Especificación Formal y Validación en el Modelado de un Sistema de control en tiempo real en Instituciones de Educación Inicial utilizando Beacons

Carlos Chullo Juan Deyby, Estudiante Universitario [1], Hancco Medina Wilder Ivan, Estudiante Universitario [2], Huallpa Tapia Luis David, Estudiante Universitario [3], Vidal Duarte Elizabeth, Mentora
Universidad Nacional de San Agustín, Perú, jcarlosc@unsa.edu.pe, whanccome@unsa.edu.pe, lhuallpat@unsa.edu.pe, evidald@unsa.edu.pe

Abstract*– En el Perú por razones de trabajo y tiempo los padres de familia envían a sus hijos a las instituciones educativas en movilidades escolares o de terceros. Es sabido que siempre existen riesgos de secuestros, accidentes y desapariciones de los menores, provocando la preocupación de los padres por la integridad de sus hijos quienes en el día a día no tienen la certeza si sus hijos llegaron con bien a su institución educativa. Al ver estos problemas se propone realizar un sistema de control en tiempo real para instituciones de educación inicial, usando Smartphone, smartwatch y beacons para obtener información sobre el arribo del niño a su institución. En este artículo presentamos la primera fase de este proyecto: la especificación de requerimientos. Dado lo critico de la fiabilidad de los requerimientos se ha realizado especificación formal con el lenguaje VDM++ y validación con la herramienta VDM++ toolbox.

Keywords—VDM++; Beacon; BLE, Smartphone, Smartwatch; Institución Educativa Inicial.

I. INTRODUCCIÓN

Durante la elaboración de sistemas de software complejos es común tener errores en la especificación requerimientos sin aún haberlos desarrollado [1]. Estos sistemas demandan que se tenga una precisión alta y los errores producidos durante la especificación de requerimientos provocan que se genere una demora en la realización del sistema, generando así también enormes gastos. Esto se produce debido a las ambigüedades que se presentan en los requerimientos, el uso del lenguaje natural es uno de los factores que causan dichas ambigüedades ya que se prestan a distintas interpretaciones [2].

El uso de Especificaciones Formales durante el desarrollo de un sistema provee varias ventajas esto debido que al usar un lenguaje estructurado y notaciones matemáticas para la especificación de requerimientos en la etapa inicial del desarrollo de software nos permite detectar y corregir errores durante la fase previa del desarrollo del sistema [2].

En este trabajo se planea implementar un sistema de control en tiempo real en instituciones de educación inicial utilizando beacons, smartwatch y smartphone. Se inició con la recolección de los requerimientos y se está procediendo a realizar el modelo del software a partir de los requerimientos con ayuda de las especificaciones formales para más adelante realizar el desarrollo propiamente dicho. Para garantizar la correcta especificación de requerimientos y evitar ambigüedades se utilizará el lenguaje VDM++ para modelar el sistema utilizando métodos formales que nos ofrece la herramienta, asimismo se realizará la validación de nuestro modelo.

El resto de este documento está organizado de la siguiente manera. En la sección 2 se presentan algunos trabajos relacionados. La Sección 3 describe lo que es Especificación Formal y su lenguaje VDM++. La sección 4 nos muestra sobre la tecnología de los beacons y el Bluetooth Low Energy (BLE). La Sección 5 describe el caso de aplicación, la arquitectura de nuestro sistema control de asistencia mediante beacons. Al final del documento son presentadas nuestras conclusiones.

II. TRABAJOS RELACIONADOS

El uso de tecnologías para la localización no es nuevo en cuanto a investigación.

Raghavan, Ananthapadmanaban, Sivamurugan y Ravindran [3] describen un método preciso de localización de un robot móvil usando bluetooth, el cual les permite tener más funcionalidad en un área determinada, lo cual ayuda en tareas como entrega de oficina, operaciones de rescate, etc. El robot ya conoce las localizaciones a través de un mapa estático, que fue aprendido mediante los beacons. Este robot está en constante consulta con los beacons.

Chawathe [4] describe un método que determina la ubicación de un dispositivo móvil en un ambiente cerrado usando beacons. Se usan varias técnicas para lograr la localización, triangulación, trilateración, multilateración y métodos basados en células. Los beacons están conectados a una fuente de alimentación y un dispositivo USB.

Lazik, Rajagopal, Shih, Sinopoli y Rowe [5] presentan un sistema de localización acústica utilizando beacons con ultrasonido para la mejora de exactitud de rango, con el uso de tres beacons se requiere una calibración con un dispositivo

móvil que tiene que recorrer puntos clave en el entorno. Este tiene un error de distancia euclidiana de 16.1 cm.

Shota, Michitoshi, Erjing y Masaru [6] [7] muestran un sistema de gestión de estudiantes mediante el uso de dispositivos Bluetooth Low Energy (BLE) el cual soluciona la problemática de las SmartCard que los estudiantes utilizaban para marcar asistencia y luego retirarse de su salón de clases, además del tiempo empleado por los estudiantes para el escaneo de sus tarjetas en un mismo terminal. Es por ello que se realiza una aplicación para dispositivos Android la cual se encarga de recibir señales de los beacons de manera periódica, garantizando así la asistencia de los estudiantes a sus centros de estudios.

A diferencia de los trabajos expuestos nuestra propuesta utiliza los beacons, smartphone y smartwatch para el control en tiempo real de los niños de educación inicial. Además se utilizó la herramienta VDM++ Toolbox para el modelado del sistema y se comprobó su correcta implementación usando la verificación y validación con el lenguaje VDM++.

III. ESPECIFICACIÓN FORMAL Y VDM++

A. *Definición*

Una especificación formal se define como una notación matemática para describir de manera precisa las propiedades que un sistema debe tener, sin importarle la forma en como son obtenidas dichas propiedades. Debido a su naturaleza matemática se puede describir lo que el sistema debe hacer sin decir cómo se va a hacer lo cual aumenta la confianza en el sistema al eliminar la ambigüedad en los requisitos del sistema y aumentar la confianza en el mismo [8]. Para los sistemas complejos en los que la corrección del sistema es muy importante, no se puede evitar el uso de lenguajes formales, para ello existen lenguajes para realizar especificaciones formales tales como: Astral [9], Lustre [16], Z [10], Object Z [1], Raise [2], CSP [12] y VDM++. Para la realización de nuestra especificación formal se utilizó el leguaje VDM++ [13] [8].

B. *VDM++*

VDM++ es un lenguaje de especificación formal para la descripción y desarrollo de sistemas informáticos. Sus descripciones formales usan notación matemática para proporcionar una notación precisa de la función prevista de un sistema. Tales descripciones se construyen en términos de modelos de un estado subyacente con una colección de operaciones que se especifican como pre-condiciones y post-condiciones. Los diseños de VDM++ están guiados por un número de obligaciones de prueba cuya descarga establece la exactitud del diseño ya sea mediante la recopilación de datos o la descomposición de la operación. Por lo tanto, se puede ver que VDM++ aborda las etapas de desarrollo desde la especificación hasta el código. La principal estructura en VDM++ es la clase [14] [15].

C. *Clase*

En VDM++ un modelo consiste en un conjunto de especificaciones de Clase. Una especificación de clase tiene los siguientes componentes que se detallan en la figura 1 [8]:

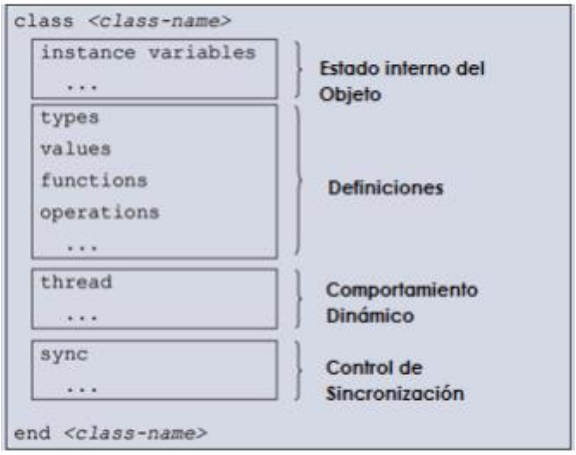

Fig. 1 Esquema de la Clase.

a) Encabezado: El encabezado contiene el nombre de la clase e información de herencia. Una o varias herencias son permitidas.
b) Variables de instancia: Las variables de instancia representan el estado de un objeto el cual consta de un conjunto de variables los cuales pueden ser *bool* o *nat*, así como también tipos complejos como *sets. Map, etc*.
c) Operaciones: Son métodos de clase que pueden modificar el estado, pueden ser definidos implícitamente, usando pre y post condiciones o implícitamente usando declaraciones imperativas y opcionalmente pre y post condiciones.
d) Funciones: Las funciones son similares a las operaciones excepto que el cuerpo de una función es una expresión en lugar de una declaración imperativa. Las funciones no están permitidas tener variables de instancia, estas son puras y libres de efectos secundarios.
e) Sincronización: Las operaciones en VDM++ son síncronas.
f) Hilos: Un hilo es una secuencia de instrucciones que se ejecutan hasta el final en el que el hilo muere. Es posible especificar hilos que nunca terminan.

D. *Tipos y Operaciones*

Al igual que en otros lenguajes se pueden definir tipos de datos en VDM++ para su posterior uso, al igual que definir operaciones; ambos tienen una estructura y son declarados respectivamente en los componentes de types y operations.

En la Tabla 1 podemos ver los operadores y tipos de datos usados en el desarrollo del modelo.

Operador	Nombre	Tipo
a and b	Conjuntion	bool* bool → bool
c1 = c2	Equal	char*char →bool
c1 <> c2	Not equal	char*char →bool
len l	Length	seq of A → nat
l(i)	Sequence App	seq of A*nat1 → A

Tabla. 1 Operaciones y tipos.

E. *Herramienta VDM++ Toolbox*

VDM++ Toolbox [16] es un conjunto de herramientas que permite el análisis y desarrollo de modelos precisos en sistemas informáticos. Cuando se utiliza en las primeras etapas del desarrollo del sistema, estos modelos pueden servir como especificaciones del sistema o como ayuda para verificar la consistencia y la integridad de los requisitos del usuario.
VDM++ Toolbox, proporciona una gama de herramientas para el control automático y la validación de modelos expresados en VDM++ Toolbox antes de la implementación. Estos van desde la sintaxis tradicional y herramientas de verificación de tipos hasta un poderoso intérprete que ejecuta modelos bajo petición y realiza la comprobación de coherencia automática durante la ejecución. Ver figura 2.

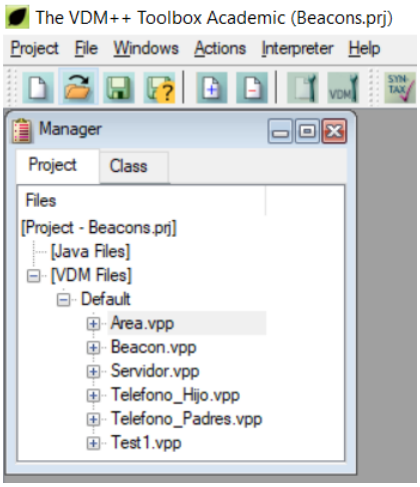

Fig. 2 Herramienta VDM++ Toolbox.

IV. BEACONS

A. *Definición*

Los beacons son pequeños dispositivos basados en la tecnología Bluetooth Low Energy (BLE) que emiten una señal que identifica de forma única a cada dispositivo [17]. Esta señal puede ser recibida e interpretada por otros dispositivos por ejemplo smartphone o smartwatch, así como se muestra en la figura 3. Cuando la señal emitida por los beacons es recibida por el smartphone o smartwatch, se puede conocer la distancia a la que se encuentra el beacon y el dispositivo receptor.

Fig. 3 Smartphone y beacon.

B. *Bluetooth Low Energy*

BLE es una versión de baja energía de Bluetooth especificada en la versión 4.0. Los dispositivos Bluetooth Low Energy funcionan en la banda libre de licencia de 2,4 GHz, por lo que comparten las mismas características de propagación en interiores que los transceptores WiFi de 2,4 GHz [18].

El modo de señalización o publicidad, permitido en el estándar BLE, permite un mensaje muy corto y no solicitado a tasas de actualización muy flexibles. Estos mensajes puedan ser usados para permitir que un dispositivo detecte la proximidad a una ubicación específica en función de la intensidad de la señal recibida (Received Signal Strength RSS por sus siglas en ingles). De esta forma, se pueden proporcionar al usuario los activadores, anuncios, comprobantes e información específicos de la ubicación [19].

C. *Características*

Un Beacon posee 5 servicios: 3 servicios BLE estándar identificados por un Universally Unique Identifier (UUID por sus siglas en ingles) de 16 bits y 2 servicios específicos de Beacon identificados por un UUID de 128 bits [19].
Las características pueden ser de cuatro tipos:

a) legible, fijo: como el número de serie o modelo único de la baliza;
b) legible, variable: como el nivel de la batería;
c) legible, escribible: como los parámetros de baliza (como el UUID de proximidad);
d) escribible: como la clave de acceso.

V. CASO DE APLICACIÓN

A. *Descripción del problema*

En los últimos años en el Perú se ha registrado un aumento en los accidentes de tránsito de movilidades escolares, secuestros desapariciones de menores, Según las denuncias realizadas en el Ministerio Público en el año 2016 se han registrado un total de 1144 denuncias por trata de personas de las cuales se han reportado como presuntas víctimas un total de 228 niños menores de 13 años [20].

Así mismo la Policía Nacional del Perú informó que en el año 2016 se registraron denuncias por trata de personas de un total de 298 personas de las cuales 199 víctimas son menores de edad siendo más vulnerables las mujeres ya que de las 199 víctimas registradas 176 son casos de mujeres [20].

Uno de los principales temores de los padres de familia son los accidentes de tránsito como se muestra en figura 4, así como también los posibles secuestros, asaltos o que el niño pueda extraviarse.

Fig. 4 Accidente de tránsito entre una unidad de transporte público y una unidad de servicio escolar

B. *Propuesta*

Se propone realizar un sistema de control en tiempo real para instituciones de educación inicial, usando smartphone, smartwatch y beacons colocándolas en las movilidades escolares y en los salones de clase como la figura 5 para así obtener información sobre la ubicación de los niños. Realizando la especificación formal con el lenguaje VDM++ y validación con la herramienta VDM++ toolbox.

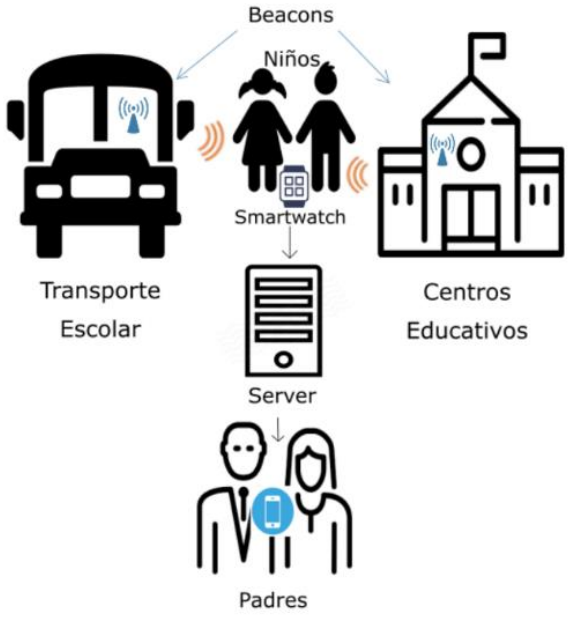

Fig. 5 Ubicación de los dispositivos beacon.

C. *Requerimientos*

De acuerdo a la propuesta y al modelo se toma el control de los celulares de los niños a través de un servidor que obtiene los datos de los teléfonos inteligentes y beacons conectados a este último, y de acuerdo a los beacons poder extraer un área de ubicación del niño y poder comunicárselo a los teléfonos celulares de los padres usando una clave de forma que sea correcta la información ofrecida a los padres.

Los siguientes requerimientos se desprenden de la anterior descripción.

R1: Un solo servidor.
R2: Un Beacon mínimo reconocido por teléfono de niño.
R3: Un smartwatch de niño mínimo
R4: Un teléfono de padre minino
R5: Un area minina
R7: Key de niño debe de ser igual al del padre.
R8: Numero de caracteres de key >= 8 caracteres

D. Modelo (Diagrama de Clases, VDM++)

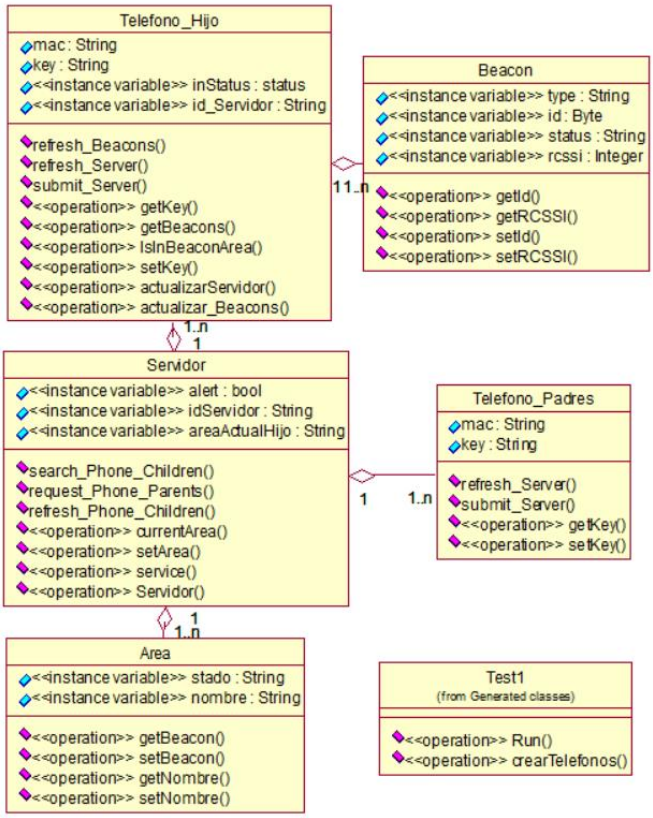

Fig. 6 Diagrama de clases.

E. Especificación formal VDM++

Haciendo uso de la herramienta VDM++ Toolbox se realizaron las especificaciones definiendo las clases de las figuras 7, 8, 9, 10 y 11 de manera formal con el lenguaje VDM++.

El requerimiento R1 se satisface definiendo la clase Servidor de la figura 7, en la línea 18 y 19 especificamos que las áreas que se tengan sincronizadas no estén vacías cumpliendo con el requerimiento R5.

En las líneas del 20 al 23 de la figura 7 se especifica que los teléfonos celulares pertenecientes a los niños y a los padres no estén vacíos, pidiendo que al menos exista uno en cada lado cumpliendo con el requerimiento R3 y R4.

En el campo de las operaciones de la clase Servidor se implementa el constructor de la clase, la operación service (línea 31) que realiza la comparación de las claves de los teléfonos celulares de los niños y los padres, en la línea 36 la operación setArea agrega un área con un beacon y la operación currentArea que busca el área de la ubicación de acuerdo a los beacons que la clase Telefono_Hijo posea.

```
6: class Servidor
7: types
8:   public String = seq of char;
9: instance variables
10:   public areas : seq of Area := [];
11:   public idServidor : String;
12:   public telefonoPadres : seq of Telefono_Padres := [];
13:   public telefonoHijo : seq of Telefono_Hijo := [];
14:   public alert : bool;
15:   public areaActualHijo : String;
16:   inv
17:     len idServidor = 8;
18:   inv
19:     areas <> [];
20:   inv
21:     telefonoPadres <> [];
22:   inv
23:     telefonoHijo <> [];
24: operations
25: public Servidor: Telefono_Hijo * Telefono_Padres ==> Servidor
26: Servidor(pc,pp) ==(
27:   telefonoHijo :=[pc];
28:   telefonoPadres :=[pp];
29:   alert := false;
30: );
31: public service: Telefono_Hijo * Telefono_Padres ==> ()
32: service(pc,pp) ==(
33:   if (pc.getKey() = pp.getKey())
34:   then alert := true;
35: );
36: public setArea : String * Beacon ==>()
37: setArea(s,b)==(
38:   dcl area : Area:=new Area();
39:   area.setNombre(s);
40:   area.setBeacon(b);
41:   areas :=[area];
42: );
43: public currentArea : ()==>()
44: currentArea()==(
45:   dcl beaconHijo : seq of Beacon :=telefonoHijo(1).getBeacons();
46:   if((areas(1).getBeacon().getRCSSI()=beaconHijo(1).getRCSSI()) and
47:     (areas(1).getBeacon().getId()=beaconHijo(1).getId()))
48:   then areaActualHijo :=areas(1).getNombre();
49: );
50: end Servidor
```

Fig. 7 Especificación de la clase Servidor

La clase Área de la figura 8 define las ubicaciones geográficas y contiene uno o varios beacons para poder identificarlos una vez se tenga la conexión con los teléfonos celulares de los niños y con el servidor.

En las líneas del 16 al 33 en el componente de las operaciones de la clase Área se implementa las operaciones de modificación y de obtención de datos.

```
6: class Area is subclass of Servidor
7: types
8:  public String = seq of char;
9:  public Byte = seq of char;
10: instance variables
11:  public nombre : String;
12:  public stado : String;
13:  public idBeacons :seq of Beacon := [];
14:  inv
15:   len nombre > 0;
16: operations
17:  public setNombre: String ==> ()
18:  setNombre(s)==(
19:   nombre := s;
20:  );
21:  public getNombre: () ==> String
22:  getNombre()==(
23:   return nombre;
24:  );
25:  public setBeacon: Beacon ==>()
26:  setBeacon(b)==(
27:   idBeacons :=[b];
28:  );
29:  public getBeacon: ()==>Beacon
30:  getBeacon()==(
31:   return idBeacons(1);
32:  );
33: end Area
```

Fig. 8 Especificación de la clase Área.

```
6: class Telefono_Padres
7: types
8:  public String = seq of char;
9: instance variables
10:  public key : String;
11:  public mac : String;
12:  inv
13:   len key >= 8;
14: operations
15:  public getKey:() ==> String
16:  getKey() ==(
17:   return key;
18:  );
19:  public setKey: String ==> ()
20:  setKey(s) ==(
21:   key :=s;
22:  );
23: end Telefono_Padres
```

Fig. 9 Especificación de la clase Telefono_Padres.

En las líneas 16 y 17 de la figura 10 y las líneas 12 y 13 de la figura 9 se especifica que las longitudes de los key sean mayores o iguales a 8 cumpliéndose el requerimiento R8.

El requerimiento R2 se especifica en las líneas 22 y 23 de la figura 10 pidiendo que la cantidad de beacons reconocido por el celular del niño sea mayor o igual a la cantidad de 1.

```
6: class Telefono_Hijo is subclass of Beacon
7: types
8:  public String = seq of char;
9:  public status = <onLine>|<offLine>
10: instance variables
11:  public key : String;
12:  public mac : String;
13:  public inStatus : status;
14:  public beacons : seq of Beacon :=[];
15:  public id_Servidor : String;
16:  inv
17:   len key >= 8;
18:  inv
19:   len mac = 16;
20:  inv
21:   len id_Servidor = 8;
22:  inv
23:   len beacons >= 1;
24: operations
25:  public actualizarServidor: status ==>()
26:  actualizarServidor(s) ==(
27:   inStatus := s;
28:  );
29:  public actualizar_Beacons: seq of Beacon ==> ()
30:  actualizar_Beacons(b) ==(
31:   beacons := b;
32:  );
33:  public IsInBeaconArea:() ==> bool
34:  IsInBeaconArea() ==(
35:   return len beacons <> 0;
36:  );
37:  public getBeacons:() ==> seq of Beacon
38:  getBeacons() ==(
39:   if(IsInBeaconArea() and (inStatus=<onLine>))
40:   then return beacons;
41:   return beacons;
42:  );
43:  public getKey:() ==> String
44:  getKey() ==(
45:   return key;
46:  );
47:  public setKey: String ==> ()
48:  setKey(s) ==(
49:   key :=s;
50:  );
51: end Telefono_Hijo
```

Fig. 10 Especificación de la clase Telefono_Hijo.

En la clase Telefono_Hijo ver figura 10 en las líneas del 24 al 51 en la declaración de la operación se implementa las características de las comunicaciones que se tienen con los

servicios (clase Servidor) en la primera operación actualizarServidor (línea 25) se maneja la variable de estados de manera que se pueda saber si el teléfono esta conectado o no al servidor.

En la línea 29 el método actualizar_Beacon se cambia el valor de la variable beacons, este método será llamado cada vez que el teléfono encuentre nuevos beacons en el área actual de su ubicación.

El método getBeacons (línea 37) retorna la lista de Beacons que detecto el teléfono con la condición de haber detectado al menos 1 y que es el estado de la conexión con el servidor sea "online".

Se modela la Clase Beacon en la figura 11 con las características y propiedades que este posee añadiéndole estados (línea 15) para un manejo adecuado en la diferenciación entre los beacons.

```
6: class Beacon
7: types
8:   public String = seq of char;
9:   public Byte = seq of char;
10:  public Integer = seq of nat;
11: instance variables
12:  public id : Byte;
13:  public type : String;
14:  public rcssi : Integer;
15:  public status : String;
16:  inv
17:    len id = 8;
18: operations
19:  public
20:  getRCSSI:() ==> Integer
21:  getRCSSI() ==(
22:    return rcssi;
23:  );
24:  public
25:  getId:() ==> Byte
26:  getId() ==(
27:    return id;
28:  );
29:  public
30:  setRCSSI:(Integer) ==> ()
31:  setRCSSI(rc) ==(
32:    rcssi:=rc;
33:  );
34:  public
35:  setId:(Byte) ==> ()
36:  setId(b) ==(
37:    id:=b;
38:  );
39: end Beacon
```

Fig. 11 Especificación de la clase Beacon.

F. Validación del modelo y análisis del resultado

En la figura 12 se muestra la validación del sistema realizado en VDM++ con la cobertura de las operaciones de todas las clases y sus funcionalidades en una clase de prueba Test1.

```
>> rtinfo vdm.tc
100%    2 Area`getBeacon
100%    1 Area`getNombre
100%    1 Area`setBeacon
100%    1 Area`setNombre
100%    1 Test1`Run
100%    1 Test1`crearTelefonos
100%    2 Beacon`getId
100%    1 Beacon`setId
100%    2 Beacon`getRCSSI
100%    1 Beacon`setRCSSI
100%    1 Servidor`service
100%    1 Servidor`setArea
100%    1 Servidor`Servidor
100%    1 Servidor`currentArea
100%    1 Telefono_Hijo`getKey
100%    1 Telefono_Hijo`setKey
100%    1 Telefono_Hijo`getBeacons
100%    1 Telefono_Hijo`IsInBeaconArea
100%    1 Telefono_Hijo`actualizarServidor
100%    1 Telefono_Hijo`actualizar_Beacons
100%    1 Telefono_Padres`getKey
100%    1 Telefono_Padres`setKey

Total Coverage: 100%
```

Fig. 12 Validación del modelo.

En la figura 13 se implementa la clase Test1 donde se define los componentes que usa el servidor tales como los teléfonos celulares de los niños, de los padres, los beacons (líneas 3 al 7) y el Área agregado en la línea 18 a través del servidor; se añade las claves de los teléfonos (líneas 25 y 26) y al beacon se añade el identificador y su valor RCSSI (líneas 12 y 13).

Los datos añadidos simulan una conexión exitosa establecida con el teléfono inteligente del niño y el servidor, donde el servidor obtiene los beacons conectados al teléfono del niño, se crea un Área para luego determinar el Área actual del niño(línea 19) según el beacon conectado y la confirmación de las claves de los teléfonos celulares(línea 14).

```
1: class Test1
2: instance variables
3:  telefonoHijo    : Telefono_Hijo;
4:  telefonoPadre : Telefono_Padres;
5:  servidor        : Servidor;
6:  beacon          : Beacon;
7:  beacons         : seq of Beacon :=[];
8: operations
9:  public Run: () ==> ()
10:  Run() ==(
11:    crearTelefonos();
12:    beacon.setRCSSI([1,2,2]);
13:    beacon.setId("12af");
14:    servidor.service(telefonoHijo,telefonoPadre);
15:    telefonoHijo.actualizarServidor(<onLine>);
16:    beacons :=[beacon];
17:    telefonoHijo.actualizar_Beacons(beacons);
18:    servidor.setArea("School",beacon);
19:    servidor.currentArea();
20:  );
21:  public crearTelefonos: () ==> ()
22:  crearTelefonos() ==(
23:    telefonoHijo  :=new Telefono_Hijo();
24:    telefonoPadre :=new Telefono_Padres();
25:    telefonoPadre.setKey("1q2w3e");
26:    telefonoHijo.setKey("1q2w3e");
27:    beacon  :=new Beacon();
28:    servidor:=new Servidor(telefonoHijo,telefonoPadre);
29:  );
30: end Test1
```

Fig. 13 clase Test1.

CONCLUSIONES

En este artículo, se describe un sistema de control en tiempo real para instituciones de educación inicial que brinda a los padres de familia la posibilidad de conocer la ubicación de sus hijos a través de sus smartphone, los smartwatch de sus hijos y de los beacons colocados en los salones de clase y movilidad escolar. Los beacons son dispositivos basados en BLE que emiten señales que identifican de manera única a cada dispositivo. Los smartphone y smartwatch son dispositivos que reciben e interpretan las señales de los beacons permitiendo conocer la distancia a la que se encuentra el beacon y el dispositivo receptor, de esta manera el padre de familia podrá conocer la ubicación de su hijo ya sea en la movilidad escolar o si ya arribo a su institución educativa brindando tranquilidad a los padres.

Se ha modelado el sistema propuesto utilizando la herramienta VDM++ Toolbox, se comprobó su correcta implementación usando la verificación y validación con el lenguaje VDM++.

REFERENCIAS

[1] R. Duke, G. Rose and G. Smith, "Object-Z: A specification language advocated for the description of standards", *Computer Standards and Interfaces*, vol. 17, no. 5-6, pp. 511-533, September 1995.

[2] S. P. Miller and M. Srivas, "Formal verification of the AAMP5 microprocessor: A case study in the industrial use of formal methods". *In Industrial-Strength Formal Specification Techniques, Proceedings*, pp. 2-16. April 1995.

[3] A. N. Raghavan, H. Ananthapadmanaban, M. S. Sivamurugan and, B. Ravindran. "Accurate mobile robot localization in indoor environments using bluetooth. In Robotics and Automation", *IEEE International Conference*, pp. 4391-4396, May 2010.

[4] S. S. Chawathe, "Beacon placement for indoor localization using bluetooth. In Intelligent Transportation Systems", *11th International IEEE Conference*, pp. 980-985, October 2008.

[5] P. Lazik, N. Rajagopal, O. Shih, B. Sinopoli and A. Rowe, "A bluetooth and ultrasound platform for mapping and localization". In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pp. 73-84, November 2015.

[6] N. Shota, N. Michitoshi, Z. Erjing and K. Masaru. "Student attendance management system with bluetooth low energy beacon and android devices", *18th International Conference on Network-Based Information Systems*, pp.710-713, September 2015.

[7] B. Cüneyt and Ö. Mehmet. "A student attendance system based on beacon and smartphones equipped with bluetooth low energy technology", *International Journal of Informatics Technologies*, vol. 9, no. 3, pp. 249, September 2016.

[8] S. Rabia, R. Iram, A. Q. Muhammad "Formal Specification Languages for Real-Time Systems", *Information Technology (ITSim), 2010 International Symposium*, vol. 3, pp. 1642-1647, June 2010.

[9] K. Weldemariam, R. A. Kemmerer and A. Villafiorita, "Formal specification and analysis of an e-voting system. In Availability, Reliability, and Security", *Availability, Reliability, and Security, 2010. ARES'10 International Conference on. IEEE*, pp. 164-171, February 2010.

[10] N. Halbwachs, P. Caspi, P. Raymond and D. Pilaud, "The synchronous data flow programming language LUSTRE", *Proceedings of the IEEE*, vol. 79, no. 9, pp. 1305-1320, September 1991.

[11] J. M. Spivey, *Understanding Z: a specification language and its formal semantics*. Cambridge University Press, 1988.

[12] A. F. Ates, M. Bilgic, S. Saito and B. Sarikaya, "Using timed CSP for specification verification and simulation of multimedia synchronization". *IEEE journal on selected areas in communications*, vol. 14, no. 1, pp. 126-137, January 1996.

[13] J. M. Spivey, *The Z Notation: A reference manual*, Nueva Jersey: Prentice Hall, 1992.

[14] C. B. Jones, *Systematic Software Development Using VDM*, Englewood Cliffs: Prentice Hall, 1990.

[15] R. Faragher and R. Harle, *"Location Fingerprinting With Bluetooth Low Energy Beacons"*, *IEEE Journal On Selected Areas In Communications*, vol. 33, no. 11, pp. 2418-2428, November 2015.

[16] Csk Corporation, *VDMTools User Manual (VDM++)* Ver 1.1.

[17] S. S. Chawathe, "Beacon placement for indoor localization using bluetooth. In Intelligent Transportation Systems", *11th International IEEE Conference*, pp. 980-985, October 2008.

[18] B. Sig, "Bluetooth specification", ver. 4.0, vol. 0, Available at http://www.bluetooth.org, June 2010.

[19] M. Hultgren, D. Papathanopoulos "Evaluating the usage of Bluetooth Low Energy Beacons and Smartphones for Indoor Positioning Systems", May 2015.

[20] INEI, Denuncias de Trata de Personas Presuntas víctimas y presuntos(as) imputados(as), 2010 – 2016.