presiona un botón, entonces la lámpara se enciende. Si el usuario presiona el botón nuevamente, la lámpara se apaga. Además, si el usuario es rápido y presiona rápidamente el botón dos veces, la lámpara se enciende y se vuelve brillante. El usuario puede presionar en forma aleatoria el botón en cualquier momento o no presionarlo nunca.

El reloj de la lámpara es usado para detectar si el usuario fue rápido (t < 5) o lento (t >= 5).

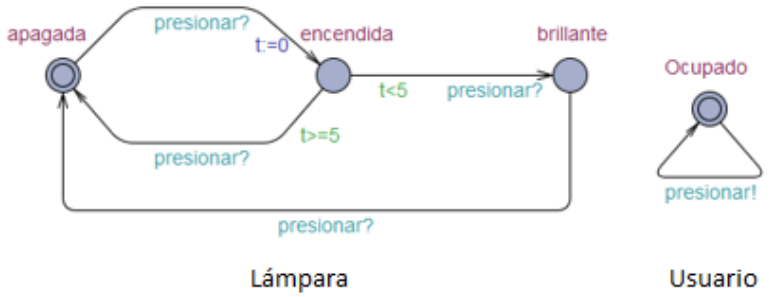

Figura 4-3

Como puede verse en la Figura 4-3 se modelaron en forma independiente al usuario y a la lámpara. La acción de presionar el botón por parte del usuario es modelada enviando un mensaje por el canal presionar. Por otra parte, cada posible estado de la lámpara se modela utilizando una ubicación: la lámpara comienza estando apagada, si se recibe un mensaje presionar por parte del usuario se pasa a la ubicación encendida y se comienza a controlar el tiempo mediante el reloj t, cuando se recibe un nuevo mensaje presionar, si el reloj t marca 5 segundos o más se pasa a la ubicación apagada. En caso de haber transcurrido menos de 5 segundos se pasa a la ubicación brillante hasta recibir nuevamente un mensaje presionar, lo que lleva a la ubicación apagada.

Ejemplo 2:

Una puerta automática con sensor de movimiento. La puerta tiene dos estados: abierta o cerrada. Si el usuario se mueve dentro del alcance del sensor, la puerta se abre. Y se cierra en forma automática luego de 10 segundos de la última vez que detectó el movimiento del usuario. El usuario puede moverse en forma aleatoria dentro del alcance del sensor o no moverse nunca.

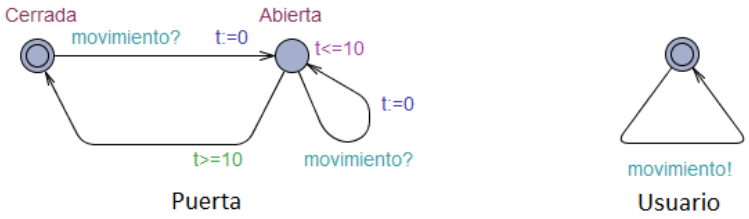

Figura 4-4

Como se puede observar en la Figura 4-4 cada vez que el usuario se mueve envía un mensaje movimiento; por otro lado, se modelan los dos posibles estados de la puerta con dos ubicaciones (Cerrada y Abierta). La puerta comienza Cerrada, si se recibe un mensaje movimiento se pasa a la ubicación Abierta y se comienza a controlar el tiempo utilizando el reloj t, en dicha ubicación el reloj t no puede marcar más de 10 segundos y cada vez que se recibe un mensaje movimiento se vuelve a cero el reloj t. En caso que haya pasado más de 10 segundos (t>=10) se vuelve a la ubicación Cerrada.

El hecho de cumplirse la guarda de una arista no implica necesariamente que se elija la misma. Por lo tanto, si se desea obligar a elegirla en un momento preciso, se la utiliza en conjunto con invariantes. En el ejemplo anterior (Figura 4-4) puede observarse el invariante de la ubicación Abierta (t<=10) utilizado en conjunto con la guarda de la arista que lleva a la ubicación Cerrada (t>=10), lo que da como resultado que dicha arista se utiliza cuando el reloj t marca exactamente 10 segundos. Es importante aclarar que la restricciones sobre el reloj t especificadas en la guarda e invariante deben poder satisfacerse a la vez en por lo menos un instante de tiempo, de ahí que ambas contemplen el caso t=10. Esto se debe a que hasta el momento mismo en que se realiza una transición (implica que se cumple su guarda) se debe permanecer en la ubicación, y por consiguiente debe cumplirse el invariante de la misma.

Existen casos en los que se desea explicitar que un sistema es un refinamiento, o especialización de otro. Se propone para ello utilizar una variante ad-hoc de los grafos temporizados donde se puede referenciar a un grafo preexistente mediante una macro. Esto se representa gráficamente mediante un rectángulo etiquetado con el nombre del grafo original, donde además es posible:

- Agregar una arista entre dos ubicaciones, alcanza con incluir dentro del rectángulo las ubicaciones originales y definir una arista entre ellos.

- Agregar invariantes a ubicaciones preexistentes, se procede de la misma forma que en el caso anterior.

- Realizar una serie de sustituciones al grafo original, las cuales indican a la derecha del rectángulo. Dicha serie es una lista de sustituciones separadas por "," que deben realizarse en forma simultánea, donde cada sustitución es de la forma X<-Y y representa la sustitución de todas las ocurrencias de X por lo especificado en Y.

En el caso de que el grafo donde se está utilizando una macro disponga de una ubicación inicial, ésta prima por sobre la que puedan contener las macros utilizadas, quedando estás últimas sin efecto, considerándoselas ubicaciones normales.

Ejemplo 3:

Considerando nuevamente el escenario del Ejemplo 1, es posible reformular su solución con el objetivo de modelar inicialmente la lámpara sin el modo brillante y concebir a este modo como un posterior refinamiento. En tal sentido, el modelo inicial del sistema es el siguiente:

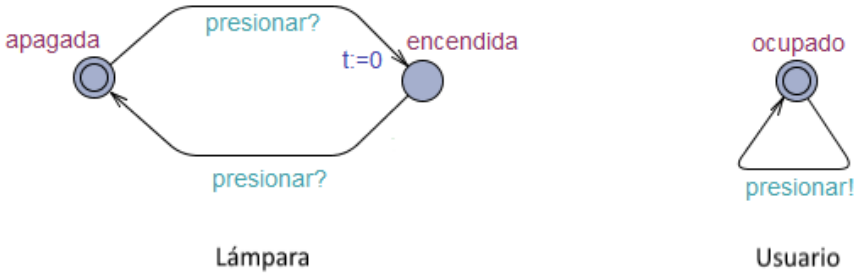

Figura 4-5

Si ahora se refina dicho sistema, considerando además que, si el usuario es rápido y presiona rápidamente el botón dos veces, la lámpara se enciende y se vuelve brillante. El nuevo sistema puede modelarse de la siguiente forma:

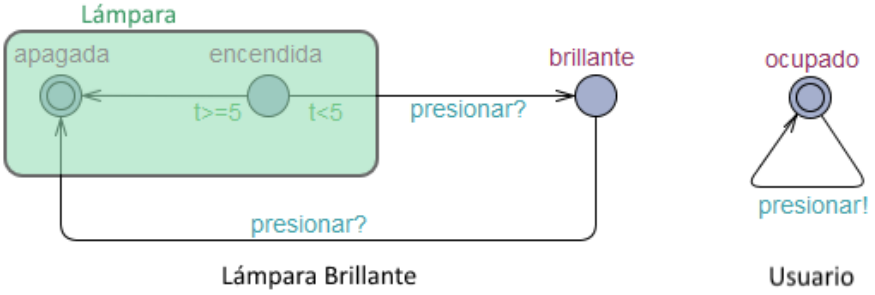

Figura 4-6

En la Figura 4-6 puede observarse cómo se extiende el modelo original de la lámpara con una nueva ubicación para representar el estado brillante de la misma y las transiciones que llevan o salen de ella. De esta forma se logra modelar el mismo escenario que en la Figura 4-3 y se deja en evidencia que la lámpara brillante es una especialización de una lámpara que sólo prende y apaga.

Se manejan además las siguientes características adicionales:

- **Funciones de usuario.** Pueden definirse tanto a nivel global como en forma local a los grafos. La sintaxis es similar a la de C, excepto que no hay punteros. Sintaxis C++ para referencias es soportada sólo para los argumentos.

- **Expresiones:** son rangos sobre relojes y variables enteras, ejemplo, x<e, donde x es un reloj y e una variable entera. Son usadas habitualmente para definir condiciones que deben cumplirse mientras se permanece en un estado o se utiliza una transición (o arista).

Finalmente, un estado del sistema se define por la ubicación de todos los autómatas, los valores de los relojes, y las variables discretas.

4.2 Verificación de Modelos

En los años 80, Clarke y Emerson desarrollan la aplicación de lógica temporal a la verificación de sistemas [23]. La lógica temporal se obtiene de añadir operadores temporales a la lógica de predicados (o de primer orden), presentando por ejemplo, al tiempo como un cambio de estado. Los tipos más comunes de lógica temporal son:

- lineal (LTL), en el cual se razona sobre una única línea temporal, En lo que respecta al análisis de programas, LTL analiza posibles ejecuciones, concibiendo a las mismas como secuencias infinitas de estados, donde la ejecución de una instrucción genera un cambio de estado.

- arborescente (CTL), en la cual se razona sobre todas las líneas temporales posibles. En la década del 90, Alur et al [24] extienden la lógica CTL con operadores temporales cuantitativos, llamando a esta nueva lógica TCTL (Timed CTL).

Como algo complementario al modelado con grafos temporizados, se puede encontrar la verificación de modelos (o model checking) [25]. Este es un mecanismo automático que permite la verificación de sistemas formales, generalmente modelos derivados del hardware o software de un sistema informático. Por un lado, se debe tener un sistema, modelo representado utilizando grafos con transiciones de estado y por otro, una fórmula que represente una especificación formal del sistema, descrita generalmente como una variante de la lógica TCTL. El problema consiste entonces en verificar dicha fórmula en todos los posibles estados del sistema, algo que normalmente resulta inviable debido a que la cantidad de combinaciones de estados posibles de un sistema es al menos de una magnitud exponencial a la cantidad de estados del problema original. El trabajo de Clarke (y luego el de otros investigadores) ha consistido en desarrollar diversos métodos que permiten reducir la cantidad de estados a verificar sin perder por eso representatividad, permitiendo que la verificación de modelos sea algo factible.

5 Modelos

Dentro del ámbito del desafío de Boston Scientific, en un trabajo presentado por Jiang et al. [19] mencionado anteriormente, se modela el funcionamiento de un marcapasos en UPPAAL. En él se analiza sólo el modo de funcionamiento DDD y su modelado se basa principalmente en la representación de los cinco diferentes tiempos que se consideran básicos para este modo de funcionamiento (LRI, URI, AVI, VRP y PVARP). En este modelado, los diferentes parámetros de configuración del marcapasos están fijos y no se representan los modos temporales de operación del marcapasos que se producen ante un bajo nivel de batería o ante la presencia del magneto. En definitiva, sólo se modela el modo DDD básico. El modelado que se realiza del corazón es muy elemental y no permite analizar el comportamiento del marcapasos frente a determinadas afecciones cardíacas. Además, al no modelarse la batería y el magneto, no se verifican aspectos críticos del funcionamiento del marcapasos, como son el hecho de indicar correctamente el nivel de batería o cambiar de modo de funcionamiento ante la presencia de un magneto, etc. Igualmente, se formalizan diferentes propiedades de seguridad básica que debe cumplir el marcapasos, las cuales se obtienen no solo de las especificaciones del desafío, sino que además de conocimiento fisiológico adicional. Se analiza por ejemplo el caso en que una violación de la seguridad en el funcionamiento del marcapasos pueda generar que el sistema degenere en un patrón no deseado de funcionamiento, como ser, taquicardia mediada por el marcapaso (TMM), donde la despolarización de una estimulación del ventrículo se conduce a la aurícula y la misma es detectada como un pulso por el marcapasos, lo que genera a su vez la estimulación ventricular, reiniciándose el circuito. Dado que los marcapasos modernos implementan algoritmos de terminación de TMM que permiten salir de tales condiciones, resulta importante identificar las mismas y chequear el correcto funcionamiento de los algoritmos. Para esto se agregan a los modelos básicos monitores que detectan patrones no deseados.

Este enfoque, si bien permite tener un nivel de aislación muy bueno en lo que respecta al manejo de los diferentes tiempos, no deja ver claramente el funcionamiento del marcapasos en su conjunto, lo que implica conocer de antemano cómo y cuándo interactúan entre sí los diferentes modelos, dificultando el análisis del modelo en tiempo de diseño, y posiblemente la comprensión del mismo por terceros en una posible transferencia de conocimiento. En un trabajo posterior, Chen, Diciolla, Kwiatkowska y Mereacre [20]**¡Error! No se encuentra el origen de la referencia.** basándose en el trabajo de Jiang et al. [19] desarrollan una metodología para la verificación cuantitativa automatizada de modelos de software de marcapasos. Ya que el modelo de corazón presentado originalmente plantea situaciones irreales, se sustituye por uno más realista. Posteriormente, con el fin de obtener el comportamiento conjunto del corazón y marcapasos, se aplican diferentes técnicas de discretización utilizando el chequeador de modelos probabilísticos PRISM. Finalmente, se demuestran algunas propiedades del modelo en MATLAB, como ser por ejemplo que el marcapasos corrige la bradicardia.

Como se mencionó anteriormente, existen partes de la especificación del marcapasos que se desconocen. En este contexto se decidió modelar sólo los modos de operación más estandarizados de los marcapasos, ya que su funcionamiento está claramente definido. Los diferentes modos de funcionamiento del marcapasos que se modelan son:

AOO, AAI, AAT, VOO, VVI, VVT, DOO, DDI, DDD. (ver Tabla 1-1) Por otro lado los modos DDDR, VDDR, DDIR, DOOR, VOOR, AOOR, VVIR y AAIR se comportan de forma similar a los modos modelados, diferenciándose de estos en el hecho de que la frecuencia utilizada, en lugar de ser fija como en los primeros, es dinámica, calculándose en base a la actividad que esté realizando el paciente (por ejemplo, mediante la utilización de un acelerómetro). Al carecer de algún tipo de información al respecto no se efectuará su modelado en este trabajo. Otros modos que tampoco se modelan ya que no resultan interesantes a la hora del chequeo de propiedades (ver sección 6.4 para mas detalles), son: OVO, OAO, ODO y OOO. Esto se debe a que sólo escuchan al ventrículo o aurícula, siendo además modos temporales.

En una primera aproximación a la solución, y con el fin de poder realizar un modelo que permita capturar la esencia del funcionamiento de un marcapasos y su interacción con el corazón es que se manejan diferentes modelos (o subsistemas). Por un lado, se modela un simulador de corazón, la batería o el magneto y por otro, un modelo para cada modo de funcionamiento del marcapasos. Esto último resulta del hecho de que luego de configurado un modo de operación, éste no cambia hasta que el marcapasos es nuevamente configurado e inicializado, lo que puede considerarse que cada modo de operación es en sí mismo un marcapasos.

5.1 Canales

Se dispone de varios canales de sincronización entre los diferentes subsistemas, ellos son:

- VPulso_D y APulso_D. Son utilizados para comunicar al simulador del corazón con el marcapasos y permiten al marcapasos escuchar los “latidos naturales”. Estos canales se definieron como de difusión, dado que permiten modelar de una forma bastante realista el hecho de que un marcapasos puede escuchar o no un latido;

- VEstimulado y AEstimulado. Utilizados para comunicar al marcapasos con el simulador del corazón, permiten estimular el ventrículo o aurícula respectivamente. En este caso los canales se definieron como binarios.

- A la hora de lograr comunicar el magneto con el marcapasos se manejan los canales de difusión MagnetoEntra_D y MagnetoSale_D.

En cambio, la comunicación entre la batería y el marcapasos se realiza mediante una variable compartida que almacena el nivel de batería en cada momento.

5.2 Relojes

Se manejan diversos relojes independientes, algunos necesarios para el correcto funcionamiento del sistema y otros necesarios para posteriormente poder verificar que se cumplen ciertas restricciones. Los diferentes relojes ventriculares manejados son:

- VTiempoArtificial. Utilizado para controlar el tiempo transcurrido entre dos pulsos ventriculares consecutivos generados por el marcapasos;
- VTiempo. Utilizado para controlar el tiempo transcurrido entre dos pulsos ventriculares consecutivos, ya sean artificiales o naturales.

Análogamente, se dispone de los relojes ATiempoArtificial y ATiempo para la aurícula.

5.3 Corazón

El objetivo es lograr obtener un modelo lo suficientemente completo que permita simular un corazón, capturando las diferentes patologías que éste puede presentar y que además sea capaz de interactuar con todos los modos de funcionamiento del marcapasos a ser estudiados. Para lograrlo, se comienza modelando un simulador de corazón inicial básico y luego, mediante una serie de refinamientos sucesivos se complejizará hasta llegar al modelo deseado.

Como punto de partida se considera un modelo del simulador (ver Figura 5-1) en el cual sólo se generan pulsos intrínsecos (o naturales) desde el ventrículo, enviando mensajes por el canal VPulso_D. El tiempo entre pulsos intrínsecos ventriculares se cronometra por intermedio del reloj vt y el período de tiempo entre pulsos está comprendido entre un período mínimo (vPeriodoMinimo) y uno máximo (vPeriodoMaximo). Resulta evidente que vPeriodoMaximo debe valer como mínimo vPeriodoMinimo y por lo tanto si en la ubicación el reloj llega a vPeriodoMaximo entonces la transición necesariamente va a producirse (ya que de otro modo el invariante dejaría de cumplirse).

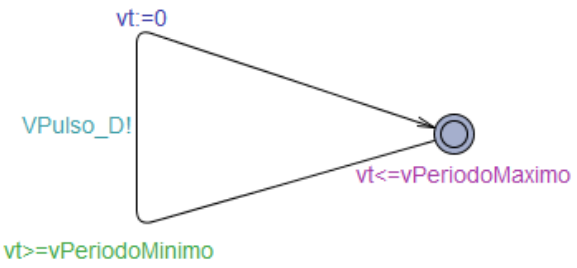

Figura 5-1

Un modelo posterior (ver Figura 5-2), quizás no muy obvio, pero que brindará beneficios al momento de verificar propiedades, consiste en agregar una ubicación comprometida justo al momento de la generación del pulso. Esto permite tener un punto al cual referirse en las propiedades a verificar, por ejemplo, una propiedad que debe cumplirse sólo al momento del pulso ventricular y no necesariamente antes o después.

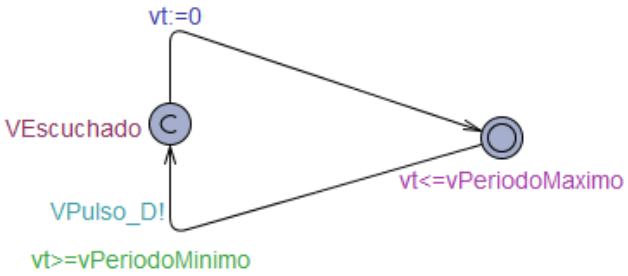

Figura 5-2

Posteriormente se considera el caso en el cual el simulador de corazón además de generar pulsos, es capaz de ser estimulado por el marcapasos al recibir un mensaje por el canal VEstimulado, considerando además (y con el mismo sentido que en el caso anterior) una ubicación comprometida para dicha transición, un reloj (VTiempo) que controla el tiempo entre dos pulsos consecutivos (intrínsecos o artificiales) y un reloj que controla el tiempo entre dos estimulaciones consecutivas (VTiempoArtificial), dando como resultado el siguiente modelo:

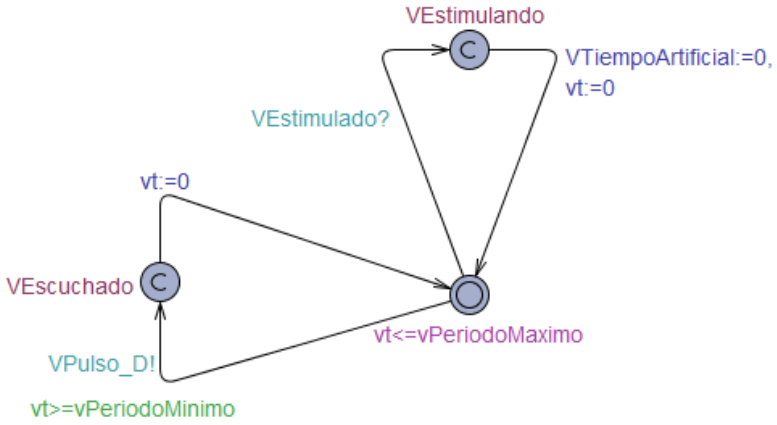

Figura 5-3

Hasta el momento solo se está considerando al ventrículo, por lo que el siguiente paso es pasar a considerar ambas cámaras, por consiguiente, y razonando en forma análoga a lo hecho para el ventrículo, se obtiene el modelo siguiente:

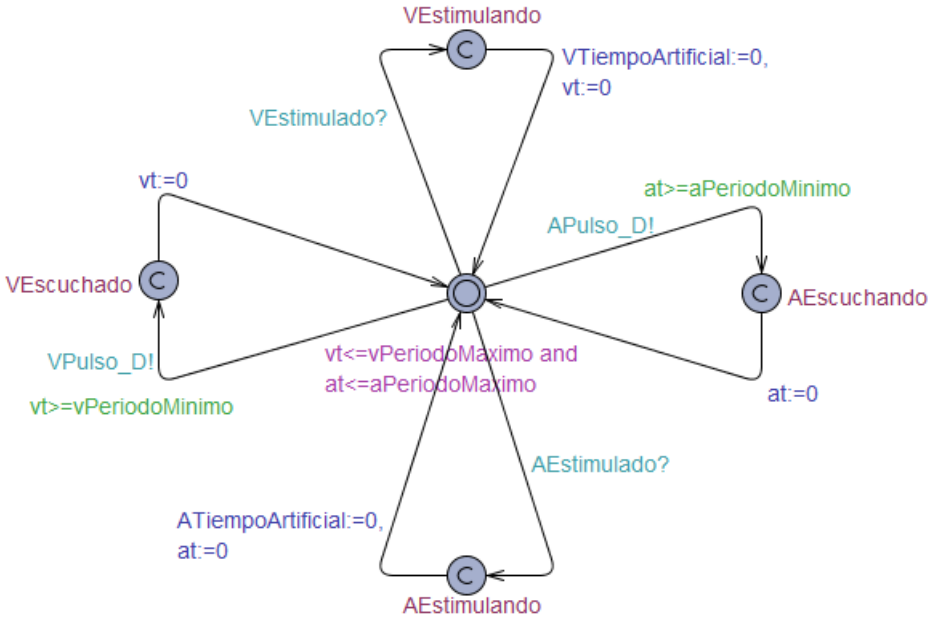

Figura 5-4

En el modelo precedente las pulsos intrínsecos ventriculares no tienen relación con los auriculares. Esto puede resultar útil si se quiere modelar un corazón donde el ventrículo y aurícula no trabajan en forma alternada, como puede ser, por ejemplo, en los casos donde el paciente tiene una patología que hace que el ventrículo tenga pulso intrínsecos cada dos pulsos intrínsecos auriculares. Por tal motivo, es importante no perder dicha forma de funcionamiento, pero también es necesario poder modelar el funcionamiento de un corazón que trabaja en forma normal, donde los pulsos ventriculares y auriculares se producen en forma intercalada. Por todo esto y como último caso, se considera un modelado del simulador de corazón, donde el intercalado es opcional y queda determinado por la variable intercalado, dando como resultado el siguiente modelo:

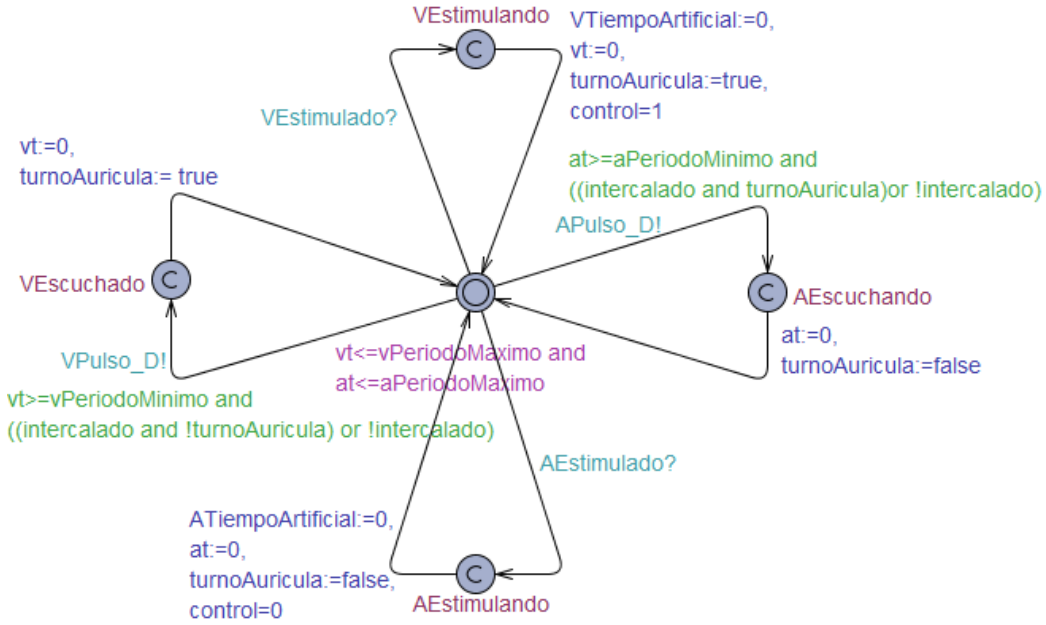

Figura 5-5

Cabe remarcar que una mala configuración de las variables vPeriodoMinimo, vPeriodoMaximo, aPeriodoMinimo, aPeriodoMaximo e intercalado puede provocar luego que no se verifiquen algunas propiedades que interesan ser garantizadas. Por esta razón es importante hacer una configuración a conciencia de sus valores, y evitar simular situaciones inexistentes en la realidad.

5.4 Batería

Existen diversas situaciones en las que se requiere modelar el nivel de carga de la batería; en otros casos, si se está analizando una cierta situación donde el desgaste de la batería resulta despreciable esto no interesa. Como por ejemplo si se está analizando el funcionamiento del marcapasos en un período de pocas horas o minutos, siendo que la batería dura entre 5 y 8 años aproximadamente.

Se consideraron además, por ejemplo, los casos en los que se quiere verificar si efectivamente el marcapasos informa correctamente su nivel de batería, o en el que interesa estudiar el comportamiento del marcapasos cuando la batería se descarga, ya que eso puede provocar el cambio en el modo de funcionamiento del marcapasos.

Con el fin de poder cubrir estos escenarios y como se puede ver en la Figura 5-6, el modelo de la batería maneja una variable eBateria, la cual puede tomar valores entre 0 y 100 y representa el nivel de carga de la batería. Al comienzo de la ejecución del sistema se inicializa el nivel de la batería con el valor estadoInicial. Luego, cada vez que el reloj t cronometra un determinado período de tiempo (indicado por el valor velocidadDescarga) se disminuye en uno el nivel de la batería hasta llegar al valor estadoFinal, a partir del cual la batería deja de disminuir su nivel de carga. En el caso en que velocidadDescarga es cero, el nivel de batería permanece siempre constante.

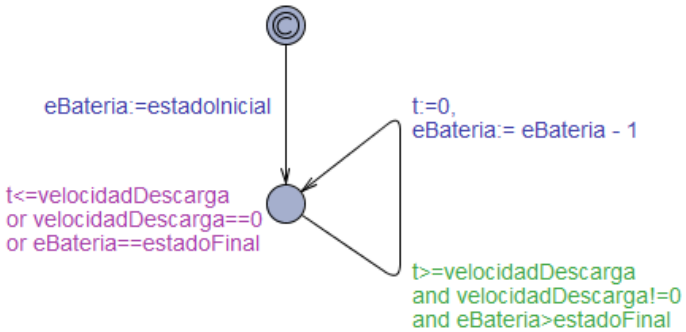

Figura 5-6

5.5 Magneto

Dado que el marcapasos cambia su modo de funcionamiento cuando un magneto se encuentra próximo, es necesario su modelado explícito (ver Figura 5-7), interesando simular su presencia, su ausencia y las transiciones entre ambos. Con el objetivo de

simular estos comportamientos se utilizan un reloj x, el cual controla el tiempo en que el magneto está próximo al marcapasos o se encuentra ausente, los períodos de tiempo que se debe permanecer en cada una de estas situaciones, y los siguientes dos estados:

- SinMagneto. Es el estado inicial y representa la ausencia del magneto. La única forma de salir es si se ha estado en él al menos el tiempo indicado por tiempoEspera. Es decir, ésta es una condición necesaria para salir.

- ConMagneto. Representa la presencia del magneto. En forma análoga al caso anterior, la única forma de salir de la ubicación es haber estado en ella al menos el tiempo indicado por tiempoDuracion.

Cuando se transiciona del estado SinMagneto al ConMagneto se envía un mensaje al marcapasos por el canal de difusión MagnetoEntra_D. De forma similar, al transicionar del estado ConMagneto al SinMagneto un mensaje es enviado al marcapasos por el canal de difusión MagnetoSale_D. En ambos casos, la elección de canales de difusión se debe a que no necesariamente el marcapasos va a escuchar los mensajes, siendo esto, por ejemplo, una propiedad que debería demostrarte posteriormente.

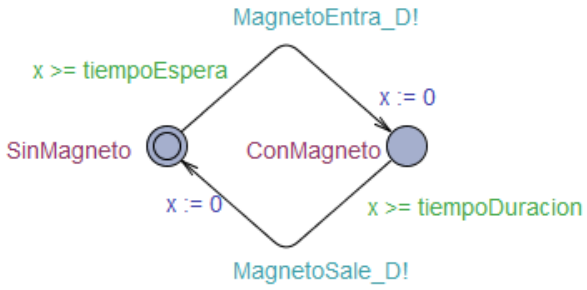

Figura 5-7

5.6 Marcapasos

En sus funciones básicas, es posible afirmar que los diferentes modos de operación de un marcapasos trabajan en forma similar. De ahí que se los pueda clasificar en dos categorías diferentes: los modos unicamerales y los bicamerales.

5.6.1 Marcapasos unicamerales

Este subgrupo incluye a los modos de funcionamiento: VOO, VVI y VVT, así como a sus análogos auriculares AOO, AAI y AAT. Esencialmente, como se puede ver en la Figura 5-8 podemos descomponer su funcionamiento de la siguiente forma:

- Evento: Representa el evento que desencadena la estimulación artificial de la cámara. Este evento para algunos modos puede ser el transcurso de un

determinado período de tiempo desde la última estimulación, para otros casos puede ser por ejemplo que se haya escuchado un pulso en una de las cámaras que se está escuchando.

- Condición: Indica la condición que debe cumplirse para poder realizar efectivamente la estimulación luego de haberse generado el evento, descartando todos los eventos que no cumplan con la misma. La condición puede ser que no se esté dentro de determinado período de tiempo (lo que permite descartar por ejemplo un evento que se produce muy próximo a otro), u otra diferente dependiendo del caso.

- Acción: Indica la acción que debe realizarse luego de realizada la estimulación. La condición puede ser, por ejemplo, volver a cero un reloj.

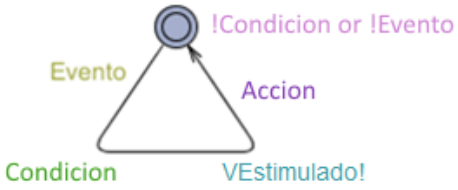

Figura 5-8

Se analizarán sólo los modos de la forma VXX, ya que sus versiones AXX son idénticas y difieren básicamente en dónde se conectan físicamente los electrodos del marcapasos y en el funcionamiento de los mismos.

5.6.1.1 VOO_Base

Este modo de funcionamiento es la versión más simplificada del modo VOO que se puede concebir, el objetivo del mismo es servir de base para futuros refinamientos que permitan obtener otros modos de funcionamiento más avanzados. En este modo de funcionamiento el evento que genera la estimulación es el transcurso de un período de tiempo previamente programado (tiempoCicloFijo). No hay una condición y la acción es volver a cero al reloj tiempo. Como resultado de esto, y considerando además que este modo será reusado posteriormente, se obtiene el siguiente modelo:

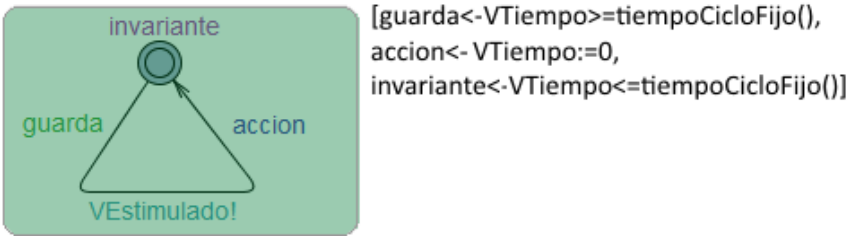

Figura 5-9

5.6.1.2 VOO_Magneto

Este modo corresponde a VOO en presencia de un magneto. No requiere necesariamente ser modelado en forma independiente ya que su funcionamiento puede incluirse dentro de los modos originales. Dado que en todos los modos VXX el comportamiento ante la presencia de un magneto es idéntico, se puede dar enfoque modular a la solución modelándolo por separado. En el mismo, el modo magneto corre en paralelo al modo original y permanece en un estado de latencia hasta que es activado por el modo original, quedando este último suspendido a la espera de que el modo magneto lo reactive y se suspenda a sí mismo, dejando todo como al principio.

Cuando se detecta la presencia del magneto y se está en un modo de funcionamiento de la forma VXX, el mismo se suspende y se pasa al modo denominado VOO_Magneto. Este es un refinamiento del modo VOO_Base donde los pulsos se generan con una frecuencia que depende del nivel de batería del marcapasos (ver Tabla 5-1) y se modelan mediante la función tiempoCicloBateria. Es a su vez un modo temporal, dado que una vez detectado el alejamiento del magneto se vuelve al modo de funcionamiento original. Por consiguiente, el modo VOO_Base (ver Figura 5-9) resulta alterado para dar como resultado el siguiente modelo:

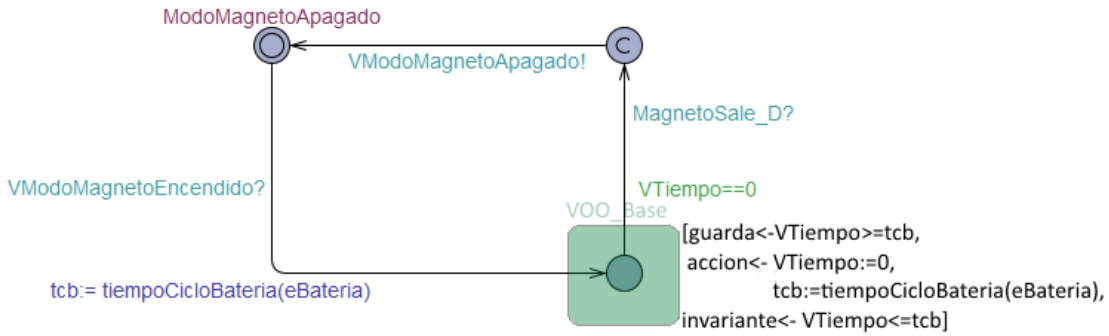

Figura 5-10

Nivel de Batería	Frecuencia Generación Pulsos (ppm)
Mayor a ERN	100
Entre ERN y ERT	90
Entre ERT y EOL	85
Menor a EOL	<=85

Tabla 5-1

Inicialmente está en un estado de latencia, esperando por la llegada de un mensaje en el canal VModoMagnetoEncendido. Éste es enviado por el modo de funcionamiento del marcapasos al momento en que se detecta la presencia del magneto. Luego de esto se inicializa el reloj tcb que indica el tiempo desde que se chequeó por última vez el nivel de batería.

Inmediatamente se transiciona a una nueva ubicación, equivalente a la única ubicación del modo VOO_Base con algunos agregados y modificaciones:

- se sustituye la función tiempoCicloFijo por tiempoCicloBateria(eBateria), de esta forma en lugar de estimular cada un tiempo fijo predefinido como lo hace el modo VOO_Base, se estimula con una frecuencia que depende del nivel de batería.

- Cada vez que se realiza una estimulación se vuelve a chequear el nivel de batería y se actualiza la frecuencia con la que se deben realizar las estimulaciones, por consiguiente, a la acción original que se realizaba en el modo VOO_Base se agrega dicha actualización.

- de detectarse el alejamiento del magneto, mediante la recepción de un mensaje por el canal MagnetoSale, se informa al modo de funcionamiento original que debe volver a su normal funcionamiento y se pasa al estado de latencia inicial, dejando todo como al principio.

5.6.1.3 VOO_Bateria

Cuando la batería está por debajo del nivel estipulado para su reemplazo (ERT por sus siglas en inglés), cualquier modo de la forma VXX pasa a un modo temporal de VOO. Siguiendo con el criterio de modularidad del caso anterior, se realiza un modelado independiente para esta versión del modo VOO, concebido este como un refinamiento del original donde se una nueva ubicación inicial (estado de latencia) en la cual se espera un mensaje por el canal VModoBateriaBajaEncendido. Este mensaje es enviado por el modo de funcionamiento original del marcapasos cuando se detecta un nivel bajo de batería. Como resultado se tiene el siguiente modelo:

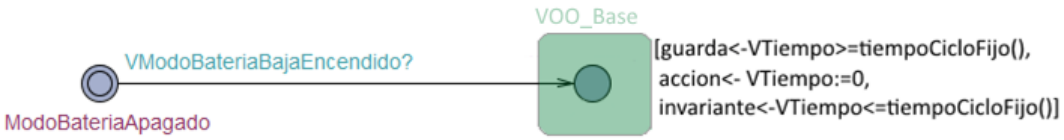

Figura 5-11

5.6.1.4 VOO

Este modo de funcionamiento es una extensión del modo VOO_Base, donde se consideran además conceptos relativos a la presencia del magneto, dando como resultado el modelo de la Figura 5-12. En el mismo, al detectarse la presencia del magneto se envía un mensaje por el canal VModoMagnetoEncendido y se pasa a un estado de latencia (quedando el modo de funcionamiento como VOO_Magneto). Solo se vuelve al modo de funcionamiento normal si se recibe un mensaje por el canal VModoMagnetoApagado.

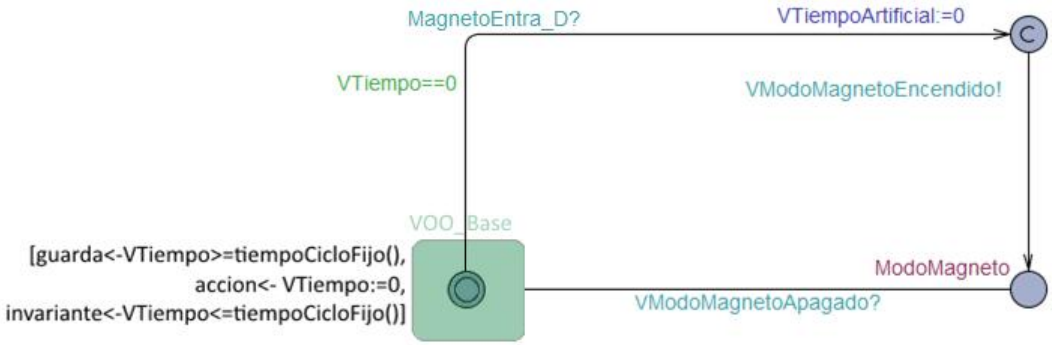

Figura 5-12

5.6.1.5 VVI

Este es un modo particular de los marcapasos que estimulan una única cámara cuyas particularidades son:

- El evento que genera la estimulación es el transcurso de un período de tiempo previamente programado (tiempoCicloFijo) desde la última estimulación o desde el último pulso ventricular natural detectado (el último de ambos).

- La condición para este caso es que un pulso natural detectado no puede haberse producido en un tiempo inferior al preestablecido como el período refractario ventricular (VRP por sus siglas en inglés), ya que podría estar detectándose por error o bien la última estimulación realizada, o bien el último pulso natural (en forma repetida). Por consiguiente dicha detección debe ser descartada.

- La acción a realizar posteriormente a la estimulación es volver a cero al reloj tiempo.

Al igual que para el modo VOO y también con el objetivo de simplificar su entendimiento, inicialmente se presenta una versión simplificada del modo VVI; en la misma no se consideran conceptos relativos a la presencia del magneto o el nivel de batería, dando como resultado el modelo de la Figura 5-13. Se puede observar además que para manejar el período refractario ventricular se utiliza una ubicación comprometida, la cual implica que no se puede permanecer en la misma y se debe transicionar a otra ubicación en forma inmediata. En esta ubicación se chequea si la detección del pulso se dio dentro o fuera del período de refracción, para luego descartar o no el pulso detectado. Si bien la utilización de una ubicación comprometida no es estrictamente necesaria y la solución puede implementarse simplemente utilizando guardas, el sistema de verificación de UPPAAL no permite la utilización de guardas temporales y canales de difusión en una misma transición, de ahí la alternativa elegida.

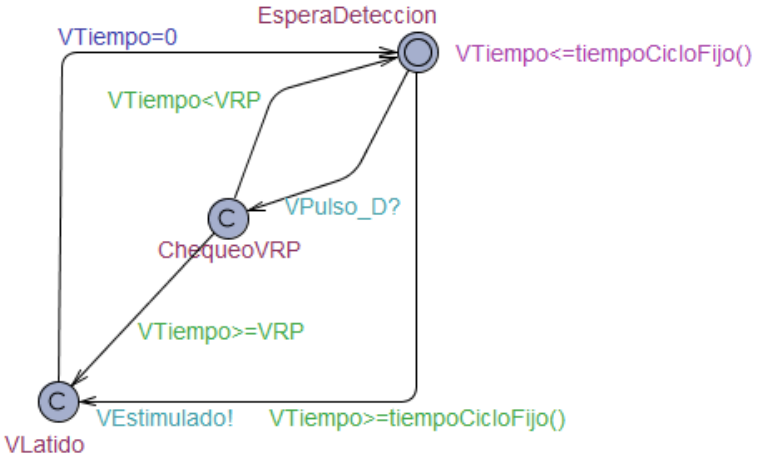

Figura 5-13

Posteriormente al modelo anterior se le agrega el comportamiento asociado a la presencia del magneto, dando como resultado el modelo de la Figura 5-14. En el mismo, al detectarse la presencia del magneto se envía un mensaje por el canal VModoMagnetoEncendido y se pasa a un estado de latencia (quedando el modo de funcionamiento como VOO_Magneto). Solo se vuelve al modo de funcionamiento normal si se recibe un mensaje por el canal VModoMagnetoApagado.

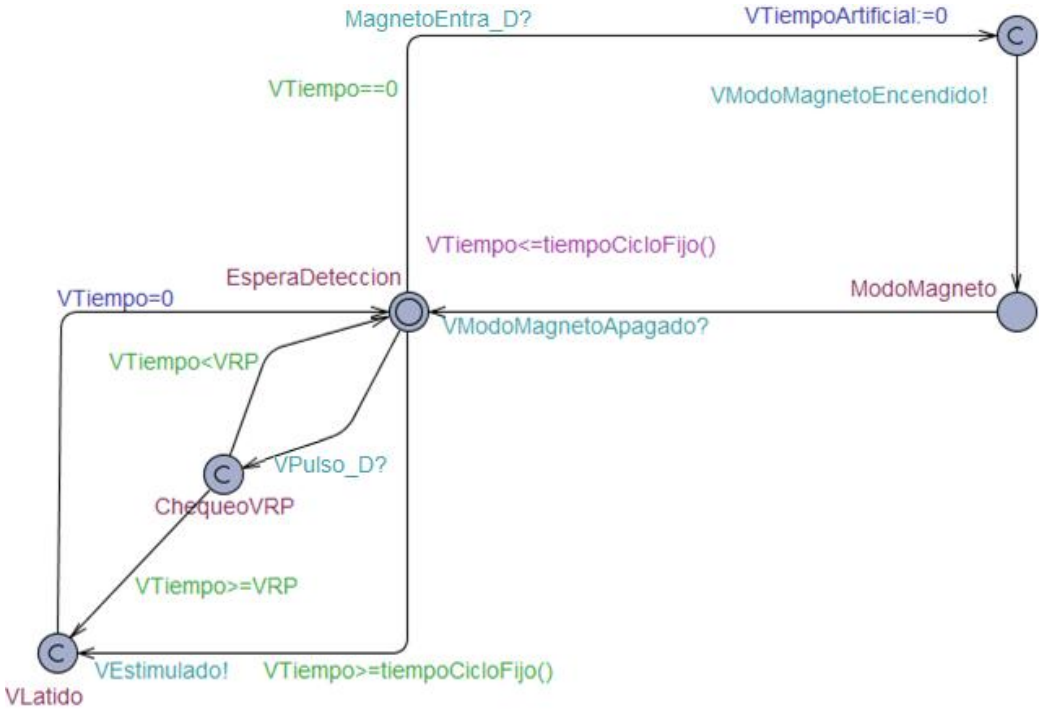

Figura 5-14

Finalmente, agregando el comportamiento asociado a un nivel de batería bajo (inferior a ERT), se obtiene el siguiente modelo:

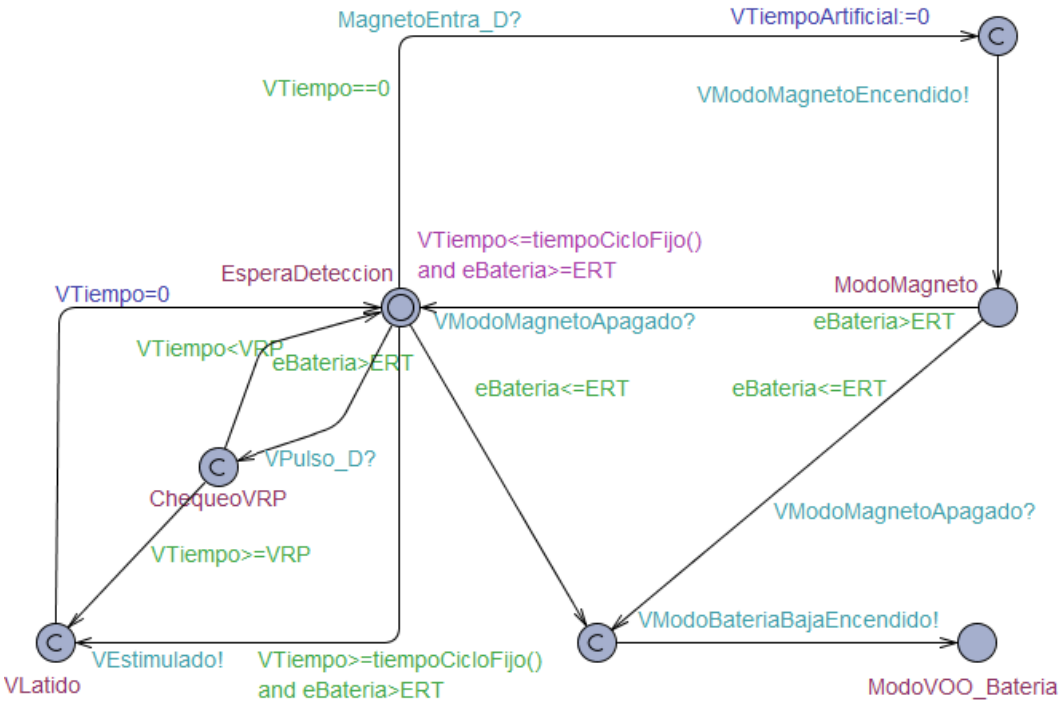

Figura 5-15

Siempre que se esté en modo VOO_Magneto y se reciba el mensaje VModoMagnetoApagado, o se esté en el modo de funcionamiento normal, si el nivel de batería inferior a el ERT se envía un mensaje por el canal VModoBateriaEncendido, quedando en un estado de latencia y pasando al modo de funcionamiento VOO_Bateria.

5.6.1.6 VVT

Este modo es un refinamiento del modo VVI, donde al momento de detectar un pulso ventricular natural, el mismo es reforzado mediante una estimulación. Por consiguiente se tiene como resultado el siguiente modelo:

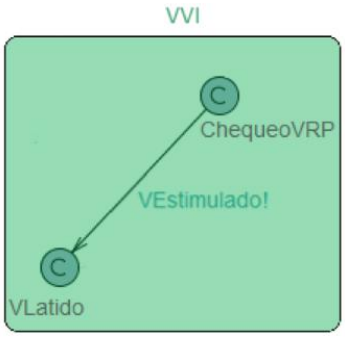

Figura 5-16

5.6.2 Marcapasos bicamerales

Este subgrupo se corresponde con los modos de funcionamiento: DOO, DDI y DDD. En esencia, su funcionamiento (ver Figura 5-17) se puede describir como se detalla a continuación:

- EventoA: Representa el evento que desencadena la estimulación artificial de la cámara auricular mediante el envío de un mensaje por el canal AEstimulado.
- GuardaA: Indica la condición que debe cumplirse para poder realizar la estimulación de la aurícula luego de haberse generado el evento.
- AcciónA: Indica la acción que debe realizarse luego de realizarse la estimulación a la aurícula
- EventoV: Representa el evento que desencadena la estimulación artificial de la cámara ventricular mediante el envío de un mensaje por el canal VEstimulado, sucediendo luego de haberse estimulado o detectado un pulso auricular.
- GuardaV: Indica la condición que debe cumplirse para poder realizar la estimulación del ventrículo luego de haberse generado el evento.
- AcciónV: Indica la acción que debe realizarse luego de realizarse la estimulación

Figura 5-17

5.6.2.1 DOO_Base

El objetivo de este modo es servir como base para futuros refinamientos, procediéndose de igual forma que para el modo VOO_Base. Por consiguiente este modo de funcionamiento es la versión más básica del modo DOO, donde el evento auricular que genera la estimulación es que haya transcurrido un período de tiempo previamente programado (tiempoCicloFijo - periodoAVFijo). Igualmente para el evento ventricular, pero con la salvedad que en este caso el tiempo a transcurrir debe ser periodoAVFijo. Por lo que el ciclo completo de estimulación aurícula-ventrículo tiene una duración igual a tiempoCicloFijo. En este modo no hay condiciones y la acción es volver a cero

el reloj VTiempo o ATiempo según corresponda. Como resultado de esto se obtiene el siguiente modelo:

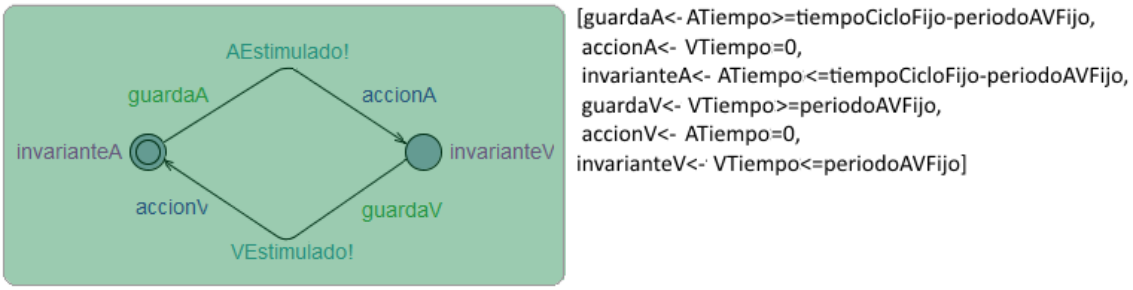

Figura 5-18

5.6.2.2 DOO_Magneto

Este modo, al igual que el modo VOO_Magneto no requiere necesariamente ser modelado en forma independiente, pero siguiendo con el mismo criterio de modularidad se lo modela por separado.

Cuando se detecta la presencia del magneto, siempre y cuando el modo de funcionamiento sea de la forma DXX, el mismo se suspende y se pasa al modo denominado DOO_Magneto. Éste es un refinamiento del modelo realizado para el modo DOO_Base, donde los pulsos se generan a una frecuencia que depende del nivel de batería del marcapasos (ver Tabla 3-1). Es a su vez un modo temporal, dado que una vez detectado el alejamiento del magneto se vuelve al modo de funcionamiento original o DOO_Bateria (dependiendo del nivel de batería). Su comportamiento es muy similar al del modo VOO_Magneto y difiere de este en la forma de estimulación a las cámaras que realiza (DOO en lugar de VOO). Por consiguiente su modelo termina resultando el siguiente:

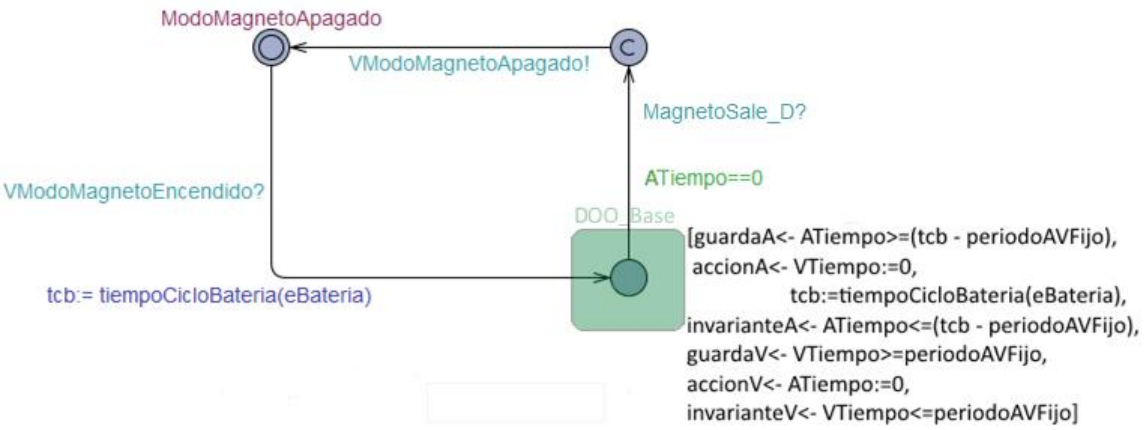

Figura 5-19

Como puede apreciarse, la representación gráfica de este modo es prácticamente idéntica a la del modelo VOO_Magneto (ver Figura 5-10), diferenciándose solamente en el grafo que referencian (VOO_Base o DOO_Base según el caso).

5.6.2.3 DOO_Bateria

En forma análoga al razonamiento utilizado para transformar el modelo que llevó del modo VOO_Base al VOO_Batería se puede inferir que el modelo DOO_Bateria es el siguiente:

Figura 5-20

5.6.2.4 DOO

Razonando de igual forma que para el modo VOO, este modo de funcionamiento es una extensión del modo DOO_Base, donde se consideran además conceptos relativos a la presencia del magneto, dando como resultado el modelo de la Figura 5-21. En el mismo, al detectarse la presencia del magneto se envía un mensaje por el canal VModoMagnetoEncendido y se pasa a un estado de latencia (quedando el modo de funcionamiento como DOO_Magneto). Solo se vuelve al modo de funcionamiento normal si se recibe un mensaje por el canal VModoMagnetoApagado.

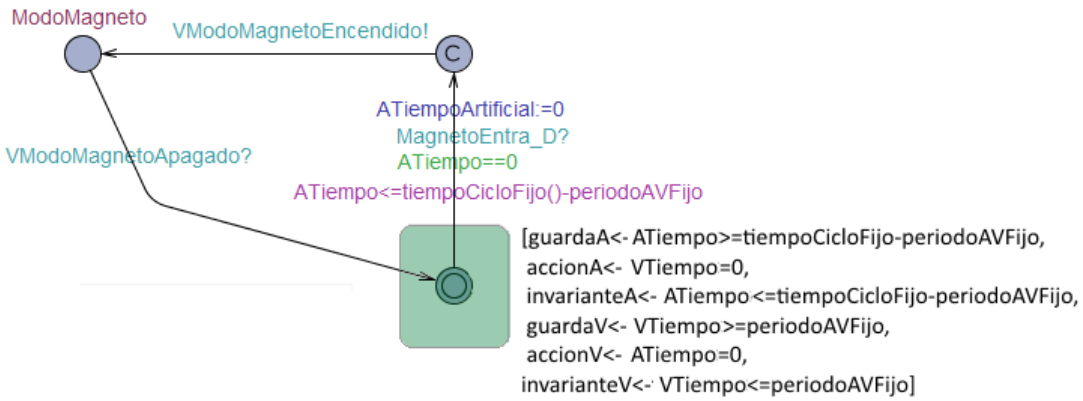

Figura 5-21

5.6.2.5 DDI

Esta modalidad de funcionamiento escucha y estimula tanto al ventrículo como a la aurícula. Si un pulso auricular es detectado esto puede causar la estimulación del ventrículo en un período periodoAVFijo, salvo que antes se detecte un pulso ventricular. Adicionalmente debe darse que la frecuencia auricular debe ser menor a URL, para impedir, por ejemplo, que ante una fibrilación de la aurícula se sobre-estimule al ventrículo.

Conjuntamente con esto, para que un pulso ventricular detectado sea considerado como tal, debe haber pasado el período refractario auricular (ARP, por sus siglas en inglés). En el caso del pulso auricular, debe haber pasado el período refractario post ventricular (PVARP).

Al igual que para el modo VVI, se considera inicialmente el modo de funcionamiento más simple, donde no se consideran comportamientos asociados a la presencia del magneto o a un nivel de batería baja, dando como resultado el modelo de la Figura 5-22. En este caso, el esquema general de marcapasos bicamerales se extiende mediante la sustitución de cada uno de los estados del esquema general por dos pequeños grafos:

- uno similar al utilizado para modelar el modo VVI simplificado (Figura 5-13).
- otro similar al utilizado para modelar el modo AAI simplificado, el cual es en esencia igual al VVI simplificado pero para la aurícula en lugar del ventrículo.

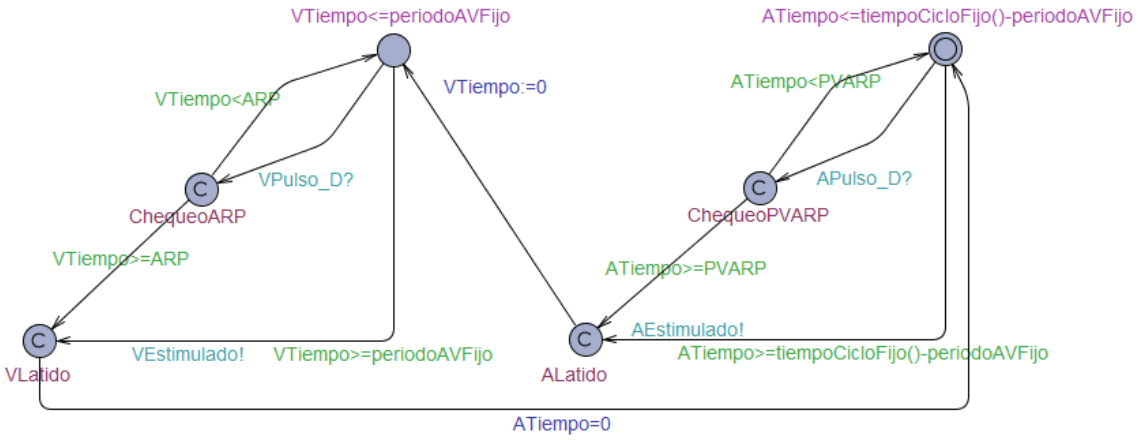

Figura 5-22

Posteriormente, y de forma análoga a lo realizado para el modo VVI se le agrega el comportamiento asociado a la presencia del magneto, dando como resultado el modelo de la siguiente figura:

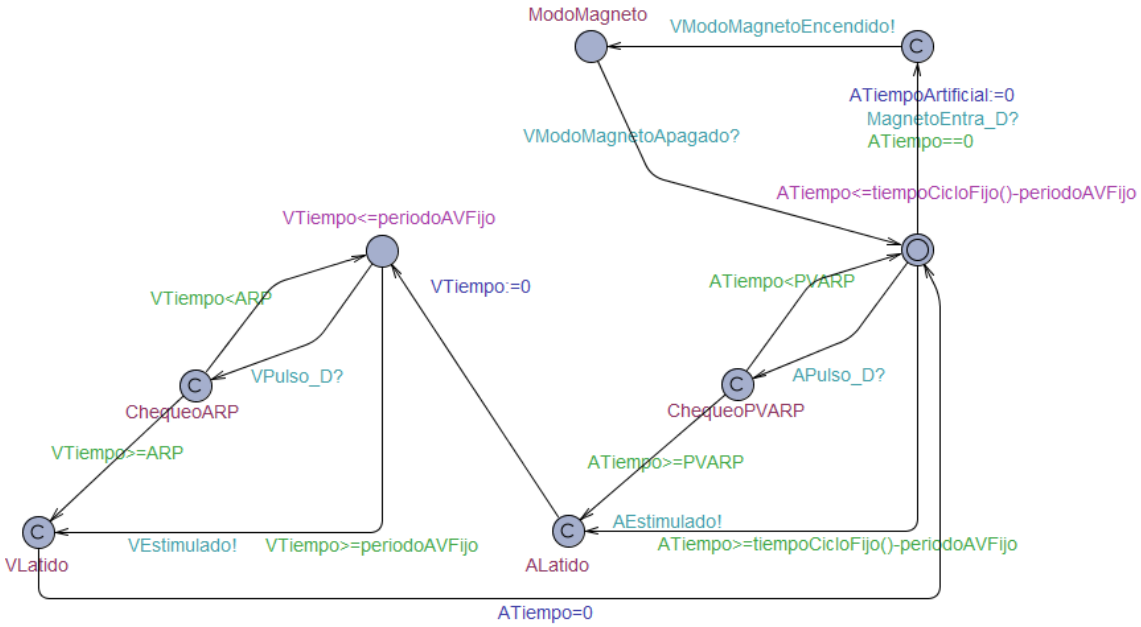

Figura 5-23

Finalmente, y al igual que para el modo VVI, se agrega el comportamiento asociado a un nivel de batería bajo, lo que resulta en el siguiente modelo:

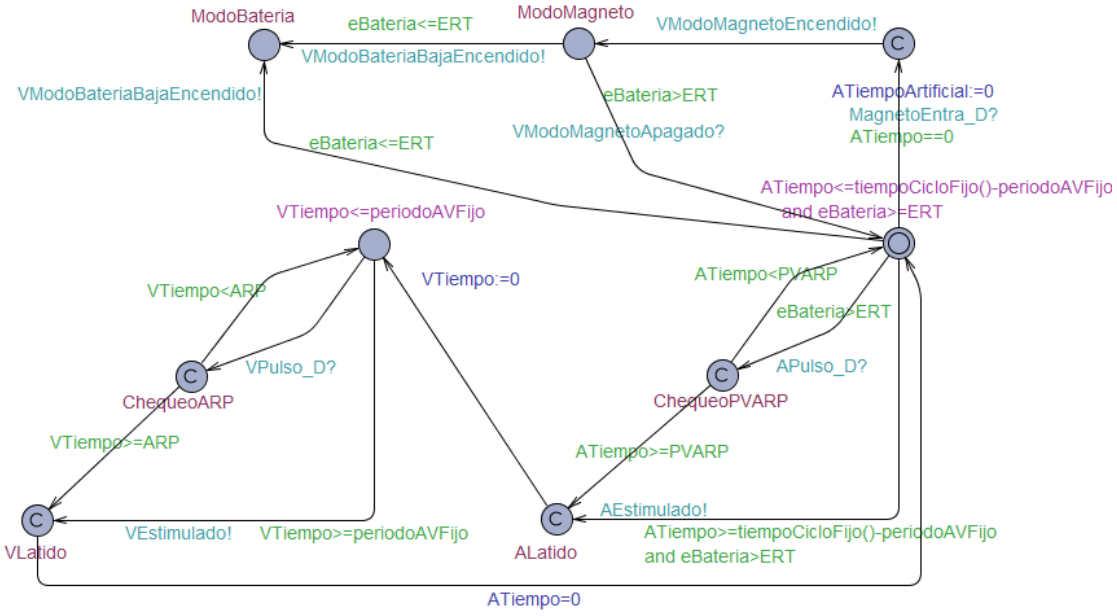

Figura 5-24

5.6.2.6 DDD

Este modo es igual al modo DDI, salvo que al momento de detectar un pulso ventricular o auricular naturales, los mismos son reforzados mediante una estimulación. Por consiguiente refinando el modo DDI se tiene como resultado el siguiente modelo:

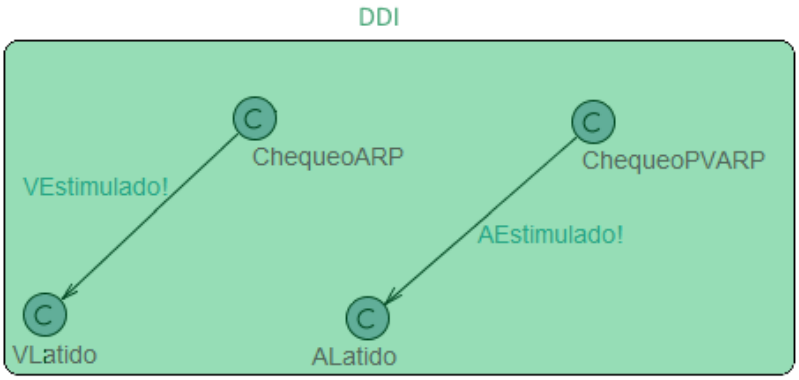

Figura 5-25

6 Verificación

La verificación de los modelos propuestos en el capítulo anterior se realiza utilizando la herramienta UPPAAL [2], la cual permite modelar, validar y verificar sistemas de tiempo real. Se basa en los grafos temporizados de Alur y Dill y fue desarrollada conjuntamente por grupos de la Universidad de Uppsala y de la Universidad de Aalborg. Está diseñada para verificar sistemas que pueden ser modelados como redes de autómatas temporales, extendidos con variables enteras, tipos estructurados de datos, funciones de usuario y canales de sincronización. Las áreas típicas de aplicación incluyen controladores de tiempo real, protocolos de comunicación y demás sistemas en los cuales los aspectos de sincronización son críticos. UPPAAL difiere en algunos puntos con lo propuesto en el capítulo 4, por lo que se deben tomar las siguientes consideraciones:

- En UPPAAL en lugar de definirse autómatas se definen plantillas de autómatas**,** con un conjunto de parámetros con tipo (ej: int, chan, etc.). Estos parámetros son posteriormente sustituidos por los argumentos dados en la declaración del proceso. De esta forma, algunos valores que inicialmente fueron definidos como constantes pueden convertirse en parámetros de la plantilla, procedimiento que no tiene ningún efecto negativo sobre el análisis previamente realizado.

- UPPAAL no permite referenciar a grafos preexistentes, en consecuencia debe sustituirse en los modelos presentados en el capítulo anterior cada referencia a un grafo preexistente por el grafo en sí mismo, aplicándole en los casos que corresponda las modificaciones pertinentes.

6.1 Modelos

Para poder realizar la verificación del modelo es necesario primeramente definir los diferentes modelos en UPPAAL, para luego, mediante el uso de dicha herramienta chequear las propiedades que se consideren necesarias. En tal sentido, es necesario presentar las declaraciones y definiciones necesarias para representar la solución propuesta en el capitulo anterior, y posteriormente se despliegan los diferentes modelos en UPPAAL.

6.1.1 Declaración de canales, relojes y funciones auxiliares

A continuación se presenta una sección del código que se debe ingresar en UPPAAL para la declaración de los relojes del sistema, los diferentes canales utilizados, las funciones auxiliares y constantes utilizadas:

```
int tiempoCicloFijo () { // ms
        return 60000 / LRL;
}
```

```
int TARP(){
        return periodoAVFijo + PVARP;
}

int tiempoCicloFijoURL () { // ms
        return 60000 / URL;
}

// Relojes necesarios para controlar el tiempo entre estimulaciones
y/o pulsos naturales.
clock VTiempo,ATiempo;
clock VTiempoArtificial, ATiempoArtificial;

// Sincronización
// Corazón
broadcast chan VPulso_D, APulso_D;
// Detección Auricular
chan AEstimulado;
// Detección Ventricular
chan VEstimulado;
// Detección Magneto
broadcast chan MagnetoEntra_D, MagnetoSale_D;

chan VModoMagnetoEncendido, VModoMagnetoApagado;
chan AModoMagnetoEncendido, AModoMagnetoApagado;

// Nivel Batería
chan VModoBateriaBajaEncendido;
chan AModoBateriaBajaEncendido;

int eBateria=100;
int BOT = 100; // Beginning of Life
int ERN = 15;  // Elective Replacement Near
int ERT = 10;  // Elective Replacement Time
int ERP = 9;   // Elective Replacement Past ERP

int tiempoCicloBateria(int eb){
    if(eb>ERN)
        return 600; //100ppm
    else if(eb>ERT)
        return 667; //90ppm
    else if(eb>ERT)
        return 706; //85ppm
    else if(eb>0)
        return (706/eb)*ERT;
    else
        return 0;
}
```

6.1.2 Corazón

El modelado en UPPAAL es esencia el presentado en el punto 5.3 (ver Figura 5-5), con la salvedad de que los intervalos de las frecuencias de los pulsos ventriculares o auriculares, al igual que si estos son o no intercalados, en lugar de manejarse por intermedio de variables se lo hace con parámetros de la plantilla. La plantilla además consta de los siguientes parámetros:

- vPeriodoMinimo. Indica el período de tiempo mínimo entre pulsos ventriculares.
- vPeriodoMaximo. Indica el período de tiempo máximo entre pulsos ventriculares.
- aPeriodoMinimo. Indica el período de tiempo mínimo entre pulsos auriculares.
- aPeriodoMaximo. Indica el período de tiempo máximo entre pulsos auriculares.
- intercalado. Indica si los pulsos naturales de las cámaras deben generarse en forma intercalada.

Adicionalmente es necesario agregar la actualización de la variable control, la cual es utilizada posteriormente en las propiedades que se verifican. El objetivo de dicha variable es reflejar cuál fue la última cámara estimulada (1 para ventrículo, 0 para aurícula).

Como resultado de todo lo anterior, la plantilla UPPAAL que modela al corazón resulta ser la siguiente:

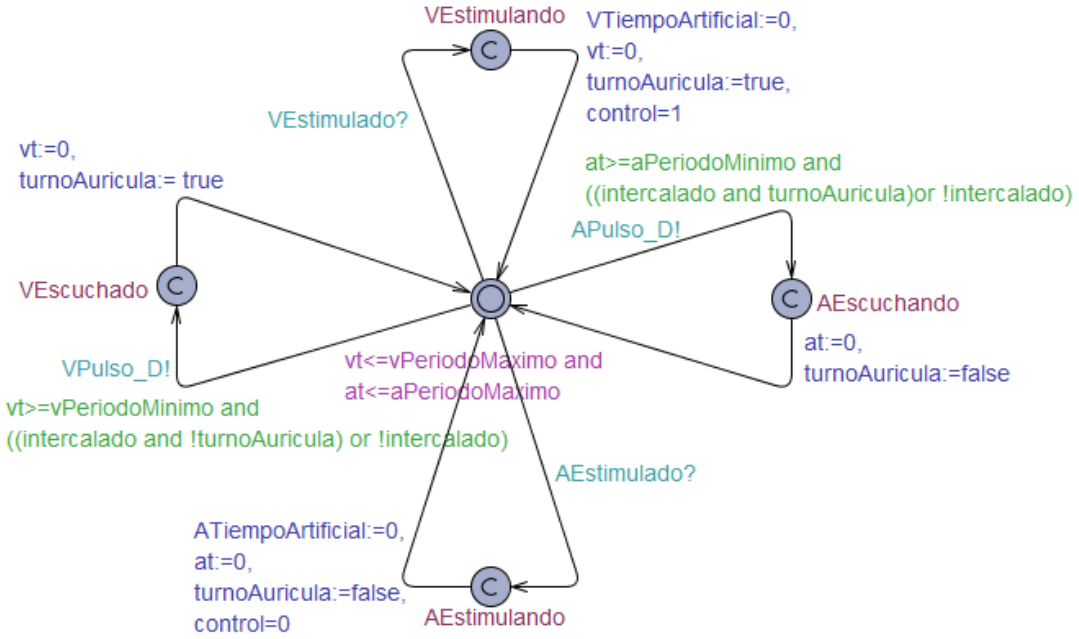

Figura 6-1

6.1.3 Batería

La plantilla utilizada para modelar la batería es igual a la presentada en el punto 5.4 (ver Figura 5-6) con los siguientes parámetros de configuración:

- estadoInicial. Permite indicar el nivel inicial de la batería y toma valores entre 0 y 100.

- velocidadDescarga. Utilizado para indicar la velocidad de descarga de la batería, indica el tiempo necesario para descontar en uno la capacidad de la batería.

- estadoFinal. Permite indicar el nivel final de la batería, toma valores entre 0 y 100. Llegado a este estado, la batería deja de descargarse.

6.1.4 Magneto

La plantilla utilizada para modelar el magneto es igual a la presentada en el punto 5.5 (ver Figura 5-7) con dos parámetros, los cuales se detallan a continuación:

- tiempoDuracion. Utilizado para indicar el tiempo mínimo en que el magneto se encuentra presente (está próximo al marcapasos).

- tiempoEspera. Utilizado para indicar el tiempo mínimo en que el magneto se encuentra ausente (no está próximo al marcapasos).

6.1.5 VOO

En este caso, la plantilla utilizada en UPPAAL para modelar el modo VOO es la siguiente:

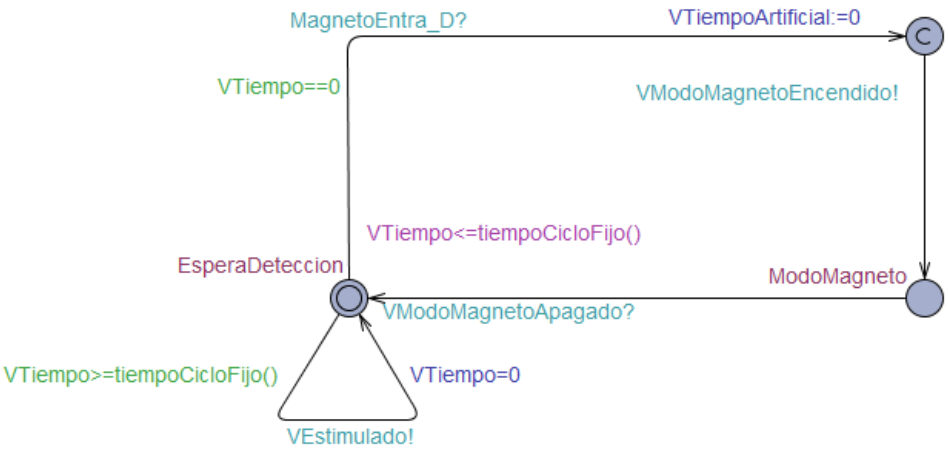

Figura 6-2

6.1.6 VOO_Magneto

Para este modo, la plantilla utilizada en UPPAAL para modelar este modo es la siguiente:

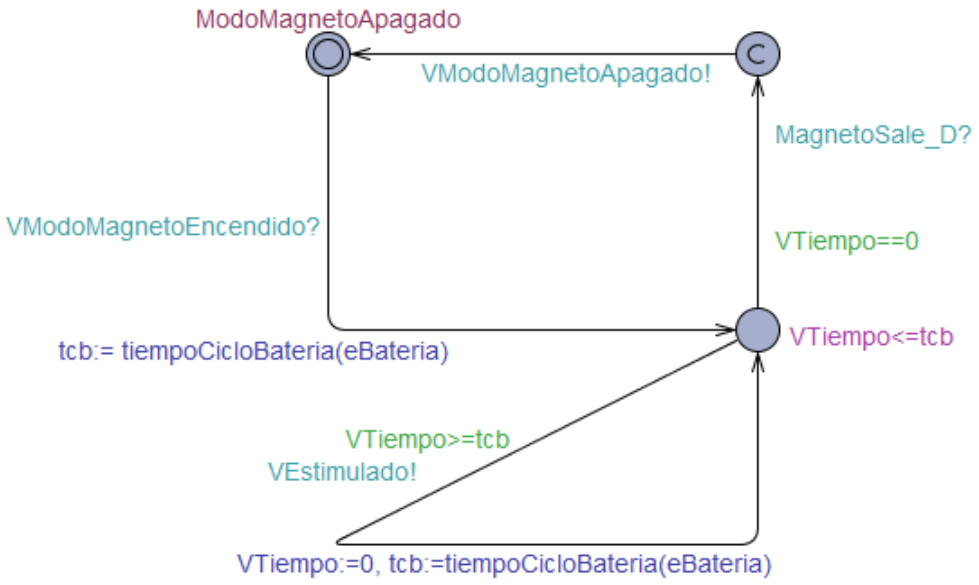

Figura 6-3

6.1.7 VOO_Bateria

La plantilla utilizada en UPPAAL para modelar este modo es la siguiente:

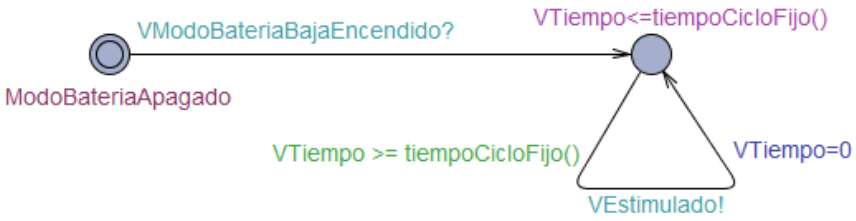

Figura 6-4

6.1.8 VVI

A continuación se presenta la plantilla que modela el modo VVI de funcionamiento del marcapasos:

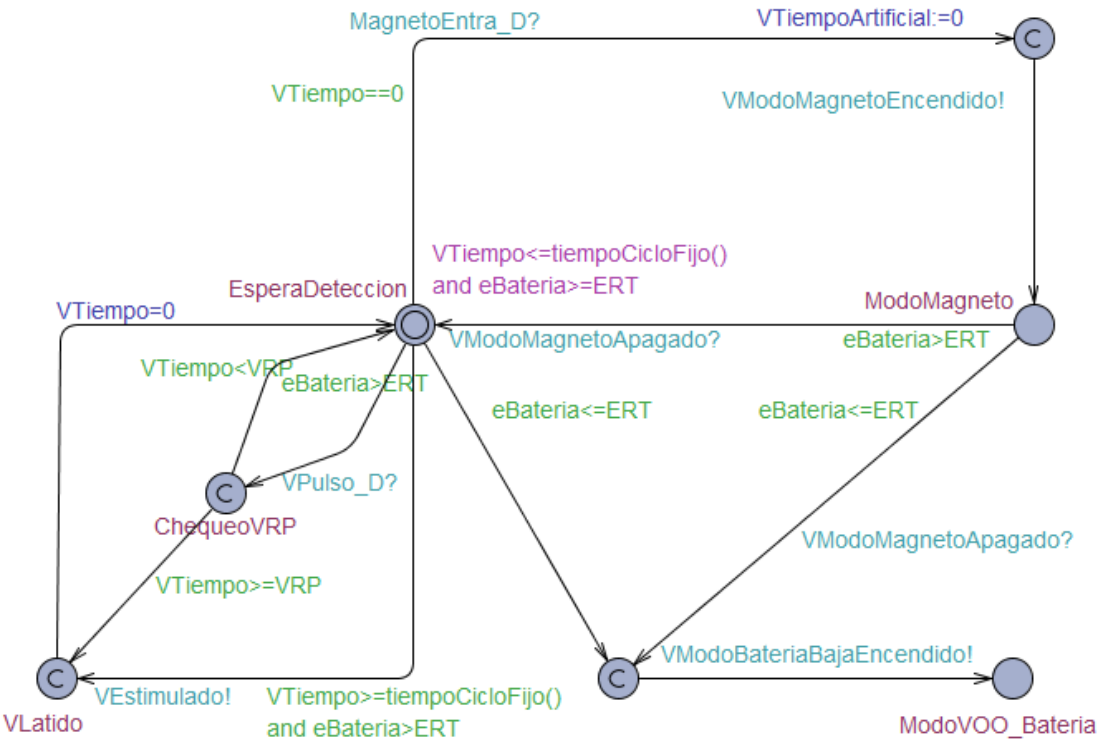

Figura 6-5

6.1.9 VVT

Si el modo de funcionamiento del marcapasos es VVT su plantilla es la siguiente:

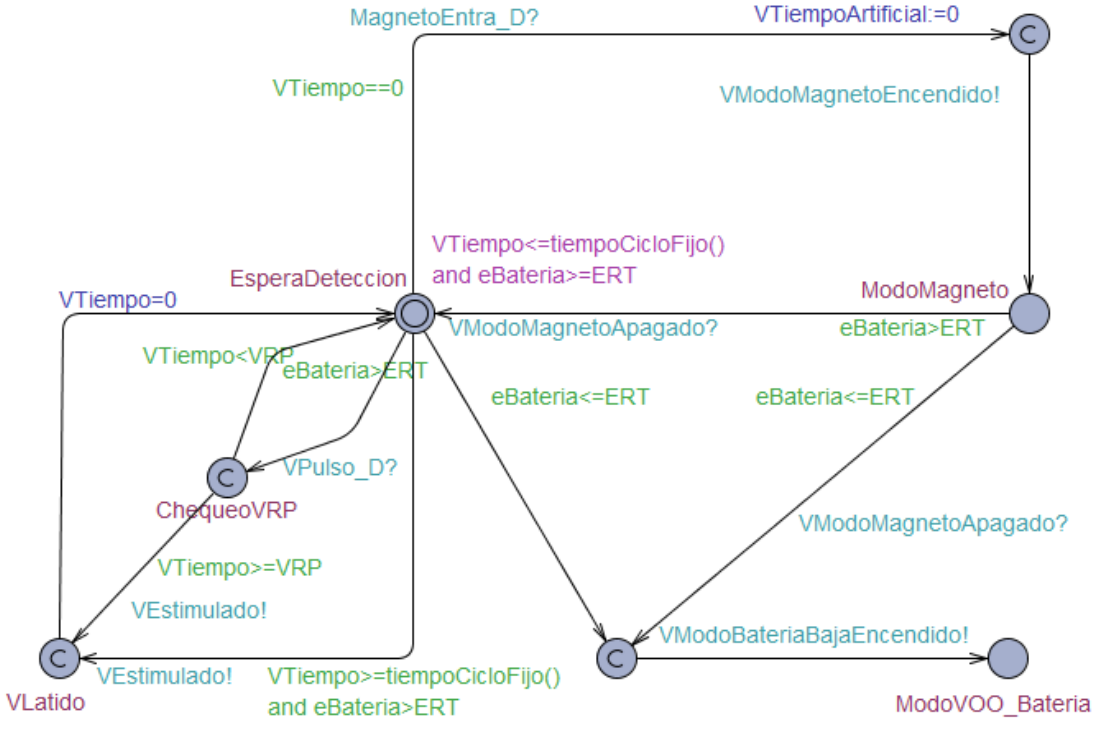

Figura 6-6

6.1.10 DOO

Para el caso en que el modo de funcionamiento del marcapasos es DOO su plantilla es la siguiente:

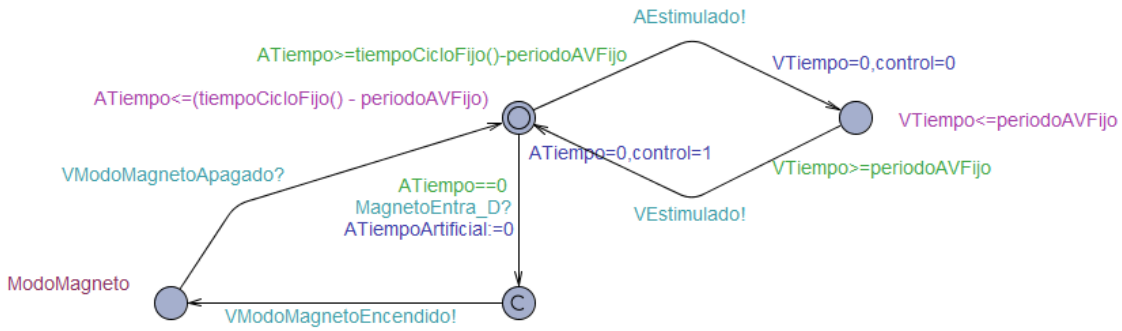

Figura 6-7

6.1.11 DOO_Magneto

Cuando el modo de funcionamiento del marcapasos es DOO_Magneto su plantilla es la siguiente:

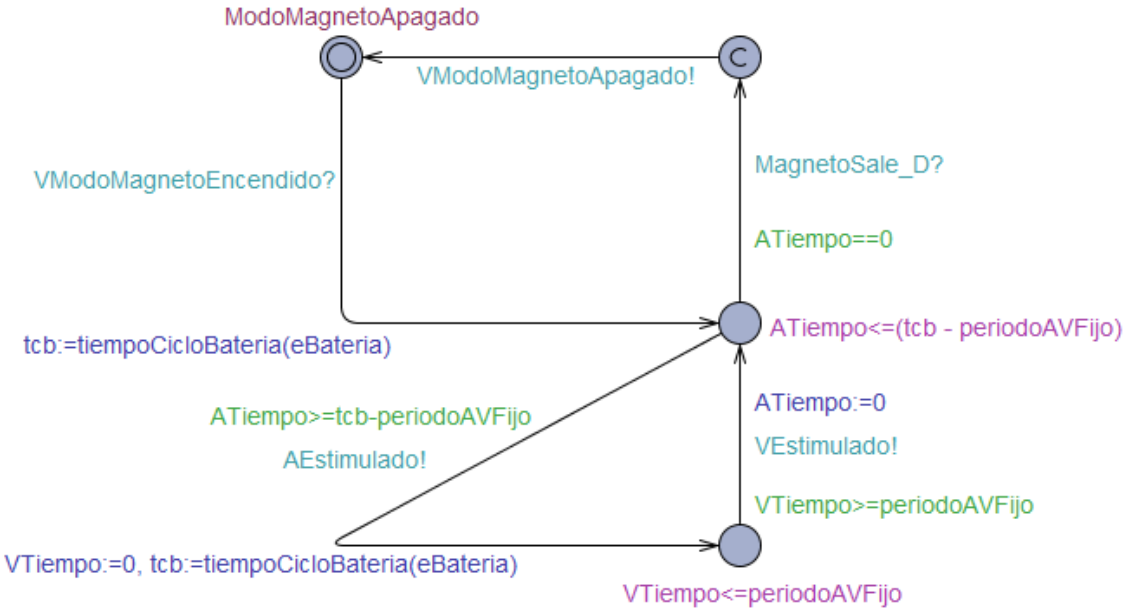

Figura 6-8

6.1.12 DOO_Bateria

La siguiente plantilla modela el modo de funcionamiento del marcapasos DOO_Bateria:

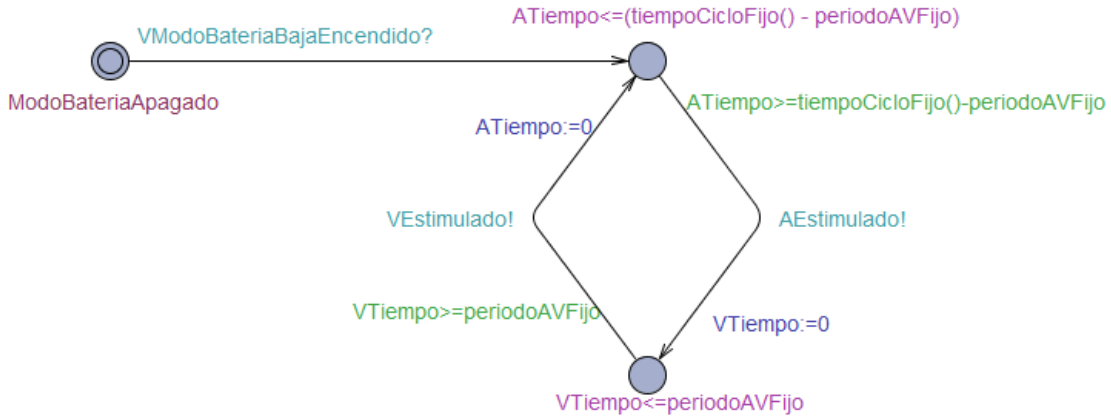

Figura 6-9

6.1.13 DDI

En el caso en que el modo de funcionamiento del marcapasos sea DDI, la plantilla que lo modela es la siguiente:

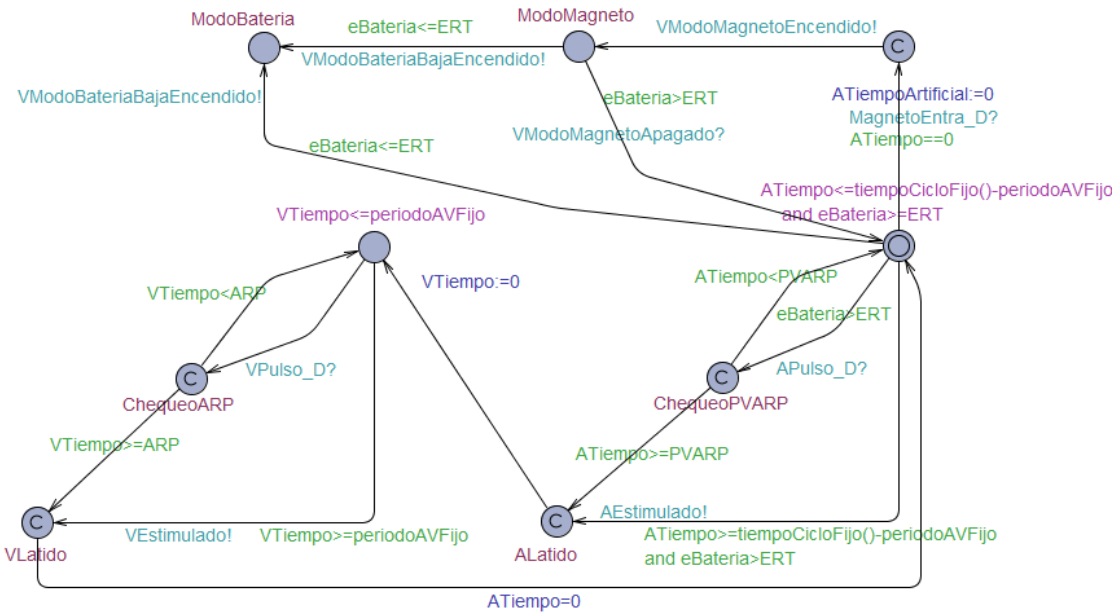

Figura 6-10

6.1.14 DDD

Finalmente, cuando el marcapasos está en el modo de funcionamiento DDD, su plantilla es la siguiente:

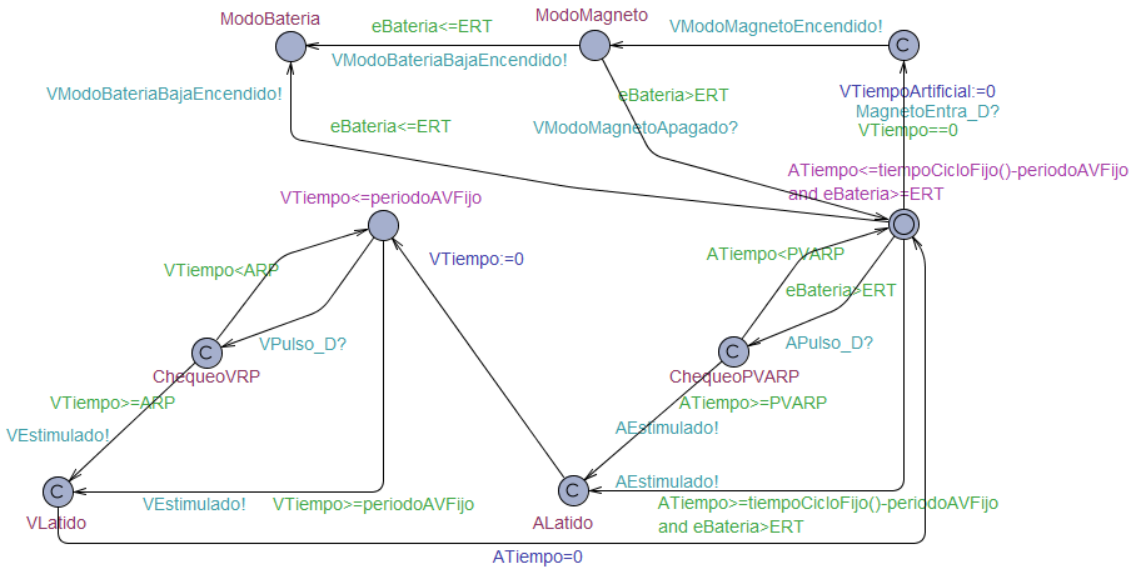

Figura 6-11

6.2 Sistema Simulador

Para simular un escenario se requiere indicar los diferentes componentes (grafos temporales) que integrarán el sistema y en caso de que alguna de las plantillas tenga parámetros, deben ser suministrado en este momento. Con el objetivo de verificar que todos los modelos presentados anteriormente son correctos se realizaron verificaciones sobre diferentes escenarios, los que se listan a continuación:

- modo de operación VOO, corazón que funciona en forma intercalada y tiene pulsos ventriculares irregulares y un magneto que se aproxima y aleja en forma aleatoria.

- modo de operación VVI, corazón que funciona en forma intercalada y tiene pulsos ventriculares irregulares, un magneto que se aproxima y aleja en forma aleatoria y una batería que comienza al 100% y se descarga hasta llegar al 1%.

- modo de operación VVT, corazón que funciona en forma intercalada y tiene pulsos ventriculares irregulares, un magneto que se aproxima y aleja en forma aleatoria y una batería que comienza al 100% y se descarga hasta llegar al 1%.

- modo de operación DOO, la aurícula y el ventrículo tienen pulsos intrínsecos en forma independiente con tiempos irregulares, un magneto que se aproxima y

aleja en forma aleatoria y una batería que comienza al 100% y se descarga hasta llegar al 1%.

- modo de operación DDI, la aurícula y el ventrículo tienen pulsos intrínsecos en forma independiente con tiempos irregulares, un magneto que se aproxima y aleja en forma aleatoria y una batería que comienza al 100% y se descarga hasta llegar al 1%.

- modo de operación DDD, la aurícula y el ventrículo tienen pulsos intrínsecos en forma independiente con tiempos irregulares, un magneto que se aproxima y aleja en forma aleatoria y una batería que comienza al 100% y se descarga hasta llegar al 1%.

Para todos los escenarios se utilizaron los valores nominales de los parámetros configurables del marcapasos, siendo estos los siguientes:

```
int LRL = 60; // Lower Rate Limit (ppm)
int URL = 120; // Upper Rate Limit (ppm)
int periodoAVFijo = 150; // Fixed AV Delay (ms)
int ARP = 250; // Período Refractario Auricular(ms)
int VRP = 10; // Período Refractario Ventricular(ms)
int PVARP = 250; // Período Refractario Post Ventricular (ms)
```

A continuación presenta un ejemplo de simulación donde:

- tanto el ventrículo como la aurícula tienen un período de frecuencia de pulsos de 1100 milisegundos, y se dan en forma intercalada.

- la batería tiene una carga inicial del 30%, pierde un 1% de carga cada 1 segundo y la carga nunca baja del 10%.

- El magneto se presenta como máximo cada 15 segundos y por un lapso de por lo menos 5 segundos.

- El modo de funcionamiento del marcapasos se definió en VVI

```
corazon = Corazon(1100,1100,1100,1100,true);
bateria = Bateria(30,1000,10);
magneto = Magneto(15000, 5000);
modoMagneto = VOO_ModoMagneto();
modoBateria = VOO_Bateria();
marcapasos = VVI();
system corazon,marcapasos,modoMagneto,magneto,bateria,modoBateria;
```

6.3 Propiedades en UPPAAL

UPPAAL soporta la verificación de fórmulas de la lógica TCTL, las cuales son de la siguiente forma:

- `A[] P` Siempre se cumple P
- `E<> P` Finalmente se cumple P

Donde P es una proposición o conjunción de proposiciones lógicas.

Sin embargo, UPPAAL incorpora otra serie de expresiones posibles, que aumentan la capacidad de verificación de la herramienta, las cuales son:

- `E[] P` Existe un camino en el cual P siempre se cumple.
- `A<> P` Para todos los caminos, P finalmente se cumple.
  Equivalente a: `not E[] not p`
- `P --> Q` Si P se cumple, Q finalmente debe cumplirse.
  Equivalente a: `A[] (P imply A<> Q)`
- `deadlock` Significa que el sistema se bloquea. Esto, por ejemplo, permite verificar `A[] not deadlock` (el sistema nunca se bloquea).

6.4 Propiedades verificadas

A continuación se listan las propiedades que se verificaron y su correspondiente forma de expresarlas en UPPAAL (TCTL):

A. En todo momento el tiempo transcurrido entre dos pulsos auriculares es menor a la frecuencia mínima programada (LRL).
```
A[] ATiempoArtificial<=tiempoCicloFijo()
```

B. En todo momento el tiempo transcurrido entre dos pulsos ventriculares es menor a la frecuencia mínima programada (LRL).
```
A[] VTiempoArtificial<=tiempoCicloFijo()
```

C. No se detectan pulsos ventriculares dentro del período refractario ventricular.
```
A[] ((corazon.VEstimulando and modoMagneto.ModoMagnetoApagado)
  imply (VTiempoArtificial>=VRP) and VTiempo<=tiempoCicloFijo())
```

D. No se detectan pulsos auriculares dentro del período refractario auricular.
```
A[] (corazon.AEstimulando imply ATiempoArtificial>=ARP)
```

E. Cuando el nivel de batería es inferior a ERT se está en modo batería.
```
A[] (eBateria<ERT  imply !modoBateria.ModoBateriaApagado)
```

F. En el modo magneto, la frecuencia de los pulsos generados se corresponde con el nivel de batería.
```
A[] ((corazon.VEstimulando and !modoMagneto.ModoMagnetoApagado)
 imply (  (eBateria>=ERN and VTiempoArtificial==600)
  or (eBateria>=ERT and eBateria<ERN and VTiempoArtificial==667)
  or (eBateria>=ERN and eBateria<ERP and VTiempoArtificial==706)
  or (eBateria> 0 and eBateria<ERP and VTiempoArtificial<706)
  or (eBateria==0) ))
```

G. No hay bloqueos.
```
A[] not deadlock
```

H. Los pulsos ventriculares y auriculares se generan en forma intercalada.
Debido al poder de expresión que brinda UPPAAL para especificar propiedades no hay una forma natural de expresar esta propiedad y es necesario dividir el problema en varias partes:

- Cada vez que se estimula al ventrículo la bandera de control está apagada.
```
A[] (corazon.VEstimulando imply control==0)
```

- Cada vez que se estimula a la aurícula la bandera de control está encendida.
```
A[] (corazon.AEstimulando imply control==1)
```

- Si la bandera de control está apagada y se estimula a la aurícula, posteriormente la bandera se enciende.
```
(control==0 and corazon.AEstimulando) --> ( control==1)
```

- Si la bandera de control está encendida y se estimula al ventrículo, posteriormente la bandera se apaga.
```
(control==1 and corazon.VEstimulando) --> ( control==0)
```

Dado que las propiedades anteriores son escritas en UPPAAL utilizando cuantificadores temporales A[], es necesario verificar que el tiempo avanza, o lo que es lo mismo, que el conjunto de estados donde se realizan las verificaciones no es vacío. Esto puede comprobarse en UPPAAL por medio de la funcionalidad de simulación que la misma dispone, corroborando que el sistema "hace algo".

En la Tabla 6-1 puede observarse para cada propiedad verificada se indica si pudo o no verificarse para cada uno de los escenarios presentados con anterioridad.

	Propiedad							
	A	B	C	D	E	F	G	H
VOO	-	Pasa	-	-	-	Pasa	Pasa	-
VVI	-	Pasa	Pasa	-	Pasa	Pasa	Pasa	-
VVT	-	Pasa	Pasa	-	Pasa	Pasa	Pasa	-
DOO	Pasa	Pasa	-	-	Pasa	Pasa	Pasa	Pasa
DDI	Pasa	Pasa	Pasa	Pasa	Pasa	Pasa	Pasa	-
DDD	Pasa	Pasa	Pasa	Pasa	Pasa	Pasa	Pasa	Pasa

Tabla 6-1

Claramente la primera propiedad no puede asegurarse si se está en un modo de la forma VXX debido a que el modo de marcapasos elegido sólo detecta o estimula al ventrículo, por lo que la aurícula funciona en forma natural. Pero de hecho los modos auriculares son similares a los ventriculares y esto se puede verificar utilizando modos de funcionamiento del marcapasos tales como AOO, AAI o AAT para citar algunos.

7 Conclusiones

En el presente trabajo, como resultado del estudio de una búsqueda en amplitud de los diferentes enfoques y líneas de investigación referentes a la especificación formal del software de marcapasos, se obtiene el estado del arte completo de un desafío considerado importante y de referencia por la comunidad de investigadores, como lo es el planteado por Boston Scientific, algo que ya es importante en sí mismo.

El problema de especificación formal del software de un marcapasos resulta ser mucho más complejo de lo que parece en un principio; muchos aspectos necesarios para su comprensión son asumidos u omitidos en la especificación brindada por Boston Scientific y requieren mucho tiempo de investigación. También ha sido necesario profundizar los conocimientos relativos al funcionamiento biológico del corazón, del cual de antemano sólo se tenían ideas generales del mismo y no eran suficientes para afrontar esta investigación. Esto confirma uno de los problemas planteados en el seminario de Dagstuhl, i.e. los problemas en la transferencia de conocimiento.

Se propone entonces un enfoque didáctico utilizando grafos temporizados para modelar los diferentes modos de operación del marcapasos. Dado que presentar un modo de funcionamiento complejo en forma extendida dificulta su comprensión, se propone la utilización de macros. Esto facilita la utilización de una estrategia de refinamientos sucesivos y permite un avance seguro a medida que se profundiza en modos de operación más complejos, lo cual favorece una mejor comprensión del problema, permite brindar una mejor solución al mismo y facilita la transferencia de conocimiento a otros técnicos (como ser, ingenieros).

Se realiza un modelado completo de los diferentes modos estandarizados de funcionamiento del marcapasos al igual que sus modos temporales de funcionamiento ante la presencia de un magneto o bajo nivel de batería. Este modelado contiene un nivel de detalle tal que lo hace lo suficientemente realista. De esta manera con la especificación y posterior verificación de un conjunto de propiedades, como ser que, el sistema no se bloquea, no transcurre más de un tiempo prefijado sin que haya pulsos (ya sea artificiales o intrínsecos), etc., se garantiza un funcionamiento seguro de los modelos propuestos. En tal sentido, la escritura de propiedades que deben satisfacerse no resulta una tarea simple e implica muchas veces tener que realizar consideraciones particulares al momento del diseño (o modificaciones al mismo si éste ya fue realizado) de forma tal que sea posible verificar las propiedades de interés.

Comparando el enfoque propuesto en este trabajo, con el utilizado por Jiang et al. [19] en UPPAAL, se puede observar que el trabajo de Jiang requiere un conocimiento mucho más profundo en lo que respecta al funcionamiento de marcapasos, siendo difícil su comprensión por personas no entendidas en el tema. El modelado del corazón resulta ser muy básico y no permite describir algunas afecciones del corazón, además la especificación es incompleta, ya que sólo se modela el modo de funcionamiento DDD y no se consideran, por ejemplo, sus modos de operación temporales. Finalmente la

solución planteada termina siendo muy específica al problema del marcapasos, y su generalización a otro tipo de dispositivos médicos implantables es difícil.

En conclusión, se considera que la estrategia utilizada logra cumplir con los objetivos planteados, da un enfoque gráfico de los modelos que además de servir para verificar el diseño del dispositivo, ayuda a la transferencia del mismo y posibilita su comprensión por personas menos entendidas en el tema, En comparación con los trabajos presentados por Jiang et al. [19] y Chen et al. [20], este trabajo resulta más completo y realista, dado que modela mayor cantidad de modos de funcionamiento del marcapasos, verifica mayor cantidad de propiedades y permite representar un número mucho mayor de escenarios de verificación. Además, el enfoque presentado puede ser utilizado como una metodología didáctica a seguir por parte de la industria, posibilitando entre otras cosas un acercamiento con la misma.

Como trabajos futuros se puede profundizar este trabajo llegando al siguiente nivel planteado por el desafío: la generación de código que cumpla con la especificación dada y a su vez sea capaz de correr en el marcapasos especificado, por ejemplo, implementando un componente que permita traducir o transformar los modelos realizados en UPPAAL a código C (desarrollo basado en modelos). Otra posible rama de investigación puede ser la generalización del enfoque propuesto en este trabajo a otros tipos de dispositivos médicos, como ser, por ejemplo, los dispositivos utilizados para el tratamiento de la apnea del sueño. Por otro lado y dado que por su facilidad de lectura en este trabajo se adoptó en forma muy temprana la utilización de grafos temporizados, puede ser de utilidad un estudio en profundidad de otras alternativas. Finalmente, otro posible trabajo a futuro puede ser la generalización y formalización de la notación gráfica presentada en el capítulo 4 para la representación de modelos.

Referencias bibliográficas

[1] "The Pacemaker Challenge: Developing Certifiable Medical Devices," in *Report from Dagstuhl Seminar 14062*, Dagstuhl, 2014.

[2] G. Behrmann, A. David y K. G. Larsen, «A tutorial on UPPAAL 4.0,» 2006.

[3] Wikipedia, the free encyclopedia, «Bradicardia,» 2014. [En línea]. Disponible en: https://es.wikipedia.org/wiki/Bradicardia. [Último acceso: 28 07 2014].

[4] A. O. Gomes and M. V. M. Olivencia, "Formal Specification of a Cardiac Pacing System," in *FM 2009: Formal Methods Lecture Notes in Computer Science*, 2009.

[5] A. O. Gomes and M. V. M. Oliveira, "Formal Development of a Cardiac Pacemaker: From Specification to Code," *Formal Methods: Foundations and Applications. Lecture Notes in Computer Science*, vol. 6527, pp. 210-225, 2011.

[6] D. Méry and N. Kumar Singh, "Pacemaker's Functional Behaviors in Event-B," 2009.

[7] B. R. Larson, P. Chalin y J. Hatcliff, «BLESS: Formal Specification and Verification of Behaviors for Embedded Systems with Software,» de *NASA Formal Methods, 5th International Symposium, NFM 2013*, 2013.

[8] J. Sun, Y. Liu, J. Song Dong, Y. Liu, L. Shi y É. André, «Modeling and Verifying Hierarchical Real-time Systems using Stateful Timed CSP,» *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 22 Issue 1, p. Art. 3, 2013.

[9] G. Reed y A. Roscoe, «A Timed Model for Communicating Sequential Processes.,» *Theoretical Computer Science*, vol. 58, pp. 249-261, 1988.

[10] A. Tuan Luu, M. ChunZheng y Q. Thanh Tho, «Modeling and Verification of Safety Critical Systems: A Case Study on Pacemaker,» *SSIRI - IEEE Computer Society*, pp. 23-32, 2010.

[11] S. Bessling y M. Huhn, «Formal Safety Analysis and Verification in the Model Driven Development of a Pacemaker Product Line,» Dagstuhl, 2014.

[12] Esterel Techonologies, «SCADE Suite,» [En línea]. Disponible en: http://www.esterel-technologies.com/products/scade-suite. [Último acceso: 28 9 2014].

[13] Haugen, Wasowski y Czarnecki, «CVL: Common Variability Language,» de *Proceedings of the 17th International Software Product Line Conference (SPLC)*, 2013.

[14] H. D. dos Santos Macedo, J. N. Oliveira and P. G. Larsen, "Validating and Understanding Boston Scientific PACEMAKER Requirements.," Portugal, 2007.

[15] Wikipedia, the free encyclopedia, «Vienna Development Method,» 2014. [En línea]. Disponible en: https://en.wikipedia.org/wiki/Vienna_Development_Method. [Último acceso: 28 7 2014].

[16] V. La Manna, A. Bonnano y A. Motta, «Poster on a simple pacemaker implementation,» *ACM*, p. 163, Mayo 2009.

[17] C. Ghezzi, D. Mandrioli y A. Morzenti, «TRIO a Logic Language for Executable Specifications of Real-time Systems,» *Journal of Systems and Software - On the role of language in programming*, Vols. %1 de %212, issue 2, pp. 107-123, Mayo 1990.

[18] Z. Jiang, M. Pajic, A. T. Connolly, S. Dixit and R. Mangharam, "Real-time Heart

Model for Implantable Cardiac Device Validation and Verification," *ECRTS - IEEE Computer Society*, p. 239–248, 2010.

[19] Z. Jiang, M. Pajic, Moarref, R. Alur and R. Mangharam, "Modeling and Verification of a Dual Chamber Implantable Pacemaker," 2011.

[20] T. Chen, M. Diciolla, M. Kwiatkowska and A. Mereacre, "Quantitative Verification of Implantable Cardiac Pacemakers," in *IEEE 33rd Real-Time Systems Symposium*, 2012.

[21] Boston Scientific, «PACEMAKER System Specification,» 2007.

[22] R. Alur y D. I. Dil, «A theory of timed automata,» *Theoretical computer science*, vol. 126, pp. 183-235, 1994.

[23] E. Clarke, E. Emerson, Austin y A. Sistla, «Automatic Verification Of Finite State Concurrent Systems Using Temporal Logic Specifications: A Practical Approach,» de *POPL '83 Proceedings of the 10th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, 1983.

[24] R. Alur, C. Courcoubetis and D. Dil, "Model-checking in dense real-time," *Information and computation*, vol. 104, p. 2–34, 1993.

[25] E. M. Clarke, «The Birth of Model Cheking,» 2007.