2016

Proyecto de Grado

Modelando los contextos

Alumnos:

- Andrés Macri
- Eleonora Etchemendy

Tutora:

- Regina Motz

Sitio Web:

- www.contextmf.com

Carrera de Ingeniería en Computación
Facultad de Ingeniería
Universidad de la República
Montevideo – Uruguay
Noviembre de 2016

Montevideo
2016

Resumen

El concepto de Contexto cada vez se tiene más en cuenta a la hora de desarrollar aplicaciones.
Hoy en día existen varias herramientas que de un modo u otro utilizan el contexto, ya sea obteniendo la información relevante para el usuario, modelando el mismo o informando sobre el contexto a las aplicaciones.

En este documento se presenta el análisis realizado durante el proyecto de grado para conceptualizar un modelo de contexto que integre las diferentes definiciones del mismo que se encuentran actualmente en la bibliografía.

El trabajo presenta también un estudio comparativo de herramientas que ayudan a captar información de contextos, o que colaboran con el desarrollo de aplicaciones sensibles al contexto.

Se reconoce que sería de gran utilidad contar con una herramienta que reúna las principales funcionalidades de las herramientas mencionadas, y para esto se implementó un Framework llamado Context Modeling Framework. Una de las características principales de este Framework es su usabilidad para personas sin conocimientos de desarrollo de software.

En el presente documento, se describe la arquitectura del Context Modeling Framework, el cual consiste en un proyecto base para desarrollar aplicaciones sensibles al contexto. Se presentan ejemplos de cómo crear contextos utilizando este Framework, ya sea mediante un archivo XML o utilizando una interfaz gráfica (wizard).El Framework definido permite establecer los elementos que deben ser medidos para controlar el contexto y definir las funciones limite que hacen cambiar de contexto.

Context Modeling Framework se desarrolló con herramientas open source, como Java, JSF(Prime Faces), Log4J, etc.

También se presenta documentación sobre las pruebas realizadas sobre la herramienta, ilustrando su funcionamiento.

Por último se exponen las conclusiones, en las cuales se muestra un análisis del trabajo realizado, tanto para el modelado de contextos como para la herramienta implementada.

Palabras Clave

Contexto, Modelo, Modelado, Entorno, Fuente de dato, Interprete, Framework, Wizard.

Contenido

Índice de Figuras

1. Introducción

Como seres humanos cada vez que actuamos o hablamos, lo hacemos de distinta manera según el contexto en el que nos encontremos. La adaptación al contexto en nuestra vida cotidiana es muy fácil para nosotros y la mayoría de las veces se trata de una acción subconsciente. Cambiamos nuestro comportamiento en función de la situación, en relación al lugar y con quien nos encontremos. El contexto nos permite discriminar qué es importante y qué no lo es, permite enriquecer nuestros conocimientos sobre una determinada situación, aprovechando recuerdos y experiencias relacionadas. A través del contexto somos capaces de distinguir la enorme cantidad de información que nos rodea en cualquier punto dado en el tiempo.

La naturaleza de la información que nos rodea no es diferente a los datos que se pueden encontrar en aplicaciones ubicuas, móviles o web. Como en la vida real, grandes cantidades de información se encuentran disponibles en un número casi ilimitado de fuentes. Para que las aplicaciones sensibles al contexto sean cada vez más exitosas, dependerán enteramente de su capacidad para entregar la información correcta al usuario correcto en el momento correcto. El contexto es un poderoso instrumento para lograr la difusión de la información relevante que esas aplicaciones demandan.

El objetivo de este trabajo consiste en estudiar las formas de representación de los contextos en informática y ofrecer una representación genérica de los contextos. Trabajar sobre una representación genérica de los contextos permitiría concentrarse en las implementaciones y aplicaciones que los utilizan brindándoles una firma transparente de usarlos.

Se plantean algunas preguntas que se contestarán a lo largo del documento.

¿Qué son los contextos? ¿Son importantes en una aplicación? ¿Cambia el comportamiento de una aplicación? ¿Cómo se relacionan los contextos? ¿Cómo se relacionan las entidades de un contexto?

1.1 Motivación

En los últimos años, la dependencia de los seres humanos por determinadas aplicaciones ha crecido y continúa haciéndolo fuertemente, lo que lleva a una evolución constante de las mismas. Estas aplicaciones tienen como objetivo ayudar, facilitar y mejorar la calidad de vida de las personas. Para estas es fundamental comprender el entorno y poder abstraerse del mismo, quedándose con la información relevante para su cometido. Para ello es necesario disponer de la información del contexto, es decir que es necesario saber quiénes la necesitan, en donde se encuentran, para que la necesitan, etc. Entender las características del entorno permite ofrecer información relevante en el momento oportuno.

Definir un modelo que ayude a las aplicaciones a abstraerse de la complejidad del entorno, modelando toda esta información adecuadamente, puede dotar a las aplicaciones de una visión simplificada del mundo real dónde se encuentran, y actuar en consecuencia. De esta manera se contribuye a la evolución de las mismas, y por lo tanto colaborar con su objetivo de ayudar, facilitar y mejorar la vida de las personas.

Estas aplicaciones permiten a los seres humanos mejorar la calidad de vida desde muchos aspectos, por ejemplo:

- Una casa inteligente que regula la temperatura según la necesidad y la presencia de personas, manejando el encendido de electrodomésticos según actividades de sus moradores.
- Rutas inteligentes capaces de señalizara los conductores diferentes alertas dependiendo de los sucesos ocurridos en el tránsito.
- Vehículos inteligentes, que interpretan la dificultad de las rutas y del tránsito forzando cierto desempeño por ejemplo velocidad máxima/mínima, radio de curvas, frenadas bruscas, etc.
- Contenedores de residuos que avisan cual es el momento adecuado para su vaciado dependiendo de su contenido y condiciones climáticas.

Teniendo la representación genérica, es decir el modelo de los contextos, el siguiente paso consiste en interpretarlos. Para esto, se decidió implementar un Framework que permite a los creadores de aplicaciones, abstraerse del razonamiento, modelado e interpretación de los contextos, para que se puedan enfocar en el desarrollo de su negocio.

1.2 Objetivos

De las motivaciones antes mencionadas, se desprenden los siguientes objetivos:

- Comprender que es un contexto, y su importancia:
  - Es un concepto fundamental a la hora de tomar decisiones. Uno de los principales objetivos es entenderlo, y comprender su rol dentro de las aplicaciones sensibles al contexto.
- Realizar un estudio del estado del arte de las herramientas existentes utilizadas para la representación de los contextos:
  - Se busca encontrar una o varias herramientas que permitan modelar los contextos, o que ayude con este propósito.
- Encontrar una forma de representar genéricamente los contextos:
  - Se busca abstraer al usuario de la complejidad del entorno, encontrando un modelo para representar los contextos, y de esta manera colaborar con el desarrollo de aplicaciones sensibles al contexto, y en particular dentro de la computación ubicua.
- Implementar un Framework que modele de forma genérica los contextos:
  - Además de encontrar una forma de representar los contextos, se busca desarrollar una herramienta que lo haga. De esta manera se puede ayudar en el desarrollo de aplicaciones sensibles al contexto, brindando una capa de abstracción que permita al usuario centrarse en lo que realmente desean implementar, es decir en su negocio.
  - Otro objetivo del desarrollo de este Framework, consiste en darle la capacidad no solo de modelar, sino de interpretar los contextos. Por interpretar se entiende evaluar los elementos de los contextos y tomar decisiones en caso de ser necesario.

1.3 Organización del Documento

El resto de este documento, se encuentra organizado en 11 capítulos, en los cuales se muestran los principales puntos del proyecto.

Capítulo 2, **Estado del arte:** Contiene documentación relacionada con la investigación introductoria a los contextos, a la necesidad de modelarlos, mostrando una visión general del tema en cuestión.

Capítulo 3**, Modelo encontrado:** En esta sección se encuentra documentación sobre el modelo definido.

Capítulo 4, **Solución Planteada:** Propuesta de solución y decisiones adoptadas.

Capítulo 5, **Desarrollo de la herramienta:** Contiene información sobre la implementación realizada.

Capítulo 6, **Forma de uso y Funcionamiento:** En este capítulo se explica su forma de uso, y se ilustran ejemplos del Framework trabajando.

Capítulo 7, **Conclusiones y trabajo a futuro:** Finalmente, se describen los aportes, dificultades y conclusiones obtenidas y el trabajo futuro.

Capítulo 8, **Glosario**: Se definen algunas de las palabras utilizadas

Capítulo 9, **Bibliografía**: Referencias de artículos, autores, imágenes utilizadas en el documento

Capítulo 10, **Anexo I:** En este anexo se ilustran ejemplos, explicando en profundidad y detalle cómo funcionan bajo el Framework.

Capítulo 11, **Anexo II:** Contiene el manual de instalación para poder trabajar con el mismo.

2. Conceptos preliminares

En este capítulo se presentan los conceptos principales que involucran a los contextos y también a los modelos. Además se describen las arquitecturas y formas de modelado de los contextos, de herramientas investigadas que trabajan con modelos de contextos.

2.1 Definiciones de contextos

Existen muchas definiciones de contexto, a continuación se cita la definición de la Real Academia Española:

Definición de contexto según la Real Academia Española:

Del lat. contextus.

1. m. Entorno lingüístico del que depende el sentido de una palabra, frase o fragmento determinados.

2. m. Entorno físico o de situación, político, histórico, cultural o de cualquier otra índole, en el que se considera un hecho.

3. m. desus. Trabazón, composición o contenido de una historia o discurso.

4. m. desus. Enredo, maraña o unión de cosas que se enlazan y entretejen.

La definición de la real academia no define al contexto como un elemento que pueda ser utilizado en computación, por esto se buscaron definiciones de contexto más específicas. A continuación se citan tres definiciones consideradas relevantes para la computación y se las relaciona con la definición de la real academia española.

SCHILIT (1), Identifica al *entorno informático* (procesadores, dispositivos, pantallas, etc), *entorno de usuario* (ubicación, situación social, las personas cercanas, etc) y *el medio físico* (por ejemplo, la iluminación y el nivel de ruido) como los componentes clave de contexto.

DEY Y ABOWD (2), describe el contexto de la siguiente manera: "El Contexto es cualquier información que se puede utilizar para caracterizar la situación de una entidad. Una entidad es una persona, lugar u objeto que se considera relevante a la interacción entre un usuario y una aplicación, incluyendo el usuario y las propias aplicaciones. El contexto es típicamente la ubicación, la identidad y estado de las personas, grupos y objetos computacionales y físicos."

DOURISH (3), dice que el contexto no es información por sí misma, sino más bien una característica de la información existente. Cada objeto de datos puede ser "contextualmente relevante "a otro objeto para una actividad particular de un usuario. Es decir, no hay separación entre los objetos de contexto y aplicación normal de objetos.

Las definiciones de contexto antes mencionados son en su mayoría influenciadas por la comunidad informática ubicua, exceptuando a la definición obtenida de la real academia española, pero están obviamente relacionadas, ya que todas toman como punto de partida un “entorno físico o de situación” como dice la real academia española.

En el caso de ***Schilit*** se puede relacionar el entorno de usuario como lo hace la real academia española en el punto 2, *“Entorno físico o de situación, político, histórico, cultural o de cualquier otra índole, en el que se considera un hecho“,* ya que considera por entorno de usuario *“ubicación, situación social, y las personas cercanas”*

Dey y Abowd utiliza la interacción o unión como el punto 4, de la real academia española: *“unión de cosas que se enlazan y entretejen”.*

Dourish en cambio considera el contexto relacionándolo con los hechos *“actividad particular”*, lo cual también considera la real academia española en su punto 2 de la definición, donde considera por contexto al entorno considerando u hecho: *“Entorno físico o de situación, político, histórico, cultural o de cualquier otra índole, en el que se considera un hecho.”*

En lo que respecta a este trabajo, utilizando las definiciones antes mencionadas, y en base a la investigación realizada, se puede definir al contexto como:

“Información relevante y variable del entorno con un determinado objetivo, para un usuario en particular.”

Por **entorno** se considera el ambiente o al conjunto de características que definen al lugar en el que está situado el usuario indicado.
Por **información relevante** se considera a la información que afecta al usuario al momento de tomar decisiones para que se cumpla el objetivo determinado.
Además se considera que esta información es **variable**, ya que la misma puede variar en el tiempo.

2.2 Modelado de contextos

Según la real academia, un modelo es aquello que se toma como referencia para tratar de producir algo igual.

También se puede considerar a un Modelo, como una abstracción o una representación idealizada de un sistema de la vida real.

Los modelos tienen las siguientes utilidades:

- Descripción de una situación de forma genérica, con el fin de que otras personas puedan comprenderlo.
- Exponer el relacionamiento de elementos simples, con otros más complejos.
- Permite representar o visualizar aspectos comunes y diferencias en distintos puntos de vista sobre un fenómeno.

Por **Modelar** se entiende dar forma a algo, o también encontrar un modelo para representar algo.

Por **Modelado** se entiende el proceso de creación de una representación o el proceso de construcción del modelo de un objeto de estudio.

Por **"Modelado de Contextos"** se puede decir que es el proceso de creación de un modelo para los contextos.

En el capítulo 3, se verá el proceso realizado para poder modelar los contextos, y también la representación o "Modelo" utilizada.

2.3 Clasificación de modelos

En esta sección se van a describir los modelos más utilizados para modelar los contextos. Estos se clasifican por el esquema de las estructuras de datos que se utilizan, para intercambiar información del contexto en cada sistema. Según el artículo A Context Modeling Survey (4), se pueden identificar los siguientes modelos.

- **Modelos Clave-Valor:**

  El modelo de pares (clave, valor) es la estructura de datos más sencilla para el modelado de información contextual. Se basa en grupos de duplas atributo-valor que se pueden codificar utilizando distintos lenguajes, por ejemplo en XML.

En particular, los pares clave-valor son fáciles de manejar, son acotados en cuanto a la hora de representar modelos complejos, es decir que no permiten representar modelos que cuentan con muchas relaciones.

- **Modelos Markup Scheme** (Modelos de esquema de marcado):

Un punto en común en todos los modelos de esquema de marcado, es que son una estructura de datos jerárquica que consiste en etiquetas de marcas con atributos y contenidos.

En particular, el contenido del marcado mediante las etiquetas es por lo general de forma recursiva definida por otras etiquetas.

Por lo general están basados en una serialización de un derivado de la norma genérica de lenguaje de marcado (Standard Generic Markup Language).

Entre los lenguajes utilizados para estos modelos, se pueden encontrar XML, RDF, etc.

- **Modelos gráficos**:

Un conocido instrumento de modelado de propósito general es el Unified Modeling Language (UML). UML tiene una componente gráfica muy fuerte, estos son los diagramas UML.

Debido a su estructura genérica, UML también es apropiado para modelar el contexto.

- **Modelos orientados a objetos:**

Los modelos orientados a objetos, utilizan el paradigma de la orientación a objetos.

Este enfoque realiza la construcción de los modelos de contextos, por medio de la identificación y la especificación de un conjunto de objetos relacionados.

- **Modelos basados en ontologías:**

Una Ontología, es una definición formal de tipos, propiedades, y relaciones entre entidades.

El objetivo de utilizar las mismas para realizar modelos, es facilitar la comunicación y la compartición de la información entre diferentes sistemas y entidades.

2.4 Tipos de Contextos

Los contextos pueden ser muy heterogéneos y por lo tanto existen varias categorizaciones, como lo muestra artículo “El contexto y su contexto” (5).

Una posible es por grados de aplicación, considerando sólo el contexto de personas, y en función de a quien aplica, se podría dividir de la siguiente manera:

- **Contexto individual:** información contextual que concierne a o describe el entorno de una persona o entidad concreta
- **Contexto de grupo:** información extraída de un grupo de personas (una familia, los pasajeros de un tren, un grupo de amigos, los asistentes a una conferencia); no se trata de la mera agregación de los contextos individuales sino que es necesario considerar qué propiedades de contexto propagar al grupo entero, y si la existencia del grupo provoca la aparición de nuevas propiedades no aplicables a título individual.
- **Contexto colectivo:** se distingue del contexto de grupo en su agregación a gran escala. Son contextos de aplicación a poblaciones grandes, y en ellos lo importante es determinar qué rasgos (con el margen de error considerado) son generalizables a las poblaciones objeto de análisis.

Otra posible clasificación es por fuentes de contexto (de dónde se extrae) Así se pueden enumerar una serie de rasgos que componen el contexto de una persona (los contextos aplicados a grupos o colectivos recolectarían estos tipos de cada uno de sus miembros):

- **Contexto espacial**: Supone, en primera instancia la localización física del usuario, y en segunda instancia tanto la interpretación semántica (“está en casa”, “en el trabajo”, “en el coche”) como el marco de ese contexto espacial (*qué hay alrededor*, *quién hay cerca del usuario*).
- **Contexto temporal**: De nuevo en primera instancia es el instante de tiempo actual, y una segunda evaluación tiene las dos vertientes de interpretación semántica (“es por la mañana en día laborable”, “es la hora de sacar a pasear al perro”) y de marco temporal a corto plazo (“queda media hora para que salga del trabajo”, “ha comido hace poco”).
- **Contexto ambiental**: se puede considerar como una extensión del contexto físico, y engloba variables más generales como el tiempo atmosférico (*temperatura*, *si hace sol*, *si llueve*), iluminación, ruido ambiente, contaminación, etc
- **Contexto personal:** Comprende el estado de la persona en el momento de evaluar el contexto. Se refiere por tanto a aspectos como la actividad que está desarrollando (*trabajando*, *leyendo*, p*racticando un deporte*), el estado de ánimo (lo que se denomina “mood” en inglés), circunstancias físicas (*si tiene medio de transporte propio*, *si va muy abrigado o poco*).
- **Contexto social:** Es el que extiende el contexto involucrando a la parte de la red social del usuario que está activa (o es relevante) en cada momento; cada uno de los individuos en este segmento de red social aporta su contexto (espacial, temporal, ambiental, personal) al conjunto total.

2.5 Fuentes de datos de los contextos

En general hay diferentes fuentes primarias de contexto, a partir de las cuales se puede inferir contexto de más alto nivel. Entre esas Fuentes primarias pueden citarse:

Contexto espacial. La posición del terminal (base del contexto espacial) puede obtenerse mediante varias técnicas, las ordenamos en orden decreciente de precisión:

- GPS: da siempre la posición más exacta, pero a veces no está disponible (el dispositivo no lo tiene, o falla la cobertura) o activado (al consumir bastante energía, en muchos casos se desactiva).
- Sensores inalámbricos no celulares, basados en identificación de nodos de red inalámbrica de posición conocida (por ejemplo, estaciones WiFi, o dispositivos Bluetooth).
- Posicionamiento de celda, para dispositivos móviles: asignar la posición a partir de la estación base o triangulando entre estaciones a partir de medidas radio.
- Posicionamiento IP (basada en bases de datos que mapean direcciones IPs o rangos a áreas geográficas, útil especialmente en terminales fijos). Tienen precisión típicamente a nivel de ciudad.

  Existen APIs estandarizadas para que las aplicaciones y los terminales puedan enviar información de posición de forma normalizada (W3C Geolocation API, GSMA OneAPI, Google Geolocation)

- Contexto temporal: se puede recurrir a registros de actividad reciente del usuario (los mismos que para evaluar el contexto personal) para el pasado inmediato; el contexto del futuro inmediato puede extraerse de la agenda del usuario o estimarlo en función de su perfil (usando razonamientos/aprendizajes que infieran patrones temporales de comportamiento).
- Contexto ambiental: las redes de sensores, cada vez más abundantes e interconectadas, proporcionan información ambiental convenientemente geo-posicionada. Es lo que se ha llamado recientemente S*ensor Web* (red de sensores interconectados, y distribuidos geográficamente).
- Contexto personal: son fuentes primarias las interacciones del usuario con la aplicación, el terminal o con servicios de red, la información explícitamente proporcionada por el usuario, o procedimientos más avanzados como análisis facial o medida de constantes biométricas.
- Contexto social: a partir del grafo social (red social activa del usuario) obtenible mediante distintos procedimientos, se puede llegar a los usuarios relevantes en cada momento para extraer el contexto social.

2.6 Herramientas sensibles al contexto (context-aware)

Las aplicaciones Contex-Aware, o sensibles al contexto consideran el qué, quién, cuándo y dónde de una entidad, y usan esa información para determinar por qué se da una situación (la forma en la que se dispone una entidad en un determinado espacio). Una entidad es una persona, lugar u objeto que es considerado relevante para la interacción entre el usuario y una aplicación, incluyendo el usuario y la aplicación misma.

Son aplicaciones que adaptan automáticamente su comportamiento y configuración, dependiendo de las condiciones del entorno y de las preferencias del usuario de dicha aplicación.

A continuación se describen algunas herramientas existentes y sus respectivas arquitecturas, las cuales fueron estudiadas para evaluar los principales componentes de las mismas y así poder observar lo que tienen en común. Estas herramientas trabajan en la obtención, modelado e interpretación del contexto.

Es importante tener en cuenta en qué consisten estas actividades:

- Obtención: Implica la actividad mediante la cual la herramienta obtiene los elementos del contexto.
- Modelado: Como se explicó anteriormente, el proceso de creación de un modelo para los contextos.
- Interpretación: Consiste en utilizar los elementos del contexto ya obtenidos, y evaluarlos.

2.6.1 Context-toolkit

Context-Toolkit (6) tiene como objetivo facilitar el desarrollo de aplicaciones sensibles al contexto. Se basa en la definición de Dey y Abowd, fue creado por ellos en la universidad de Berkeley su objetivo es ayudar en la dificil tarea de la obtención de la información relevante para el usuario, con un objetivo determinado. Se basa en el concepto de widgets, que permiten separar los detalles de cómo se hacen las cosas o como se hacen en realidad.

Context-Toolkit(CTK) esta diseña en Java, se puede descargar a través de SourceForge (7), allí se encuentra disponible la última versión (2013-04-02) y acceso al repositorio de código a través de un usuario anónimo.

CTK tiene una guía de usuario. La cual contiene tutorial, documentación del código, manual de instalación. Además, se pueden visitar los foros del proyecto, alojados en SourceForge, para preguntas y respuestas de otros usuarios y desarrolladores CTK.

Los servicios de Context Toolkit (6)son:

- Encapsulación de sensores. Permite obtener información de distintos sensores, traducirla a información útil para la herramienta. Estos sensores pueden ser útiles para captar información. Por ejemplo sensores de movimiento, temperatura, etc.
- El acceso a los datos de contexto a través de una API de red.
- Abstracción de datos de contexto a través de intérpretes.
- Intercambio de datos de contexto a través de una infraestructura distribuida.
- El almacenamiento de datos de contexto, incluyendo el histórico, es decir que se guarda la información pasada.
- Acceso básico para la protección de la privacidad.

Componentes y Arquitectura

Para brindar estos servicios Context-toolkit se basa en la arquitectura que se muestra en la Figura 1 donde se pueden observar lo siguientes componentes:

- Widget: Recogen la información de los sensores y la envía al servidor o al intérprete. Son responsables de la separación de los detalles de detección de contexto y de determinar cómo utilizarla. Ellos abstraen los detalles de cómo el contexto se detecta a partir de las aplicaciones y otros componentes de contexto que necesitan el contexto. Los widgets pueden ser considerados similares a los widgets de interfaz gráfica que utiliza un usuario (por ejemplo, campos de texto), pero aplicados a los sensores en los sistemas de computación ubicua.
- Server: Recogen la información de los Widgets. Evitan que los usuarios se deban conectar a n máquinas para recoger n piezas de información, pueden agregar y combinar información de varios widgets para ofrecer una única pieza de información. Las aplicaciones deben hablar con los servidores en lugar de los widgets. A su vez, estos componentes abstraen a la aplicación de combinar diferentes informaciones de contexto.
- Intérprete: Los intérpretes son los encargados de recoger cierta información en conjunto e interpretarla. Reciben un contexto de entrada y como salida se obtiene un contexto con más información. Puede ser simple como tomar un nombre y devolver el teléfono correspondiente o más complejo y tomar el número de personas en una sala, el nivel del audio, la hora del día y devolver si es o no una reunión.

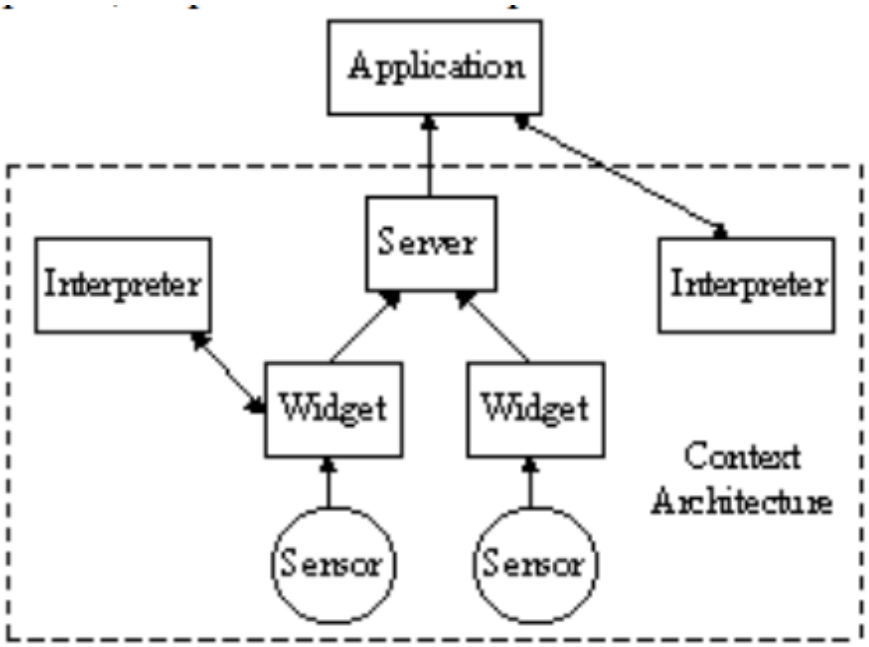


FIGURA 1- ARQUITECTURA DE CONTEXTTOOLKIT (6)

Modelo utilizado por la herramienta:

Se basa en grupos de duplas atributo-valor codificadas en XML. Es un modelo de información muy sencillo pero capaz incorporar nuevos componentes en tiempo de ejecución.

Evaluación de la herramienta:

Esta herramienta se centra casi exclusivamente en la captación del contexto, cuando este es solo uno de los problemas de este tipo de aplicaciones.

Si bien está disponible para descargar, no logramos hacerla funcionar.

Los foros están inactivos desde el año 2007, de todas formas intentamos comunicarnos con ellos pero falla su sistema de registro, por lo cual no tuvimos forma de hacerlo.

Esto nos llevó a descartar la posibilidad de utilizar esta herramienta, pero tomamos alguna idea a partir de la misma, como por ejemplo, nos interesó el uso de XML como lenguaje para definir el modelo del contexto.

2.6.2 The context managing framework

Context Managing Framework (8)se centra básicamente en la obtención, modelado y razonamiento del contexto.

Este Framework fue desarrollado como un experimento para la plataforma móvil Symbian (www.symbian.com), se centra más en las capacidades del dispositivo móvil que en la infraestructura.

Se utiliza el sistema de pizarra (blackboard) para la comunicación entre las entidades del Framework.

Su arquitectura está dispuesta en forma jerárquica teniendo un componente central al que la aplicación accede, se puede observar en la Figura 2. De esta forma se crean dos capas: una con el componente central y otra con los componentes secundarios que este componente utiliza.

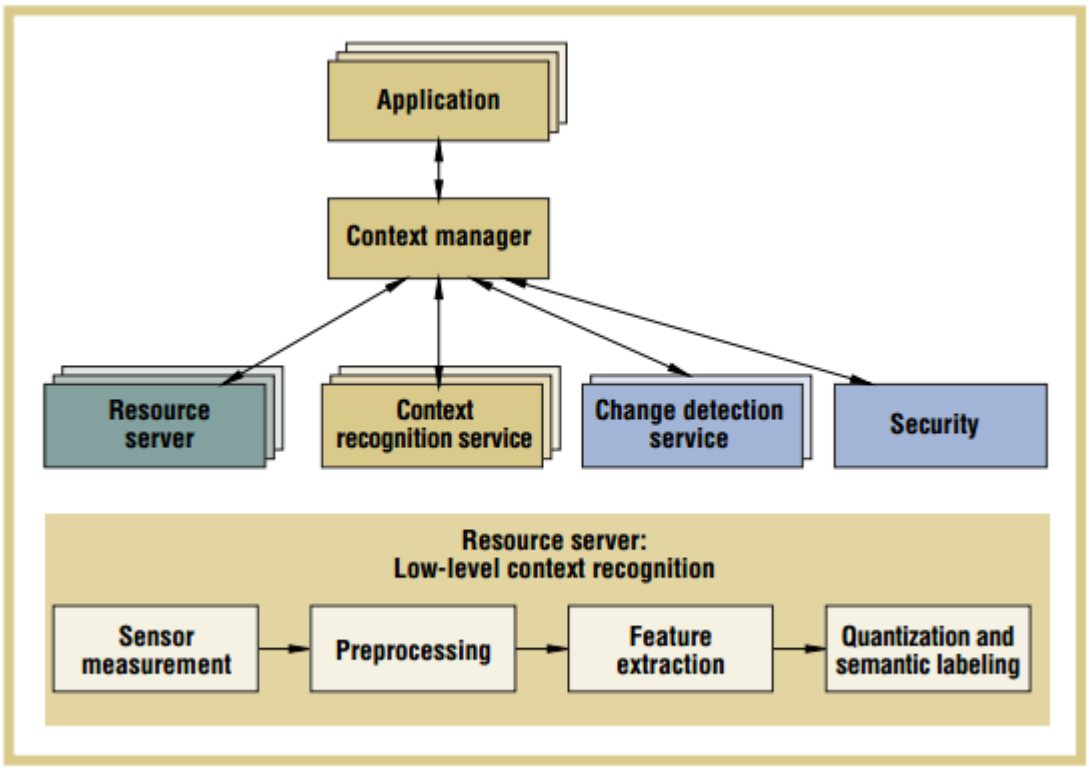


FIGURA 2- CONTEXT MANAGEMENT FRAMEWORK (8)

El Context manager (manejador de contextos) funciona como un servidor central mientras que las otras entidades (excepto la seguridad) actúan como clientes y utilizan los servicios que el context manager proporciona.

Los clientes (aplicaciones, servidor de recursos, servicio de reconocimiento) pueden agregar, suscríbase y solicitar información del contexto. Los contextos se almacenan en manejador de contexto proporcionando respuestas y notificando los cambios a los clientes. En la figura también se muestra el flujo del Servidor de recursos (Resource server) se ilustran las fases requeridas para transformar los datos del sensor en información de contexto interpretable por el Framework.
El context manager, algún servidor de recursos, y las aplicaciones se ejecutan en el dispositivo móvil en sí, y los servicios pueden ser distribuidos o locales. En el corazón del sistema de la terminal móvil está la pizarra (blackboard) que se encarga de manejar el contexto. Este nodo central almacena información del contexto de cualquier fuente disponible en la terminal, la cual se sirve a los clientes en tres formas:

- Los clientes pueden consultar directamente al manejador (Como una base de datos de contexto) para obtener datos del contexto.
- Los clientes pueden suscribirse a varios contextos y cambiar los servicios de notificación.
- Los clientes pueden utilizar contextos compuestos de forma transparente, donde el manejador de contexto se contacta con los servicios de reconocimientos necesarios.

Fuentes de información:

Un dispositivo móvil puede adquirir información del contexto de varias fuentes. Es este experimento se utilizan los sensores (para la temperatura, humedad y la pantalla táctil), micrófono, acelerómetros y canales de luz. Se diseñó el Framework para manejar fuentes tales como:

- Los procesos de los dispositivos móviles, que representan la información interna de los mismos (como aplicaciones que se están ejecutando).
- La red Internet y otros recursos.
- El GPS, entre las más importantes fuentes de información de la ubicación
- Procesos internos del dispositivo, que proporcionan información explícita de los usuarios y las aplicaciones del dispositivo por ejemplo “tareas programadas”, preferencias del usuario e información sobre las redes sociales.
- La información de tiempo, que se utiliza para asociar ciertos eventos con los demás y las secuencias de eventos se podría utilizar para predecir un evento a futuro.

Información de contextos basados en sensores:

Para estos se utiliza ontologías, para gestionar la información sistemáticamente. Las entidades del Framework deben tener una estructura común para la representación de información. Por lo tanto se diseñó una ontología para representar la información de los sensores, consiste en un esquema que representa la estructura y las propiedades de todos los conceptos de la ontología y un diccionario para el usuario, extensible donde se presenten los términos para describir la información del contexto. Para facilitar la comunicación, se utilizó RDF para describir la sintaxis. RDF ofrece propiedades y sintaxis comunes para describir información y permite compartir la ontología entre proveedores diferentes de información o compartir contexto entre dispositivos de comunicación de forma colaborativa.

Modelo utilizado por la herramienta:

Modela cada contexto utilizando seis propiedades:

- **ContextType**: Se refiere a la categoría de contexto. Todas las suscripciones y consultas deben tener un tipo de contexto o fuente como parámetro primario.
- **ContextValue**: Se refiere a la semántica o "valor" absoluto del tipo contexto y por lo general se utiliza junto con el ContextType, formando una descripción verbal. En algunos casos, el valor de contexto puede contener un valor numérico o características que describen el contexto.
- **Confidence**: Es una propiedad opcional, describe la incertidumbre del contexto, como una probabilidad de error en función de la fuente.
- **Source**: Describe la fuente del contexto (semántica). Un cliente interesado en contextos de una fuente específica puede utilizar esta propiedad.
- **Timestamp**: Indica la última vez que ocurrió el contexto.
- **Attributes**: Especifican la expresión “contexto libre”, puede contener cualquier detalle adicional no incluido en el otras propiedades.

Las propiedades contextType y ContextValue, son necesarias para poder modelar un contexto, el resto son opcionales.

Evaluación de la herramienta:

Si bien la herramienta cumple con la premisa de modelar el contexto como se planteó como objetivo este proyecto, lo acota al terreno de los dispositivos móviles, más precisamente a los dispositivos que cuentan con el sistema operativo Symbian. Este sistema operativo, fue implementado para los celulares Nokia y ya prácticamente no se utiliza en el mundo.

Por otra parte no está disponible para descargar.

Ambas razones llevaron a desechar la posibilidad de utilizar esta herramienta.

2.6.3 Aura

El proyecto Aura (9) es un Framework diseñado para la movilidad del usuario, cuando este se encuentra ejecutando aplicaciones en su dispositivo. Cuando un usuario se mueve a otro lugar, con diferentes recursos, Aura le asiste y se adecua al nuevo entorno. El objetivo es mover sus tareas de un entorno a otro sin que se vean afectadas. Esta herramienta pretende hacer uso de los recursos del entorno lo máximo posible sin que las tareas que el usuario realiza se vean afectadas en su rendimiento, avisándole si una correcta ejecución no es posible con los recursos disponibles.

La arquitectura consta de los siguientes componentes principales:

- Gestor de tareas, llamado Prism, que representa el mismo concepto de Aura Personal

- Observador del contexto, que monitorea el contexto, sus recursos y sus cambios de estado. Da su información al resto de elementos.
- Gestor del entorno, que actúa de enlace con el entorno.
- Proveedores, que representan los servicios básicos de los que se componen las tareas.
- Conectores, que gestionan las conexiones los diferentes componentes y sirven de abstracción del medio de conexión.

A continuación se muestra la arquitectura de Aura

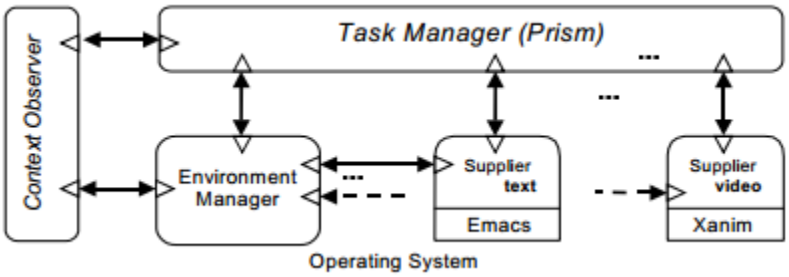


FIGURA 3- ARQUITECTURA DE AURA (9)

El Gestor de Tareas (Prism) representa al Aura Personal de usuario y es el elemento central de la arquitectura. Para Aura manejar una tarea puede implicar manejar varios programas a la vez. El gestor tiene en cuenta los siguientes sucesos:

- El usuario se traslada a otro ambiente: Prism coordina la migración de toda la información de la tarea al nuevo entorno y negocia la conexión con el nuevo gestor de entorno.

- El contexto cambia: si varían las condiciones relacionadas con la tarea en curso, Prism analiza los nuevos valores de Calidad del Servicio y si son incompatibles (bien porque los nuevos componentes no soportan ciertas características o porque simplemente componentes desaparecen) se negocia con el Gestor de Entorno una nueva configuración para dar soporte a la tarea.

- La propia tarea cambia: Cuando Prism recibe notificación de que se desea cambiar de tarea (el usuario lo desea, llega un evento en el calendario u ocurre algún cambio en el contexto que fuerza un cambio de tarea) gestiona el almacenamiento del estado de la tarea actual e instancia la nueva.

Los proveedores encapsulan todos los servicios que componen las tareas. Puede haber varios proveedores en el mismo entorno con diferentes capacidades y condiciones. Estos son capaces de representar el estado del servicio para obtener y dar información sobre él.
El Observador de Contexto es el elemento que abstrae la captación de toda la información contextual del entorno. Se encarga de la notificación de cambios de contexto tanto a Prism como al Gestor de Entorno. Podrá haber un Observador de Contexto en cada entrono diferente.

El Gestor de Entorno se abstrae de la comunicación entre entorno y aplicación. Sabe qué componentes hay disponibles en cada momento y qué servicios pueden ser cubiertos.
Los conectores gestionan todos los aspectos relacionados con la comunicación y abstraen de ello a los componentes. En el cambio de contexto o de entorno, es necesario una reconfiguración, añadir o quitar proveedores y quizá incluso conectarse con nuevos Observadores de Contexto y Gestores de Entorno.
Estas conexiones implican diferentes medios físicos de transmisión de datos, diferentes tecnologías de comunicación distribuida e invocación remota. Los conectores existen para ocultar todos esos detalles a los componentes. Existen cuatro tipos diferentes de conectores: los que conectan Prism a los proveedores, los que lo conectan al Gestor de Entorno, los que lo conectan al Observador de Contexto y los que conectan el Gestor de Entorno con el Observador de Contexto. Cada uno de estos tipos tiene su propio protocolo y puede ser implementado de varias formas diferentes dependiendo de la naturaleza y los medios de los componentes.

Modelo utilizado por la herramienta:

Utiliza un modelo basado en presencia, proximidad física, localización y comunicación espontánea. En esta aproximación, cada objeto individual liga un perfil descriptivo. Este modelo es almacenado en una base de datos relacional.

Evaluación de la herramienta:

Si bien el enfoque de este Framework es muy interesante, está enfocado totalmente a la movilidad del usuario, lo que deja afuera contextos que no están relacionados con la ubicación.

Una característica fundamental, es que guarda el modelo del contexto en una base de datos para usarla luego, pero por otra parte no ofrece ninguna facilidad para el razonamiento o interpretación del contexto.

No permite modelar genéricamente los contextos, siendo este uno de los objetivos planteados. Por este motivo se decidió no utilizar esta herramienta.

2.6.4 Gaia

El proyecto Gaia (10) es una infraestructura que pretende servir de apoyo a aplicaciones Context-Aware. El objetivo es diseñar un middleware para un sistema operativo que gestione los recursos disponibles en un espacio activo. Pretende ampliar el alcance de los sistemas tradicionales para abarcar el espacio físico y los dispositivos que lo rodean, convirtiendo estos espacios en sistemas interactivos denominados Active Spaces. Para esto Gaia pretende trasladar la funcionalidad de un sistema operativo a los Active Spaces, estos espacios están coordinados por un software que ayuda a la realización de las actividades del usuario.

El proyecto Gaia tiene una arquitectura específica para este tipo de sistema que podemos observar en la figura 4.

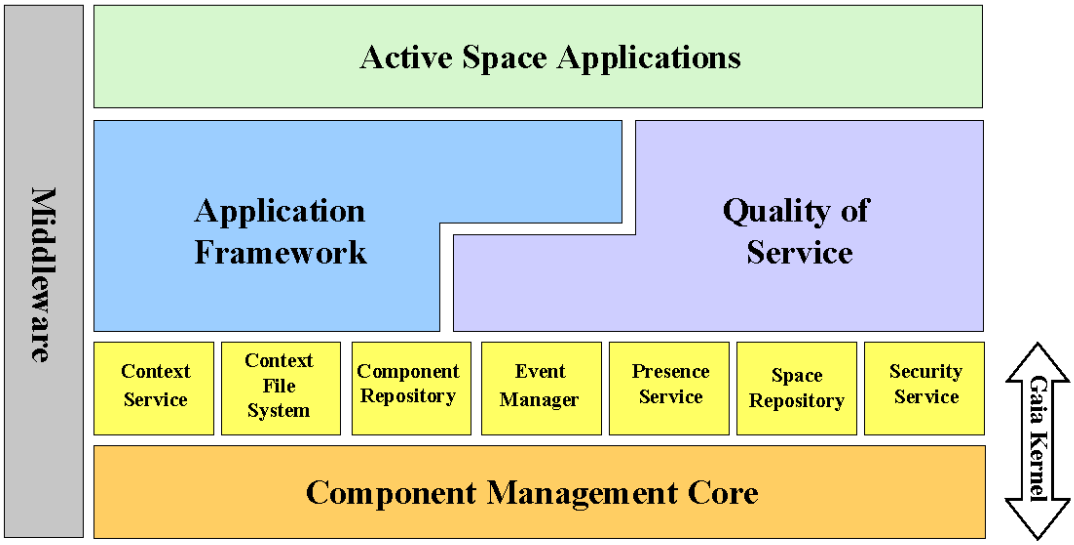


FIGURA 4 - GAIA ARQUITECTURA (10)

El núcleo de Gaia consta de los siguientes elementos, que se muestran gráficamente en la figura 4:

- Servicio de Contexto: El modelo define varias propiedades del contexto, por ejemplo las operaciones que se pueden realizar en contexto. Además proporciona información del propio contexto del Espacio Activo. Actúa de punto de contacto entre el contexto y las aplicaciones del sistema, posibilitando a las mismas el poder informarse, reaccionar y adaptarse a él.
- Sistema de Archivos de Contexto (CFS): Es un sistema de archivos sensible al contexto para espacios activos. CFS mejora la funcionalidad tradicional de un sistema de archivos para el entorno de Active Space mediante el suministro y transformación de datos. Los Active Space pueden estar ocupados por muchas personas y el contexto del espacio pueden estar en constante cambio. Un espacio puede ser utilizado para una conferencia y más tarde a una reunión. CFS utiliza esta información de contexto para organizar la información de modo que sea más fácil de encontrar material adecuado para aplicaciones lanzadas de forma automática o de larga duración.
- Repositorio: Implementa un servicio de repositorio para poder almacenar todos los componentes de software que se conocen en el Active Space.
- Gestor de Eventos: Proporciona un modelo para la comunicación desacoplada entre diferentes entidades en un Active Space.
- Servicio de Presencia: Los Active Spaces están compuestos por muchas entidades, incluidas las personas, componentes de software y dispositivos de hardware. Algunas de estas entidades son de carácter permanente y disponible para su uso, siempre y cuando el sistema está en funcionamiento, mientras que otros pueden ser transitorios. Se requiere un control para reconocer cuando las entidades están activas y si están presentes en un espacio físico.
- Repositorio de espacio: es una base de datos centralizada que contiene información sobre todos los dispositivos y servicios activos en un Active Space.

- Servicio de seguridad: El dinamismo y el alcance que tienen los Active Space requieren mecanismos eficaces para asegurar la infraestructura utilizada.

Modelo utilizado por la herramienta:

El modelo de del contexto se hace en forma de predicado de cuatro elementos:
Context (<ContextType>, <Subject>, <Relater>, <Object>)

- El tipo de contexto (ContextType) se refiere al contexto que el predicado está describiendo
- El sujeto (Subject) es el objeto, persona o lugar a quien se refiere
- El objeto (Object) es el valor que adquiere ese contexto
- La relación (Relater) relaciona el sujeto con el objeto comúnmente mediante operadores relacionales (>= <), un verbo o una preposición.

Ejemplo de predicado

Context (temperatura, sala, is, 20ºC)

Los mismos predicados que sirven para constatar hecho, sirven también para enunciar reglas, que pueden combinarse utilizando operaciones lógicas como cuantificadores, implicación, conjunción o disyunción:

Context (cantidad de personas, sala, >,4) AND Context(app, Power point, is, running)=>Context(Actividad Social, sala, is, presentacion)

De esta forma, se crea un conjunto de reglas y una base de conocimiento que sirve para inferir contexto de más alto nivel a partir del contexto de bajo nivel recogido. Esto implica que el Gestor de Contexto lleva embebido un motor de reglas.

Evaluación de la herramienta:

Gaia no está disponible para su descarga, por lo tanto no pudimos utilizarla.

De todas formas, al investigar sobre la misma encontramos la información mencionada sobre el modelo utilizado, la cual nos sirvió para ser tenida en cuenta en el momento de diseñar y definir el modelo encontrado.

2.6.5 Socam

Socam (11)es un middleware para aplicaciones Context-Aware, dirigido especialmente al prototipado rápido general. Tiene varios componentes dispuestos en una forma distribuida en lugar de jerarquizada o en capas.
El prototipo de esta herramienta está en construcción, lo que se quiere hacer es crear una infraestructura adecuada para sistemas sensibles al contexto, para ello esta debe proporcionar soporte a la mayoría de la tareas que intervienen, como la adquisición del contexto de distintas fuentes, tales como sensores,

bases de datos. Realizar la interpretación del contexto, encargarse de la difusión del contexto a las partes interesadas en forma distribuida y en el momento oportuno.

En la figura 5 podemos observar su arquitectura

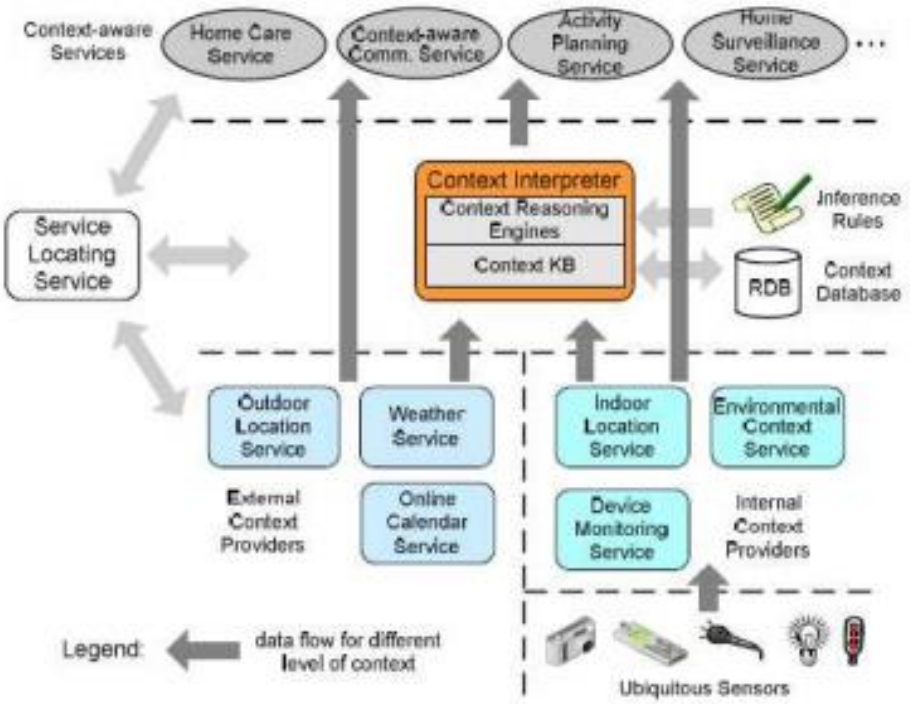


FIGURA 5 - ARQUITECTURA SOCAM (11)

Los elementos que la componen se describen a continuación:

- **Proveedores de Contexto (Context Providers):**

Gestionan la captación de contexto a través de sensores y crean el contexto de bajo nivel. La información de contexto es representada con una ontología utilizando OWL.

- **Interpretador de Contexto (Context Interpreter):**

Utiliza el contexto de bajo nivel para inferir contexto de alto nivel. También actúa de proveedor ofreciendo nueva información de contexto. Está compuesto de un razonador de contexto y una base de conocimiento. El razonador se encarga de la inferencia del nuevo contexto y el mantenimiento del buen estado de la base de conocimiento. Está diseñado para que nuevos razonadores puedan ser añadidos al mismo interpretador. La base de conocimiento es el conjunto de ontologías de contexto almacenadas en Base de Datos, además de un grupo de APIs para añadir y manipular nuevas.

- **Servicio de Localización de Servicios (Service Locating Service):**

Este servicio actúa de índice y directorio de los Proveedores de Contexto existentes en el sistema. Cada uno de ellos se registra en él y da una descripción en OWL del tipo de contexto que ofrece.

- **Servicios Context-Aware:**

Son los servicios o aplicaciones Context-Aware que usan la arquitectura, buscan proveedores de contexto y los utilizan. Pueden asimismo añadir nuevas reglas de contexto de alto nivel al razonador para obtener la información que necesitan.

Modelo utilizado por la herramienta:

El contexto es representado en un modelo basado en ontologías utilizando OWL - Web Ontology Language, que proporciona un vocabulario para representar y compartir el conocimiento contexto en un dominio de la computación ubicua, incluyendo definiciones de máquina-interpretables de conceptos básicos en el dominio y las relaciones entre ellos. Al estar basado en ontología el modelo de información de contexto permite describir los contextos semánticamente de una manera que es independiente del idioma de programación, del sistema operativo o middleware; permite un análisis formal de los conocimientos de dominio, es decir, el razonamiento contexto.

Las estructuras y propiedades de los diferentes contextos se describen en una ontología que puede incluir descripciones de clases, propiedades y sus instancias. La ontología está escrita en OWL como un conjunto de tripletas RDF, cada declaración esta de la forma (SUBNAME, predicado, objName), donde SUBNAME y objName son objetos de la ontología o individuos, y el predicado es una relación de propiedad definida por la ontología.

Evaluación de la herramienta:

Si bien la documentación encontrada sobre Socam dice que modela los contextos de forma genérica, no está disponible para su uso y la información encontrada sobre la misma es apenas una introducción al trabajo que se quiere realizar. Por lo tanto no nos fue posible comprobarlo ni utilizarla.

2.6.6 Cobra

Cobra (Contexto Broker Architecture) (12) es una arquitectura de agente intermediario para apoyar los sistemas sensibles al contexto en espacios inteligentes (por ejemplo, salas de reuniones inteligentes, casas inteligentes y los vehículos inteligentes). El elemento central de la arquitectura es la presencia de un agente inteligente llamado el agente de contexto.
El agente de contexto es un servidor especializado que se ejecuta en una ordenador fijo con buenos recursos (procesador y memoria potente). En un espacio inteligente, el papel del agente de contexto es la de mantener un modelo compartido de contexto en el nombre de una comunidad de agentes y dispositivos en el espacio y proteger la privacidad de los usuarios mediante la aplicación de las políticas definidas por el usuario al compartir información con los agentes del espacio asociado.
En la Figura 7 se muestra el diseño de alto nivel del agente y su relación con los otros agentes en un espacio inteligente. Todas las entidades de computación en un espacio inteligente, se presume que tienen conocimiento a priori acerca de la presencia de un agente de contexto.

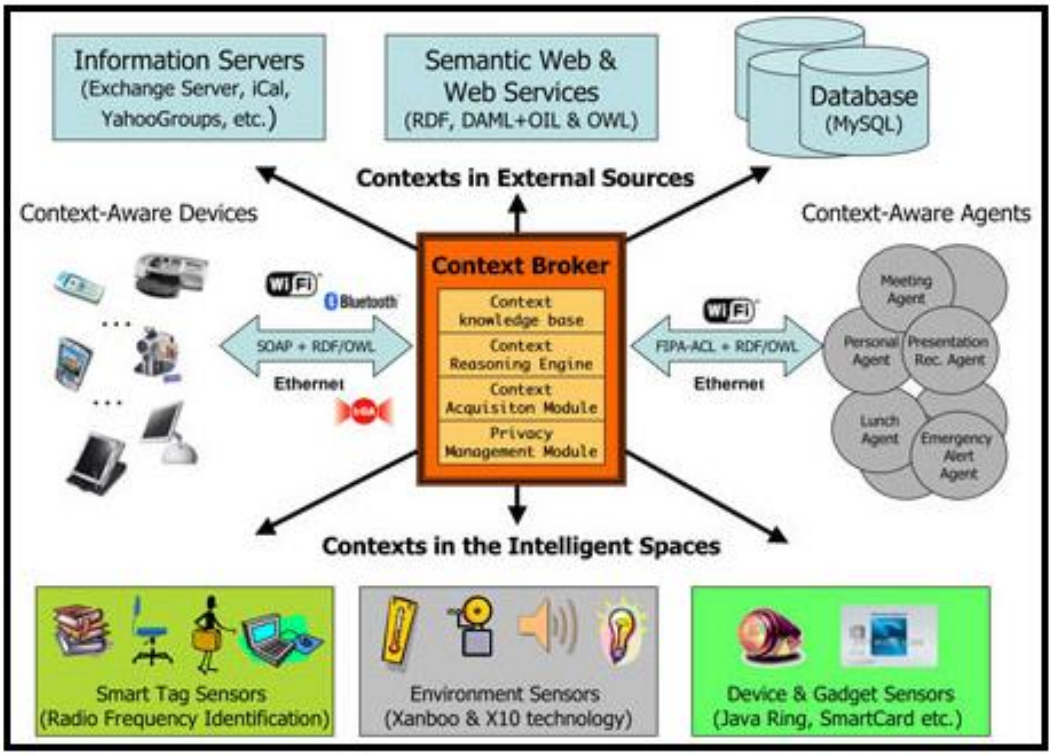


FIGURA 6 - COBRA

El diseño del agente de contexto está compuesto por cuatro componentes:

- Context Knowledge Base: Almacenamiento persistente del conocimiento del contexto.
- Context Reasoning Engine: Maquina de inferencia (es quien controla qué reglas disparan) que razona sobre el conocimiento contexto almacenado.
- Context Acquisition Module: Es una librería de procedimientos que forman un middleware para la adquisición de contextos.
- Policy Management Module: Conjunto de reglas de donde se deducen las instrucciones para decidir los permisos adecuados para diferentes entidades de computación para compartir una pieza particular de información del contexto y para la selección de los destinatarios que recibirán las notificaciones cuando el contexto cambie.

Modelo utilizado por la herramienta:

En CoBrA se definen un conjunto de ontologías COBRA-ONT para el modelado el contexto en espacios inteligentes. Para COBRA-ONT se utilizó el Lenguaje de Ontologías Web OWL (13)que define los conceptos típicos asociados con lugares, agentes y eventos.

Evaluación de la herramienta:

Leyendo la documentación menciona que permite exportar los modelos, pero no se puedo realizar ninguna prueba ya que siguiendo los pasos para ejecutar la herramienta da error. No pudimos resolver el problema, además de que no tiene movimientos desde el año 2004 en su sitio, por lo que se descartó la utilización de esta herramienta.

Además, como desventaja se puede mencionar que solo usuarios con conocimientos de desarrollo de software con el lenguaje Java pueden utilizarlo.

2.6.7 Tabla Comparativa de las herramientas

A continuación se muestra una tabla comparativa de las herramientas, con las características antes mencionadas:

Nombre	Descripción	Disponibilidad	Características
Context Toolkit	Es una plataforma que tiene como objetivo simplificar el proceso de construcción de aplicaciones context-aware	Herramienta gratuita, de código abierto. Desarrollada con java. Se puede descargar de http://contexttoolkit.sourceforge.net/documentation/InstallationGuide.html (Última Visita: marzo - 2016)	Cuenta con documentación pero desactualizada. Infraestructura distribuida. Grupos de duplas clave-valor para el modelado del contexto. Historial de datos.
The context managing framework	Este Framework ideado por Korpiää se centra básicamente en la captación, modelado y razonamiento del contexto.	No se encuentra disponible para descargar	Cuenta con muy poca documentación. Desarrollado para celulares con sistema operativo Symbian. Modelo de capas y Etiquetado semántico. Utiliza RDF para el modelado
Aura	Es un Framework que pretende hacer uso de los recursos del entorno al máximo posible sin que las tareas que el usuario realiza se vean afectadas en su rendimiento.	Herramienta gratuita desarrollada con java https://github.com/forcedotcom/aura http://documentation.auraframework.org/auradocs (Última Visita: marzo - 2016)	Cuenta con buena documentación. Diseñado para la movilidad del usuario. Abstracción del medio de transmisión de datos y mecanismos de comunicación Base de datos relacional para el modelado del contexto. Sistema de Archivos distribuido para almacenar la información. Historial de datos
Gaia	Es una infraestructura (además de un framework) que pretende servir de apoyo a aplicaciones Context-Aware.	No esta disponible para descargar.	Cuenta con buena documentación. Infraestructura es descentralizada y desacoplada (muy tolerante a fallos). Utiliza modelos distribuidos de componentes. Sistema de Archivos distribuido para almacenar la información. Historial de datos.
Socam	Provee de un middleware para aplicaciones Context-Aware dirigido especialmente al prototipado rápido general.	No está disponible para descargar	Cuenta con muy poca documentación. Componentes distribuidos. Utiliza OWL para representar la información del contexto Almacenamiento en base de datos relacional. Historial de datos.
Cobra	Es un Framework que pretende dar soporte a aplicaciones Context-Aware, los autores lo llaman Espacios Inteligentes.	Herramienta gratuita, de código abierto. Desarrollada con java. Se puede descargar de http://cobra.umbc.edu/ (Última Visita: Setiembre - 2016) Pero no funciona.	Documentación desactualizada. Arquitectura centralizada. Utiliza OWL para representar la información del contexto. Seguridad, políticas de acceso. Almacenamiento en base de datos relacional. Historial de datos.

Tabla 1- comparativa de herramientas para aplicaciones sensibles al contexto

Si bien, como se indica en la tabla comparativa de herramientas para aplicaciones sensibles al contexto, algunas de las herramientas cuentan con buena documentación, no se encontró una que cumpla con la capacidad de modelar y a su vez interpretar los contextos de forma genérica. La que más se aproximó a la necesidad del proyecto fue “the context managing Framework”, pero esta no se puede descargar, ya que no tienen ningún sitio público, simplemente tienen el paper desde el cual se obtuvo la información. Se intentó contactarse con los autores, pero no se encontró forma. Por ende tampoco se puede utilizar.

Por esta razón se decidió implementar una herramienta que cumpla con las necesidades de obtener, modelar e interpretar los contextos. La especificación de la herramienta desarrollada se realiza a partir del capítulo 4.

3. Modelo de Contextos

Como se mencionó previamente, por **"MODELANDO LOS CONTEXTOS"**, se puede decir que se busca una forma de modelar, o encontrar un modelo para todos los contextos, es decir que se busca encontrar la forma genérica de representar los contextos. En este caso se buscó desarrollar un modelo para los contextos cuyos elementos se pueden evaluar cuantitativamente.

3.1 Motivación para definir un modelo de contexto

Dada la heterogeneidad de contextos existentes, es muy difícil comprenderlos a todos sin entrar a analizarlos individualmente. Esto trae varios problemas a la computación ubicua, ya que para desarrollar sistemas sensibles al contexto, se requiere interactuar con el contexto para el cual se aplica de forma individual.

Es por esto, que la capacidad para comprender el entorno, y el proceso para adaptarse al contexto y por lo tanto contar con un modelo que represente a todos los contextos es la solución a muchos problemas como se mencionó en las motivaciones, y esto es parte fundamental de este proyecto.

En un análisis inicial, hay cinco aspectos importantes a considerar, conocido como la teoría de las cinco W's. Es un concepto utilizada en múltiples áreas, que consiste en contestar 5 preguntas básicas: ¿qué? (WHAT), ¿por qué? (WHY), ¿cuándo? (WHEN), ¿dónde? (WHERE) y ¿quién? (WHO). Esta regla creada por Lasswell (1979) puede considerarse como una lista de verificación mediante la cual es posible generar estrategias para implementar una mejora, en nuestro caso identificamos:

- When: El momento en el que se encuentra.
- Where: Tener idea de la locación es importante, para poder deducir distancias o aspectos importantes para el usuario dada su ubicación.
- What: Se refiere a lo que el usuario está haciendo o quiere hacer.
- Who: Perfil del usuario.
- Why: Objetivo del usuario.

3.2 Construcción de un modelo

La construcción del modelo se realizó siguiendo las siguientes fases:

1 Identificación de un problema o situación compleja que necesita ser simulada, optimizada o controlada y por tanto requiere un modelo.
2 Elección del tipo de modelo, esto requiere precisar qué tipo de respuesta pretende obtenerse, cuales son los datos de entrada o factores relevantes, y para qué pretende usarse el modelo.
3 Formalización del modelo en la que se detallan qué forma tienen los datos de entrada y qué tipo de herramienta se usará.
4 Comparación de resultados: los resultados obtenidos como predicciones necesitan ser comparados con los hechos observados para ver si el modelo está prediciendo bien. Si los resultados no se ajustan bien se vuelve a la fase 2.

3.3 Modelo Utilizado

Con el fin de encontrar un modelo, se plantearon diversos escenarios donde se comprobó que para definir los contextos interesan los elementos (o fuentes de entrada) y la relación entre los mismos.

De esta manera se detectó que la representación debe tener 2 capas fundamentales:

1- Definición de elementos (o fuentes de entrada) y su tipo.
2- Relación entre los mismos, la cual se puede evaluar cuantitativamente. En el modelo se añade una función objetivo impulsado por un límite en dichas relaciones, la cual ayuda a entender más el contexto.

La semántica utilizada para desarrollar el modelo, es la de modelo orientado a objetos. Un modelo de datos orientado a objeto que se diseñó utilizando un diagrama de clases UML. El diagrama de clases UML utilizado es el siguiente:

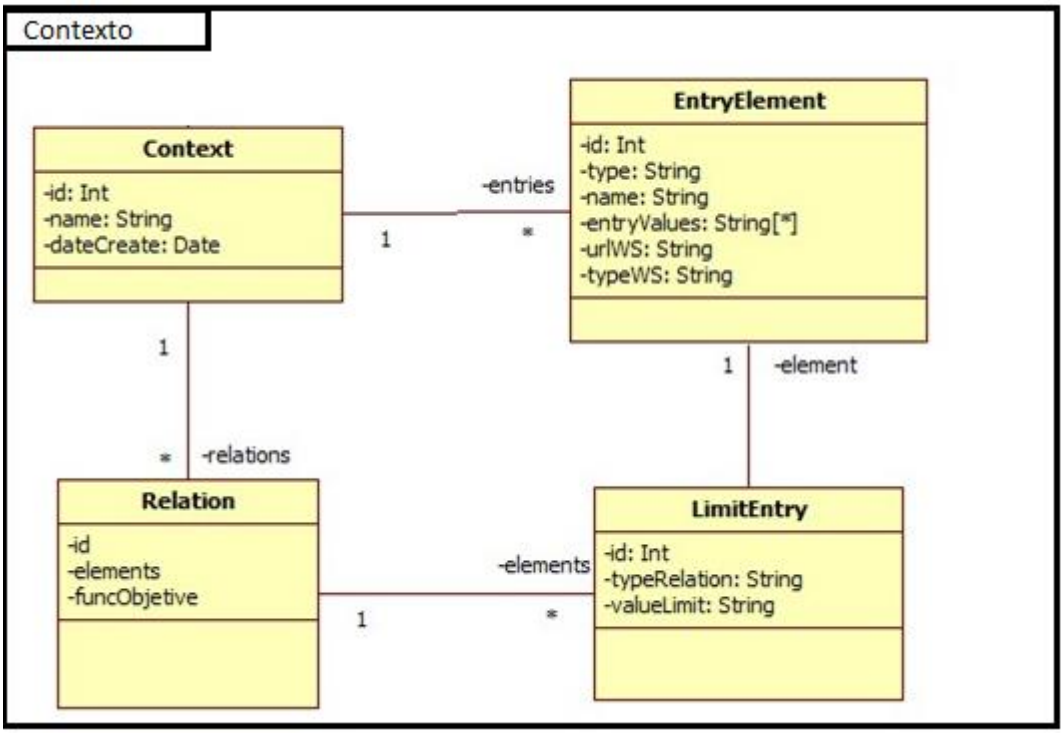


FIGURA 7 - DIAGRAMA DE CLASES (UML) MODELO DE CONTEXTO

3.4 Lenguaje utilizado para representar el contexto

Para definir el modelo diseñado utilizando UML que se mostró en la sección anterior, se utilizó el lenguaje XML (14).

XML:

XML (Extensible Markup Language) es un lenguaje de etiquetas, es decir, cada paquete de información está delimitado por dos etiquetas.

Proporciona información sobre datos estructurados.

Es un metalenguaje que separa la presentación del contenido.

Ventajas de utilizar XML:

Un programa informático puede estar escrito en Java, Visual Basic, PHP, .Net y cualquier otro lenguaje. En esencia, todos los programas procesan información, entendiéndose por información “dato + significado”. Por lo tanto un documento escrito en XML tendría la información que necesitan los programas para procesar.

XML se plantea como un lenguaje estándar para el intercambio de información entre diferentes programas de una manera segura, fiable y libre, ya que no pertenece a ninguna compañía.

Al ser una de las premisas colaborar con la computación ubicua, se considera que definiendo el modelo utilizando XML, se puede utilizar por distintas aplicaciones, sin importar el lenguaje en que lo escriban.

Otra ventaja importante, es la existencia de múltiples librerías, y herramientas existentes que son gratuitas, para el manejo del contenido de los documentos XML, lo que sin dudas reduce los tiempos de desarrollo para cualquier integración que quiera utilizarlo.

Modelo definido utilizando XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
      <contextname="Nombre del contexto"date="fecha de creación">
      //Se pueden definir todas las entradas que se desee
            <entrys>                                          1- Elementos o fuentes
                  <entryid="">                                   de entrada
                        <name></name>
                        <type></type>
                        <typeWS></typeWS>
                        <urlWS/>
                  </entry>
                  <entryid="">
                  ........
                  </entry>
            </entrys>
            <relations>                                       2- Relaciones
                  <relation>
                        <funcObj>Nombre de la función objetivo</funcObj>
                        <limitEntrys>
                        <limitEntry>
                        <entryId>id de fuente de entrada</entryId>
                        <typeRelation>Enumerado{Mayor,Menor,Igual,
                        Distinto}</typeRelation>
                        <valueLimit>Numero</valueLimit>
                        </limitEntry>
                        <limitEntry>
                        ……
                        </limitEntry>
                        </limitEntrys>
                  </relation>
            </relations>
      </context>
</contexts>
```

En el siguiente capítulo se muestra la creación de un Framework, el cual utiliza este modelo, con lo cual queda demostrada su utilidad.

4. Context Modeling Framework

En este capítulo se describe la herramienta desarrollada, sus componentes y su estructura.

4.1 Motivación

La necesidad de contar con una herramienta que permita obtener información de los contextos, poder modelarla o representarla de manera genérica e interpretarla para poder ser utilizada en distintos sistemas, llevo al desarrollo del "Context Modeling Framework".

Esta herramienta consiste en un Framework para la creación, configuración, modelado y monitoreo de contextos.

Con este Framework, usuarios desarrolladores (o no desarrolladores) pueden de manera simple modelar y utilizar los elementos del contexto y sus relaciones. Además provee pantallas de configuración, de tal manera que no es necesario ser desarrollador para poder utilizarlo, es decir que no es necesario contar con conocimientos de desarrollo de software para utilizarlo. De este modo no se limita su uso a solo un grupo restringido de usuarios.

Una motivación importante, dado lo ya mencionado, es que con esta herramienta se puede colaborar con el desarrollo de aplicaciones sensibles al contexto, que tienen como objetivo ayudar, facilitar y mejorar la calidad de vida de las personas.

A continuación se describe la solución, analizando la arquitectura, elementos, funcionamiento, visión a futuro y algunos ejemplos de funcionamiento.

4.2 Objetivos

Los objetivos del Context Modeling Framework son modelar, interpretar y monitorear contextos, realzando acciones en caso de que el contexto lo requiera.

Para esto, la herramienta deberá ofrecer funcionalidades para captar las fuentes de entrada como sensores, velocímetros, etc.

También deberá comprender estas entradas, es decir evaluarlas, y por último realizar acciones en caso de que así se defina.

Cumpliendo con los objetivos mencionados, la herramienta puede contribuir a solucionar o mitigar distintos problemas de la vida cotidiana, como por ejemplo:

- Contaminación del medio ambiente: Aplicación que avise a la empresa recolectora cuando un contenedor llega a cierto límite evitando así los contenedores desbordados.
    - En este caso los elementos identificados son:
        - Fuentes de entrada: Sensor de movimiento que detecta que se llegó al límite del contenedor.
        - Limite se ve determinado por el volumen.
        - Función Objetivo: Al alcanzar el límite, se le avisa a la empresa recolectora, para evitar que se desborde el contenedor.
- Transito: Rutas inteligentes, capaces de avisar a conductores por qué carril deben conducir de acuerdo a la velocidad que llevan, avisarles si más adelante hay algún desperfecto en la ruta o si hay algún accidente, etc.
    - En este caso algunos de los elementos identificados son:
        - Sensores de velocidad y de movimiento que indican por cual carril se mueve el vehículo.
        - Límites: Velocidades máxima y mínima para los carriles.
        - Funciones objetivos: Indicar a los conductores que deben cambiar de carril, o simplemente informar en caso de accidentes en su camino.
- Calidad de vida de los hogares: Si bien ya existen aplicaciones que hacen las casas más inteligentes, con este Framework sería sencillo crear climatizadores en caso de que estén personas en las salas, iluminación de acuerdo a lo que necesiten, etc.
    - Es sencillo crear un climatizador con el Framework ya que simplemente se deben indicar los siguientes elementos:
        - Fuente de entrada: Sensores que indiquen la cantidad de personas de una sala y sensores de temperatura.
        - Límites: Los mínimos de personas y de temperatura para los que se desea mantener la temperatura.
        - Funciones Objetivo: En caso de que se cumpla el límite inferior de temperatura con la cantidad de personas indicada, la función sería “Subir temperatura”, en el caso de que la temperatura pase la máxima deseada, la función es “bajar temperatura”.

- Vehículos que se manejen solos (Piloto Automático): Con el Framework se podría implementar un piloto automático para un avión que mantenga la estabilidad y también que siga un rumbo determinado.
    - En este caso los elementos son (por simplicidad se nombran solo algunos):
        - Fuentes de entrada: GPS que determina la ubicación y el movimientos, sensores para detectar inclinación de las alas.
        - Límites: por ejemplo si se desea que el ala o baje a más de 45°, este sería el límite. En el caso del GPS, que no se desvíe más de 20 metros.
        - Función Objetivo:

        - Si el ala se inclinó a menos de 45°, la función objetivo es “levantar ala”

        - Si el avión se desvió del rumbo más de 20 metros a la derecha, la función objetivo es “doblar a la izquierda”.

4.3 Estructura del Context Modeling Framework

La estructura utilizada en el desarrollo del Framework se basa en tres grandes módulos relacionados entre sí: Fuentes de Datos, Intérprete y el entorno.

El sistema obtiene la información de las fuentes de datos (pertenecientes al entorno), la interpreta, y se lo comunica al entorno cuando se cumple un determinado objetivo.

A continuación, en la figura 6 se puede observar la relación entre los módulos antes mencionados.

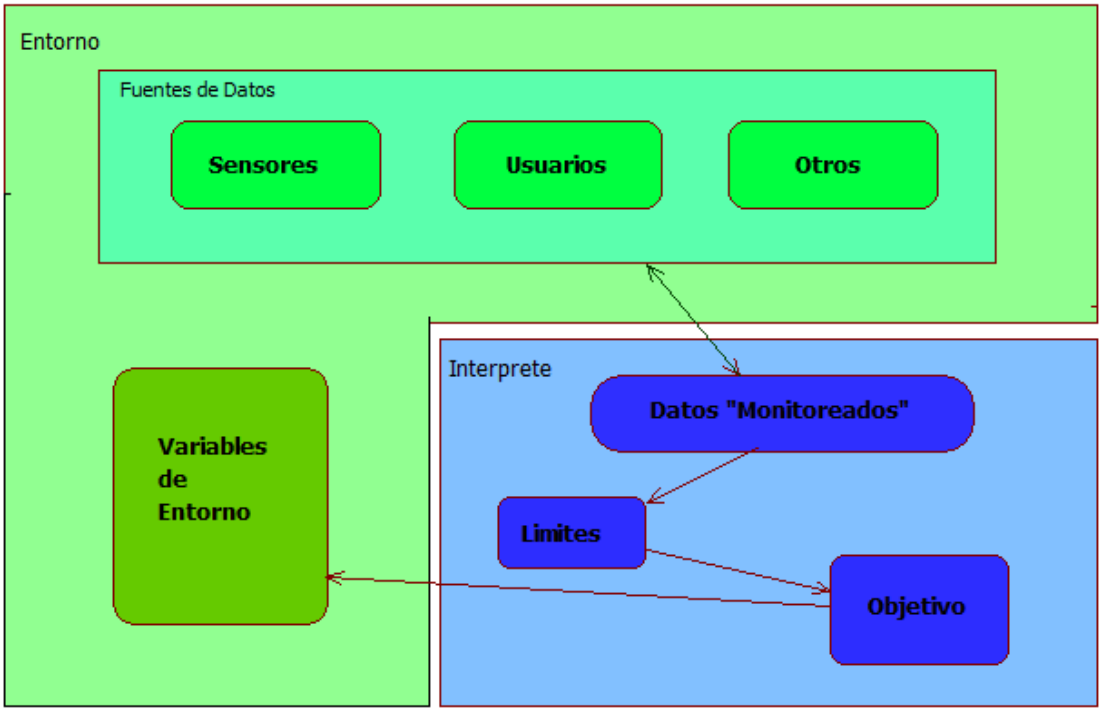


FIGURA 8 - ESTRUCTURA DE LA SOLUCIÓN

Por otra parte, el desarrollo del Framework está basado en MVC (Modelos Vistas Controladores) para facilitar su lectura, y además está estructurado de la siguiente forma:

- Carpeta System:

Contiene el config_ctx.xml que es donde se guarda la configuración de los contextos, elementos y relaciones.

Carpeta Intérprete, contiene un MVC:

- Models: Tiene todas las clases encargadas de la persistencia, tanto JPA como a XML.
- Controllers: En esta se incluyen los controladores y clases encargadas del manejar los contextos.
- Views: Contiene los manejadores de las vistas correspondientes al wizard de configuración y dashbord de monitoreo. Las vistas propiamente dichas se incluyen en el Web-Content (por utilizar Prime Faces)

Web-Content, contiene las vistas:

- Wizard
- Dashboard
- Monitor

4.3.1 Fuentes de Datos

Las fuentes de datos son parte del entorno, son las que brindan la información de la cual se extrae la que es relevante para el contexto en cuestión.

Por ejemplo, de los sensores pueden tomar temperatura, movimiento, velocidad, luminosidad, cantidad, etc. Estos datos pueden ingresarse manualmente a través de una interfaz para el usuario, o un servicio web el cual se publica para esto al momento de crear un contexto. Eventualmente en un futuro se podrían ingresar datos desde fuentes externas, como por ejemplo desde un LMS del tipo "Moodle" donde se obtienen estadísticas. Dentro de estos se podrían definir límites dentro de las estadísticas, disparando (funciones objetivo) alertas o notificaciones para los administradores y los usuarios (ej. docentes y estudiantes) para que puedan tomar decisiones.

4.3.2 Interprete

Se encarga de tomar los datos (elementos, relaciones, límites y valores), y los interpreta, es decir que los toma y los evalúa para detectar cuando se alcanza un límite, y cuando esto sucede se encarga de mandar eventos al entorno según el objetivo deseado por el usuario. Esto se debe definir en la función objetivo indicado al momento de crear el contexto mediante el wizard o mediante el XML. Esta función objetivo interactúa con el entorno, modificando lo que se indique en la misma (subir o bajar temperatura, encender o apagar una luz, etc.)

Para evaluar si se alcanza un límite, se utilizan los operadores relacionales:

- Mayor, Menor, Igual, Mayor o Igual, Menor o Igual.

Y también se consideran los operadores lógicos:

- And y OR.

En el capítulo 3 se mencionó que se utiliza XML para representar el modelo, la estructura de este XML es la siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
	<contextname="Nombre del contexto"date="fecha de creación">
	//Se pueden definir todas las entradas que se desee
		<entrys>
			<entryid="">
				<name></name>
				<type></type>
				<typeWS></typeWS>
				<urlWS/>
			</entry>
			<entryid="">
			........
			</entry>
		</entrys>
		<relations>
			<relation>
				<funcObj>Nombre de la función objetivo</funcObj>
				<limitEntrys>
				<limitEntry>
				<entryId>id de fuente de entrada</entryId>
				<typeRelation>Enumerado{Mayor,Menor,Igual,
				Distinto}</typeRelation>
				<valueLimit>Numero</valueLimit>
				</limitEntry>
				<limitEntry>
				……
				</limitEntry>
				</limitEntrys>
			</relation>
		</relations>
	</context>
</contexts>
```

Se pueden definir varios contextos, dentro de cada uno se pueden definir varias fuentes de entrada como también distintas funciones objetivos que sean determinadas por las distintas fuentes entradas ingresadas. Las funciones objetivos pueden tener varios límites, por ejemplo Cantidad de Personas >2 y temperatura >20

Ejemplo Sala Climatizada:

A continuación se detalla el XML de un contexto de una sala climatizada, cuya función objetivo es bajar la temperatura si la cantidad de personas que se encuentran en la sala es mayor a dos y la temperatura de la sala supera los 20°

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
	<contextname="Sala"date="20/06/2015 22:21">
		<entrys>
			<entryid="2">
				<name>Cantidad de Personas</name>
				<type>Entero</type>
				<typeWS>consume</typeWS>
				<urlWS/>
			</entry>
			<entryid="3">
				<name>Temperatura</name>
				<type>Entero</type>
				<typeWS>consume</typeWS>
				<urlWS/>
			</entry>
		</entrys>
		<relations>
			<relation>
				<funcObj>BajarTemperatura</funcObj>
				<limitEntrys>
				<limitEntry>
				<entryId>2</entryId>
				<typeRelation>Mayor</typeRelation>
				<valueLimit>2</valueLimit>
				</limitEntry>
				<limitEntry>
				<entryId>3</entryId>
				<typeRelation>Igual</typeRelation>
				<valueLimit>20</valueLimit>
				</limitEntry>
				</limitEntrys>
			</relation>
		</relations>
	</context>
</contexts>
```

4.3.3 Variable de entorno

Las variables de entorno son las que se ven afectadas por las funciones objetivos definidas al momento de ser ejecutadas, es decir cuando se alcanza un límite.

Por ejemplo en el caso de una sala climatizada, puede ser la temperatura, y se puede modificar mediante una llamado a la función “modificar temperatura” (bajar o subir) del aire acondicionado.

5. Implementación del Context Modeling Framework

En el presente capítulo se especifican los aspectos técnicos adoptados para la implementación del Context Modeling Framework.

5.1 Arquitectura propuesta

La Figura 8 ilustra la arquitectura del sistema, identificando los principales componentes de la misma:

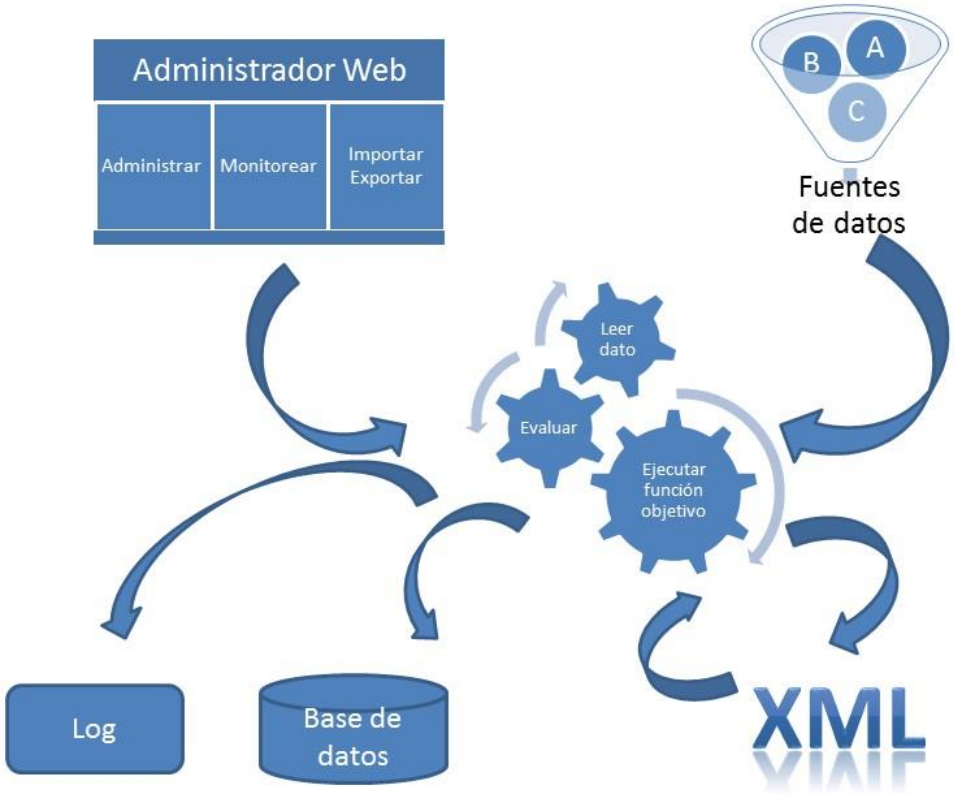


FIGURA 9- ARQUITECTURA DEL CONTEXT MODELING FRAMEWORK

La figura 9 indica los principales componentes:

Administrador Web:

- Contiene pantallas que permiten crear, editar, borrar, monitorear los contextos, así como también importar y exportar los XML con los contextos modelados, y también permite evaluar el Log con la información de todas las actividades ocurridas utilizando el Framework.

Fuentes de datos:

- Son las fuentes de datos que brindaran la información del contexto. Por ejemplo: sensores de temperatura, velocímetros, etc.
- Al momento de crear el contexto, el Framework proporciona dos opciones:
    - En una de ellas, se debe indicar si brindara un servicio web para obtener el valor de la fuente de entrada,
    - La otra opción es indicar si va a consumir un servicio para enviar el valor.

  Esta información queda asociada a cada Fuente de dato, y de esta manera el Framework sabe cómo obtener dicha información.

XML:

- Son los archivos que se pueden importar o exportar dentro del Framework. La estructura de este XML será la mencionada en la sección 3.4.

Log:

- El Context Modeling Framework, cuenta con la capacidad de registrar las actividades. El mismo genera un log indicando día y hora en que se realizó cualquier actividad dentro de un contexto, ya sea por parte del usuario, como por las fuentes de entradas mismas (aumento de temperatura, etc.).

Base de datos:

- Para persistir la información de los contextos, se utiliza la herramienta JPA. Esto permite guardar la configuración de los contextos en una base de datos MySQL. Esto es un aporte para los usuarios desarrolladores, que podrán extender el Framework, y pueden tomar desde esta base de datos los elementos del contexto y manipularlos según su necesidad.

Relación entre los elementos:

- El Context Modeling Framework, trabaja de la siguiente forma:
    1. Se crea uno o más contextos, ya sea utilizando el Wizard que provee el administrador web, o cargando un XML. Este contexto debe tener los elementos definidos, con sus relaciones y funciones objetivo.
    2. Toma valores de los elementos desde las fuentes de datos.
    3. Luego toma los valores de los elementos, desde las fuentes de entrada (como indica el primer engranaje).
    4. Evalúa los valores obtenidos de los distintos elementos, verificando en cada relación si se cumple algún limite (segundo engranaje).
    5. Por último y como resultado de la evaluación del punto anterior, si se llegó a un límite se ejecuta la función objetivo, como está indicado en el tercer engranaje.
- En todo momento se puede monitorear el estado de los distintos contextos desde el administrador web, de 2 formas distintas:

    - Para esto se puede ver la consola en tiempo real.
    - También se puede exportar el log, ya que se registra toda actividad relacionada con los contextos en el archivo destinado para el Log (LogContextModeling).
- Se puede exportar también la representación de uno o más contextos utilizando el administrador web, esto genera un archivo XML.
- Además, toda la información de los contextos, incluyendo los valores de los elementos se almacena en la base de datos del sistema.
- Es importante aclarar, que el Context Modeling Framework permite crear múltiples contextos, y monitorearlos a todos juntos.

5.2 Requerimientos

En esta sección se presentan el diagrama de clase, requerimientos funcionales, casos de uso, etc. utilizados en el desarrollo del Framework.

5.2.1 Diagrama de clases

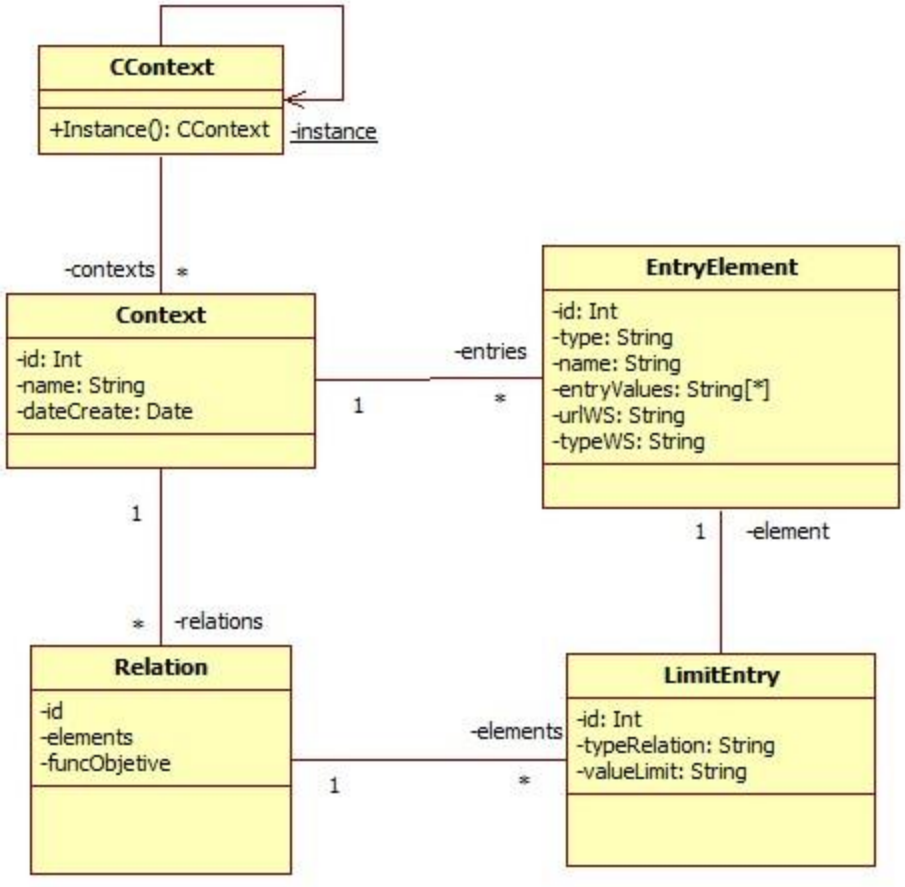


FIGURA 10 - DIAGRAMA DE CLASES (UML) CONTEXT MODELING FRAMEWORK

5.2.2 Requerimientos Funcionales

El Framework tiene como objetivo simplificar el trabajo de los usuarios, para que puedan enfocarse en su negocio, por esto el Framework se encarga del monitoreo de los contextos y la ejecución de las funciones objetivos. Para lograr esto, la herramienta abstrae la capa de la lógica compleja, ocultándola a los ojos del usuario.

A continuación se listan y detallan los requerimientos funcionales:

Alta de contexto:

RF-01	Alta Contexto	
Descripción	Se da de alta un nuevo contexto	
Precondición		
Secuencia Normal	Paso	Acción
	1	El usuario selecciona la opción “crear contexto”
	2	Se indica el nombre del contexto a crear
	3	Se agregan las fuentes de entrada que van a formar parte del contexto, se indica nombre, tipo (atributo cerrado), tipo de servicio (atributo cerrado) y url del servicio.<RF-09>
	4	Se crea la relación, para ello se definen los límites de los elementos y las funciones a ejecutar en caso de alcanzarlos<RF-10>.
	5	Se confirma el alta del contexto
	6	El sistema lo agrega al controlador de contextos, seteando la fecha en que fue creado.

		Se registra la actividad en el log. Se puede ver en la grilla de contextos
Post-condición	Se crea un nuevo contexto, este se agrega al controlador de contextos..	
Importancia	Importante	

Modificar Contexto:

RF-02	Modificar Contexto	
Descripción	Se modifican los datos de un contexto ya creado	
Precondición	Contexto creado	
Secuencia Normal	Paso	Acción
	1	Se ingresa a través de la pantalla principal donde se puede ver una grilla con los contextos creados y se selecciona el contexto que se quiere modificar
	2	Modificar nombre del Contexto
	3	Se pueden ingresar nuevas fuentes de entrada o modificar los que ya existen.<RF-09>
	4	Se pueden ingresar nuevos límites y funciones así como editar las que ya tiene el contexto<RF-10>
	5	Se confirman los datos ingresados. Se registra la actividad en el log.
	6	Se vuelve a la pantalla principal
Postcondición	Se modifica el contexto	

Importancia	Importante

Baja Contexto:

RF-03	Baja Contexto	
Descripción	Se elimina un contexto creado.	
Precondición	Contexto creado	
Secuencia Normal	Paso	Acción
	1	Se ingresa a través de la pantalla principal donde se puede ver una grilla con los contextos creados y se selecciona el contexto que se quiere eliminar
	2	Se elimina el contexto. El sistema lo borra del controlador. Se registra la actividad en el log.
Postcondición	Se elimina el contexto	
Importancia	Importante	

Guardar en XML:

RF-04	Guardar en XML	
Descripción	Se guardan los contexto creados en un XML	
Precondición	Contexto creado	
Secuencia Normal	Paso	Acción
	1	Se ingresa a través de la pantalla principal donde se puede ver una grilla con los contextos creados y se selecciona la opción de “guardar XML”

	2	El sistema sobrescribe el archivo que se encuentra en la dirección que indica el archivo de configuración confProp.properties. Se registra la actividad en el log.
Postcondición	Se crea el archivo XML.	
Importancia	Importante	

Cargar XML:

RF-05	Cargar XML	
Descripción	Se cargan los contexto a través de un archivo XML	
Precondición	No tiene.	
Secuencia Normal	Paso	Acción
	1	Se ingresa a través de la pantalla principal donde se puede ver una grilla con los contextos creados y se selecciona la opción de “Cargar XML”
	2	El sistema crea los contextos a partir del archivo que se encuentra en la dirección que indica el archivo de configuración confProp.properties. En el tag:dir Se registra la actividad en el log.
Postcondición	Se crean los contexto en el sistema	
Importancia	Importante	

Agregar Valor:

RF-06	Agregar Valor	
Descripción	Se agrega un nuevo valor a un elemento de un contexto dado	
Precondición	Contexto creado, elemento creado	
Secuencia Normal	Paso	Acción
	1	Se consume el servicio web publicado para este propósito. Los parámetros son nombre del contexto, nombre del elemento y valor.
	2	Se busca el contexto, el elemento y se monitorea el contexto<RF-07> Además se registra la actividad en el log.
Postcondición		
Importancia	Importante	

Monitoreo de contexto:

RF-07	Monitoreo de Contexto	
Descripción	Se monitorea el contexto, dado los valores que se ingresan verifica si se tiene que ejecutar la función objetivo.	
Precondición	Contexto creado, elemento creado	
Secuencia Normal	Paso	Acción
	1	Dado el contexto, el elemento y valor se verifica si con ese valor el elemento llega a un límite
	2	Se ejecutan las funciones objetivos que corresponden.

		Se registra la actividad en el log.
Postcondición		
Importancia	Importante	

Listar Contextos:

RF-08	Listar contextos	
Descripción	Se listan los contextos ya ingresados en el sistema	
Precondición	No tiene	
Secuencia Normal	Paso	Acción
	1	Acceder al dashboard (pantalla principal del administrador)
Postcondición		
Importancia	Importante	

Agregar fuente de entrada:

RF-09	Agregar fuente de entrada

Descripción	Dado un contexto, se agrega un nuevo elemento como fuente de entrada.	
Precondición	Contexto creado	
Secuencia Normal	Paso	Acción
	1	Listar contextos
	2	Seleccionar contexto para editarlo
	3	Agregar nuevo elemento de entrada. Se registra la actividad en el log.
Postcondición		
Importancia	Importante	

Agregar Relación:

RF-10	Agregar relación	
Descripción	Dado un contexto, se agrega una nueva relación	
Precondición	Contexto creado	
Secuencia Normal	Paso	Acción
	1	Listar contextos
	2	Seleccionar contexto para editarlo
	3	Agregar nueva relación indicando los elementos y sus límites.
	4	Indicar función objetivo. Se registra esto en el log.

Postcondición	
Importancia	Importante

5.2.3 Casos de Uso

En esta sección se incluyen los diagramas de casos de uso del sistema:

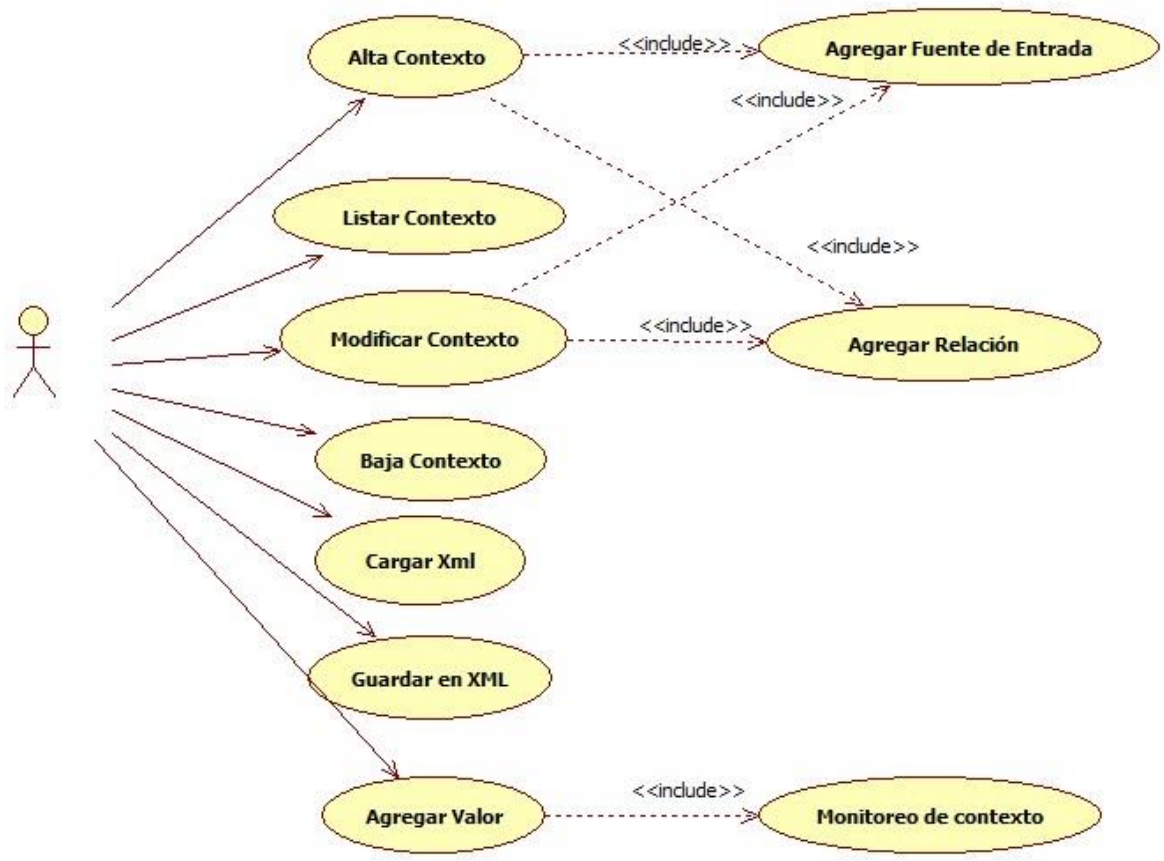


FIGURA 11 - DIAGRAMA DE CASOS DE USO

5.3 Herramientas utilizadas

Las herramientas utilizadas en el desarrollo del Framework son:

5.3.1 Apache Tomcat 6.0

Tomcat es un contenedor web basado en el lenguaje Java que actúa como motor de servlets y JSPs. Apache Tomcat se utiliza para servir Servlets y Java Server Pages.

¿Por qué se decidió utilizarlo?

- Servidor de aplicaciones “open source”.
- Fácil integración con Apache HTTP Server y con IIS.
- Muy estable en sistemas UNIX.
- Mucha y muy buena documentación, incluso foros activos.
- No requiere mucha memoria para funcionar.
- Gratuita

Se decidió utilizar este servidor, ya que estamos más familiarizados con él, y de esta manera se ahorra tiempo en cuanto a la curva de aprendizaje.

5.3.2 PrimeFaces 4.0

Es una librería de componentes para JavaServer Faces (JSF) de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web. Primefaces está bajo la licencia de Apache License V2.

En el desarrollo del Framework se utilizó la versión 4.0.

¿Por qué se decidió utilizarlo?

- Documentación muy práctica y foros activos.
- Algunas vistas requieren tablas dentro de otras, y Prime Faces brinda prolijidad para trabajar con las mismas.

- Gratuita.

Utilizamos Prime Faces ya que luego de investigar durante un tiempo, encontramos ejemplos prácticos que resolvían nuestras necesidades.

5.3.3 lOG4j

Log4J es una herramienta Open Source, que nos permite realizar un registro de las aplicaciones en tiempo de ejecución, y permite a los desarrolladores tanto en la fase de desarrollo como en ejecución, poder capturar, determinar o controlar eventos, comportamientos, o excepciones que puedan interferir en la normal ejecución de las mismas.

¿Por qué se decidió utilizarlo?

- Es gratuito
- Es muy simple de integrar, es configurable y que permite escalar en cuanto al registro de actividades en caso de querer extender o utilizar el Framework en una aplicación más grande.
- Mucha documentación, con ejemplos prácticos y concretos que ahorran mucho tiempo en cuanto a la curva de aprendizaje.

Realizamos pruebas con distintas librerías entre ellas SLF4j, LOG4j. Para registrar actividades son muy similares, por lo que la facilidad a la hora de configurar las librerías inclino la balanza por LOG4j.

6. Forma de uso y Funcionamiento

En esta sección describe la forma de utilizar el Framework y también se muestran ejemplos de funcionamiento.

6.1 Forma de uso

A continuación se describe la forma de uso de la herramienta, tanto para usuarios desarrolladores como para usuarios no desarrolladores.

El usuario deberá importar el proyecto desde eclipse (ver Anexo II con el Manual). Y luego deberá seguir los siguientes pasos:

1. Ejecutar el proyecto
2. Ingresar a "*dominio_donde_esta_hosteado/config*". En este paso tendrá un Wizard para crear nuevos contextos.

Pasos del Wizard:

Paso 1:

Colocar Nombre del Contexto, es un campo obligatorio y único, por lo cual no puede existir otro contexto con ese nombre.

FIGURA 12 - FORMA DE USO - WIZARD PASO 1

Paso 2:

Dar de alta los elementos (Fuentes de Entrada) que se van a relacionar, además se debe indicar si se va a consumir un servicio para añadir un nuevo valor al elemento de entrada (Debe seleccionar consumir y se le muestra la URL que debe utilizar), o si se va a publicar un servicio para que el intérprete lea el valor(y se debe indicar la URL del servicio).

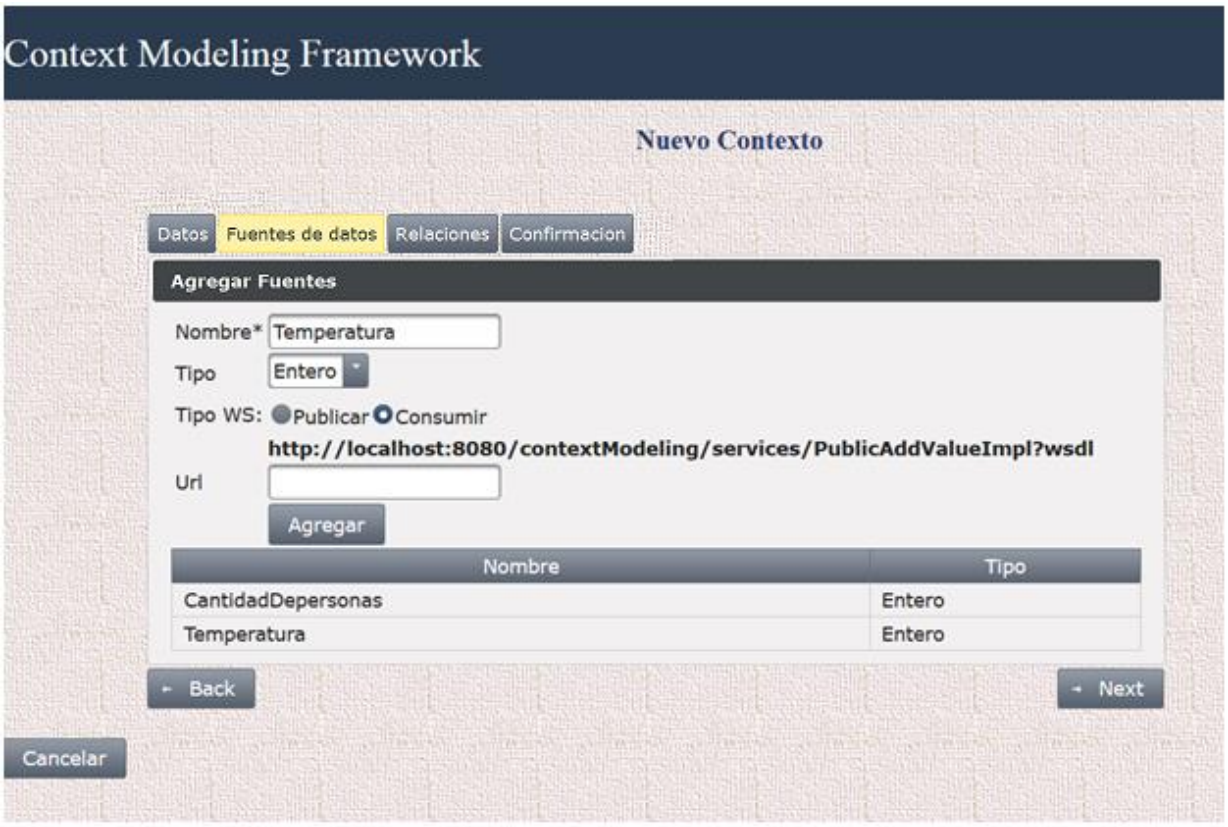


FIGURA 13 - FORMA DE USO - WIZARD PASO 2

Paso 3:

Dar de alta la relaciones, se configuran las relaciones de los elementos (Fuentes de entrada) dados de alta en el paso anterior. Se debe indicar la URL del servicio a invocar (por ejemplo, la URL del aire acondicionado para subir la temperatura)

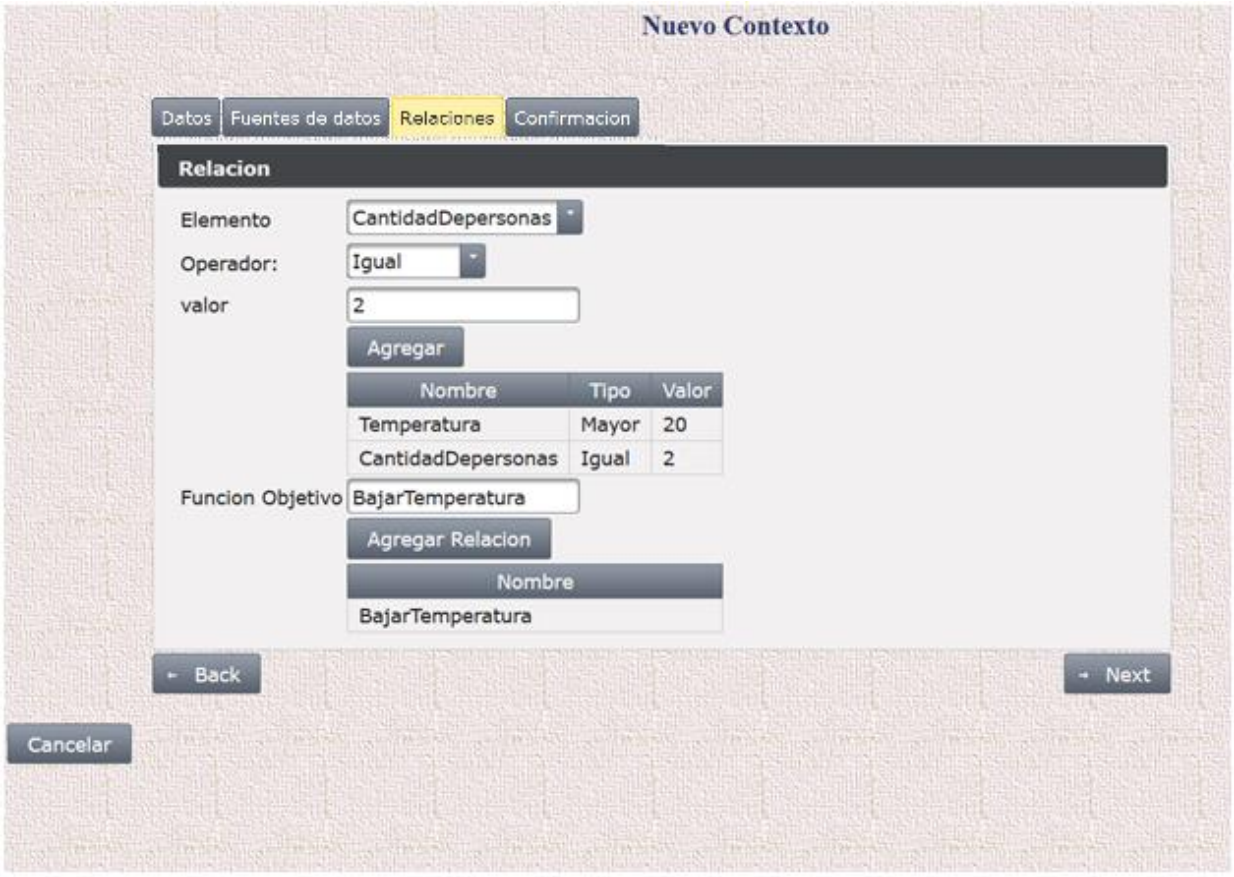


FIGURA 14 - FORMA DE USO - WIZARD PASO 3

Paso4:

Confirmación, queda dado de alta el contexto.

FIGURA 15 - FORMA DE USO - WIZARD PASO 4

Panel Principal:

Aquí puede ingresar distintos contextos, y configurarlos. De esta manera se va modificando el Config.xml. *Opcionalmente un usuario puede escribirlo a mano respetando su estructura (la cual fue explicada en la sección anterior).

Luego de configurados los contextos si el usuario desea modificarlos, puede hacerlo directamente en el XML, o seleccionando la opción editar desde el panel principal.

Si el usuario desea ver representado el contexto en un determinado momento, solo debe ingresar al Panel principal: "*dominio_donde_esta_hosteado/dashboard*" y seleccionar el contexto deseado.

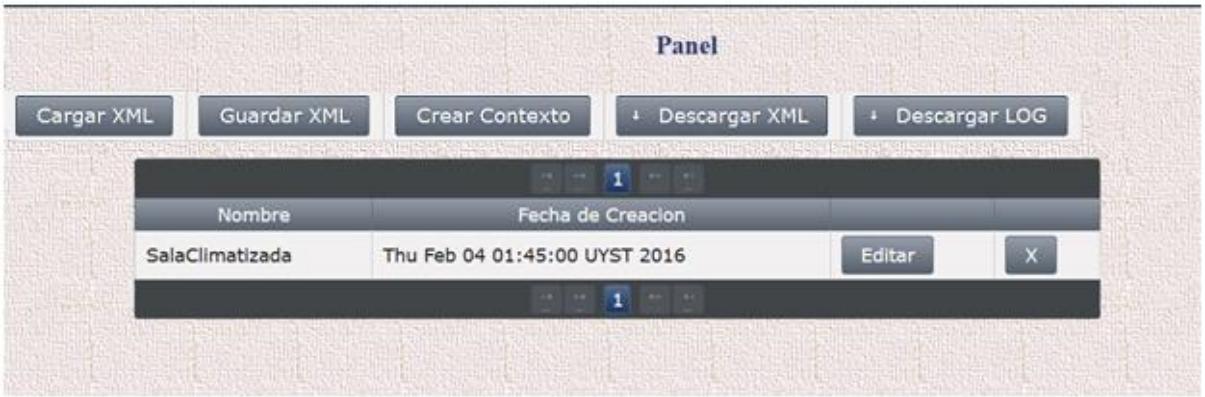


FIGURA 16 - PANEL PRINCIPAL

3. Una vez creado el contexto, ahora debe ingresar los valores, de la manera que lo indico al momento de agregar las fuentes de entradas, las opciones son las siguientes:
    a. Consumiendo servicio web: Se agrega el valor (Ej temperatura) utilizando el servicio provisto por el Framework.
    b. Publicando servicio web: El Framework cada X segundos (indicado en el properties) chequea los contextos configurados y consume servicio indicado para la fuente de entrada.

    En ambos casos el Framework ingresa el nuevo valor y evalúa las relaciones correspondientes, en caso de llegar a un límite se ejecuta la función objetivo indicado en la relación.

6.1.1 Forma de extenderlo para Usuarios desarrolladores

Además del uso normal indicado en el punto anterior, un usuario desarrollador podría extender las funcionalidades de esta herramienta, o utilizar esta como punto de partida para su propia aplicación.

Para esto, un usuario desarrollador puede extender las funciones relacionadas con el intérprete, ubicadas en el controlador de los contextos, llamado CContexts.java. Este archivo se puede encontrar en: *Interprete -> Controllers.*

Dentro de este controlador, el desarrollador deberá modificar las funciones:

- addValue: Destinada para obtener los valores desde las distintas fuentes de entrada.
- monitor: Encargada de evaluar los valores ingresados utilizando la función anterior, y luego interpretarlos según el caso:
    - Caso 1: Alcanzó limite, entonces debe ejecutar función objetivo.
    - Caso 2: No alcanzo limite, entonces simplemente lo ingresa y continua esperando información desde las fuentes de entrada.

A modo de ejemplo ilustrativo, supongamos que podemos correr el Framework en la computadora de un aire acondicionado que tiene las funciones públicas “subir_temperatura”, “bajar_temperatura”, y “mostrar_temperatura”.

En la función add_value debemos llamar a “mostrar_temperatura”, quedándonos con el valor resultado. Luego en la función monitor, según el límite alcanzado, debemos llamar la función “bajar_temperatura” o “subir_temperatura” según sea el caso.

Si bien en ese ejemplo la implementación es totalmente local, también se puede aplicar a desarrollos distribuidos, consumiendo o publicando servicios web para las funciones anteriores.

6.2 Context Modeling Framework como Software as a Service

El Context Modeling Framework cuenta con la capacidad de ser utilizado directamente desde la nube, utilizándolo como un SAAS[1]sin necesidad de instalarlo en una infraestructura propia.

Para utilizarlo como SAAS solo se debe acceder a la siguiente URL: saas.contexmf.com y podrán contar con los servicios mencionados.

6.3 Pruebas Realizadas

A continuación se muestran las pruebas realizadas con el fin de demostrar el correcto funcionamiento del Framework y también su utilidad.

6.3.1 Sala climatizada

Se describe el funcionamiento con un ejemplo práctico, en este caso una sala climatizada:

Descripción:

Se busca climatizar un salón (entre 18°C y 20°C) en caso de que se encuentre al menos una persona adentro.

La siguiente tabla muestra los elementos considerados por el Framework:

Elemento	**Descripción:**

[1] SAAS: Software as a Service: Es un modelo de distribución de software donde el soporte lógico y los datos que maneja se alojan en servidores de una compañía de tecnologías de información y comunicación (TIC), a los que se accede vía Internet desde un cliente.

Entorno:	Salón, aire acondicionado, cantidad de personas en el salón.
Fuentes de datos:	Sensor de temperatura del aire acondicionado, sensor de movimiento (para detectar entrada y salida de personas).
Objetivo:	Mantener la temperatura en entre 18 y 20°C en caso de que se encuentre una persona dentro del salón.
Usuario:	Persona que ingresa al salón.
Límite:	Límite inferior 18°C Límite superior 20°C

Funcionamiento del intérprete:

El intérprete monitorea el ingreso de personas y también la temperatura utilizando los sensores. En caso de que se encuentre una persona dentro del salón y además:

- La temperatura sea menor a los 18°C se dispara el evento "subir temperatura"
- La temperatura sea mayor a los 20°C se dispara el evento "bajar temperatura"

XML Generado:

Luego de creado el contexto, el Framework mediante la web de administración, permite exportar modelado el mismo. El XML generado es el siguiente, y se pueden ver los elementos antes mencionados:

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
	<context name="SalaClimatizada" date="01/02/2016 00:32">
		<entrys>
			<entry id="1">
				<name>SensorTemperaura</name>
				<type>Entero</type>
				<typeWS />
				<urlWS />
			</entry>
			<entry id="2">
				<name>ContadorPersonas</name>
				<type>Entero</type>
				<typeWS>consume</typeWS>
				<urlWS />
			</entry>
		</entrys>
		<relations>
			<relation>
				<funcObj>BajarTemperatura</funcObj>
				<limitEntrys>
				<limitEntry>
				<entryId>2</entryId>
				<typeRelation>Mayor</typeRelation>
```

```
                        <valueLimit>1</valueLimit>
                        </limitEntry>
                        <limitEntry>
                        <entryId>1</entryId>
                        <typeRelation>Mayor</typeRelation>
                        <valueLimit>20</valueLimit>
                        </limitEntry>
                        </limitEntrys>
                  </relation>
                  <relation>
                        <funcObj>SubirTemperatura</funcObj>
                        <limitEntrys>
                        <limitEntry>
                        <entryId>2</entryId>
                        <typeRelation>Mayor</typeRelation>
                        <valueLimit>1</valueLimit>
                        </limitEntry>
                        <limitEntry>
                        <entryId>1</entryId>
                        <typeRelation>Menor</typeRelation>
                        <valueLimit>18</valueLimit>
                        </limitEntry>
                        </limitEntrys>
                  </relation>
            </relations>
      </context>
</contexts>
```

Con el fin de probar funcionando el Framework, se decidió emular el comportamiento mencionado.

Para esto se desarrolló un proyecto que consiste en:

- Página web emulando una sala climatizada, donde se muestra un aire acondicionado con la temperatura actual y una persona dentro de la sala.
- Tres Servicios web:
  - Incrementar temperatura: Para sumar 1ºC a la temperatura actual.
  - Decrementar temperatura: Para restar 1ºC a la temperatura actual.
  - Ver temperatura: Para mostrar la temperatura actual.

Interacción con el Framework:

La “sala climatizada” interactúa con el Framework de la siguiente manera:

- Se crea el contexto sala climatizada, indicando que la fuente de entrada es un servicio publicado bajo la url:

http://sala.lh:8080/index.php/api/sala/ver_temp (Este invoca el servició add_value del Context Modeling Framework)

- Como relaciones se indican el mínimo y máximo de temperatura, y se indican los servicios a consumir para subir y bajar temperatura. En este caso:
    - Si temperatura menor a 18ºC se invoca el servicio: *http://192.168.1.46:8080/sorteoci/index.php/api/sala/incrementar/format/json*
    - Si temperatura mayor a 20ºC se invoca el servicio: *http://192.168.1.46:8080/sorteoci/index.php/api/sala/decrementar/format/json*

El Framework y la "sala" pueden no estar en la misma computadora, ya que se trata de un sistema distribuido que se puede integrar mediante servicios web como el caso anterior.

Casos de prueba:

Se realizaron casos de prueba modificando y llamando a "ver temperatura" que es la fuente de entrada del contexto en el Framework, y se monitorea la salida. Los resultados fueron los siguientes:

IdCaso	Descripción	PreCondiciones	Resultado Esperado	Resultado Obtenido	Estado
c1	Se crea el contexto sala en el cual la temperatura tiene que estar entre 18°C y 20°C		Se crea un contexto sala	Se creó el contexto	Finalizado
c2	Verificar que al superar los 20°C de temperatura se ejecuta la función objetivo decrementar	- Contexto sala creado. - Una persona dentro del salón.	La temperatura baja a 20°	La temperatura bajó a 20°	Finalizado
c3	Verificar que al bajar la temperatura a 17° se ejecuta la función objetivo incrementar	- Contexto sala creado. - Una persona dentro del salón.	La temperatura sube a 18°	La temperatura subió a 18°	Finalizado
c4	Verificar que si la temperatura esta entre 18° y 20° no se ejecuta la función objetivo	- Contexto sala creado. - Una persona dentro del salón.	La temperatura no cambia	La temperatura no cambio	Finalizado

Log:

Además de comprobar que los resultados son los esperados, se puede revisar el Log, el cual muestra todos los sucesos que pasaron por el Framework.

Un tramo del log es el siguiente:

2016-08-24 21:09:01 INFO Logger:63 - Creando Contexto

2016-08-24 21:09:01 INFO Logger:63 - Se creo el contexto con los siguientes datos:

-Nombre:sala

-Fecha de creación:Wed Aug 24 21:10:00 GMT-03:00 2016

-Entradas[temp]

-Relaciones:[funcion Objetivo:http://192.168.1.46:8080/sorteoci/index.php/api/sala/incrementar/format/json, Elementos=[[Elemento=temp, tipo de relacion=Menor, valor limite=18]],

funcion Objetivo:http://192.168.1.46:8080/sorteoci/index.php/api/sala/decrementar/format/json, Elementos=[[Elemento=temp, tipo de relacion=Mayor, valor limite=20]]]

2016-08-24 21:26:49 INFO Logger:63 - Para el elemento temp se carga el valor:21

2016-08-24 21:26:49 INFO Logger:63 - Se consume el servicio: http://192.168.1.43:8080/sorteoci/index.php/api/sala/decrementar/format/json

2016-08-24 21:26:49 INFO Logger:63 - Output from Server ....

2016-08-24 21:26:49 INFO Logger:63 - Respuesta del WS: "20"

2016-08-24 21:29:11 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:29:17 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:29:18 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:29:21 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:29:45 INFO Logger:63 - Para el elemento temp se carga el valor:21

2016-08-24 21:29:45 INFO Logger:63 - Se consume el servicio: http://192.168.1.43:8080/sorteoci/index.php/api/sala/decrementar/format/json

2016-08-24 21:29:45 INFO Logger:63 - Output from Server ....

2016-08-24 21:29:45 INFO Logger:63 - Respuesta del WS: "20"

2016-08-24 21:30:02 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:30:21 INFO Logger:63 - Para el elemento temp se carga el valor:20

2016-08-24 21:31:03 INFO Logger:63 - Para el elemento temp se carga el valor:17

2016-08-24 21:31:03 INFO Logger:63 - Se consume el servicio: http://192.168.1.43:8080/sorteoci/index.php/api/sala/incrementar/format/json

2016-08-24 21:31:03 INFO Logger:63 - Output from Server ....

Herramientas utilizadas:

Para el desarrollo de la sala climatizada se decidió utilizar:

PHP: Es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico.

¿Por qué?

Es simple, rápido a la hora de implementar una página web simple y servicios web.

Codeigniter: Es un Framework para el desarrollo de aplicaciones en php que utiliza el MVC (Modelo Vista Controlador).

¿Por qué?

Simplifica mucho el trabajo al desarrollador, ya que el lenguaje para trabajar con él es PHP, el cual es muy simple. Además, al estar basado en el patrón Modelo Vista Controlador, permite separar fácilmente la lógica de la siguiente manera:

- Modelo: manejamos la base de datos donde solo se guarda la temperatura actual
- Vistas: Una página web simple con la sala climatizada

- Controlador: Se desarrolló un controlador simple que provee los servicios web para manipular y ver la temperatura, y uno para modificar en la página el valor de la temperatura.

6.3.2 Contenedor de residuos

La idea es mejorar la logística de una empresa recolectora de residuos, ya que es un problema crítico en la actualidad.

Descripción:

En este caso el contenedor recibe información de la cantidad de residuos que contiene. Al llegar a 500 kg, o cuando se alcance un 80% del volumen máximo, le avise a la empresa recolectora para que proceda con la recolección y vaciado del mismo, mejorando así la logística y evitando contenedores desbordados.

La siguiente tabla muestra los elementos considerados por el Framework:

Elemento	**Descripción:**
Entorno:	Contenedor, residuos.
Fuentes de datos:	Sensor de peso (balanza), sensor de volumen.
Objetivo:	Evitar desbordamiento del contenedor, controlando el peso y el volumen máximo.
Usuario:	Empresa recolectora.
Límite:	* Limite de peso 500 kg * Limite de volumen 80%

Funcionamiento del intérprete:

El intérprete monitorea el ingreso de residuos, entonces:

- Si detecta que se alcanzaron los 500kg, se dispara el evento “recolectar”.
- Si detecta que se alcanzó el 80% de la capacidad del mismo, se dispara el mismo evento (recolectar).

XML Generado:

```
<?xml version="1.0" encoding="UTF-8"?>
	<contexts>
	<context name="Contenedor" date="25/08/2016 17:37">
	<entrys>
		<entry id="2">
			<name>peso</name>
			<type>Entero</type>
			<typeWS>public</typeWS>
			<urlWS />
		</entry>
		<entry id="3">
			<name>volumen</name>
			<type>Entero</type>
			<typeWS>public</typeWS>
			<urlWS />
		</entry>
	</entrys>
	<relations>
		<relation>
			<funcObj>recolectar</funcObj>
			<limitEntrys>
				<limitEntry>
					<entryId>3</entryId>
					<typeRelation>Mayor</typeRelation>
					<valueLimit>80</valueLimit>
				</limitEntry>
			</limitEntrys>
		</relation>
		<relation>
			<funcObj>recolectar</funcObj>
			<limitEntrys>
				<limitEntry>
					<entryId>2</entryId>
					<typeRelation>Mayor</typeRelation>
					<valueLimit>500</valueLimit>
				</limitEntry>
			</limitEntrys>
		</relation>
	</relations>
	</context>
	</contexts>
```

Con el fin de probar funcionando el Framework, se decidió emular el comportamiento mencionado.

En este caso no se necesitó desarrollar ningún servicio web, sino que se utiliza directamente el Context Modeling Framework de la siguiente manera:

Interacción con el Framework:

Cuando se alcanzan los 500 kg, o cuando se llega al 80%, se invoca el servicio publicado por el Framework, addValue. Este detecta que se llegó al límite y dispara el evento "recolectar" que para simplificar las pruebas solo manda un mail al encargado de la empresa recolectora.

Para esto se debió extender el Framework agregando la función "recolectar" que lo que hace es mandar el correo electrónico, indicando el id del contenedor y que llego al límite.

Casos de prueba:

Se realizaron casos de prueba, invocando la función addValue mediante el servicio web:

http://localhost:8080/contextModeling/services/PublicAddValueImpl

Y se monitoreó la salida los resultados son los siguientes:

IdCaso	Descripción	PreCondiciones	Resultado Esperado	Resultado Obtenido	Estado
c5	Se crea el contexto Contenedor, donde el peso no puede ser mayor a 500Kg y el volumen ocupado no puede superar el 80%		Se cree un contexto Contenedor	Se creó el contexto Contenedor	Finalizado
c6	Verificar que al superar los 500Kg se ejecuta la función objetivo: Recolectar()	Contexto Contenedor creado	Se enviara un correo a la recolectora indicando el contenedor que requiere atención	Se envió el correo	Finalizado
c7	Verificar que al no superar los 500Kg NO se ejecuta la función objetivo: Recolectar()	Contexto Contenedor creado	No se ejecuta la función	No se ejecutó la función	Finalizado
c8	Verificar que al superar el 80 % del espacio se ejecuta la función objetivo: Recolectar()	Contexto Contenedor creado	Se enviara un correo a la recolectora indicando el contenedor que requiere atención	Se envió el correo	Finalizado

c9	Verificar que al no superar el 80 % del espacio NO se ejecuta la función objetivo: Recolectar()	Contexto Contenedor creado	No se ejecuta la función	No se ejecutó la función	Finalizado

Log:

Además de comprobar que los resultados son los esperados, se puede revisar el Log, el cual muestra todos los sucesos que pasaron por el Framework.

Un tramo del log es el siguiente:

2016-08-25 17:36:01 INFO Logger:63 - Creando Contexto

2016-08-25 17:37:54 INFO Logger:63 - Se creo el contexto con los siguientes datos:

-Nombre:Contenedor

-Fecha de creación:Thu Aug 25 17:37:00 GMT-03:00 2016

-Entradas[peso, volumen]

-Relaciones:[funcion Objetivo:recolectar, Elementos=[[Elemento=volumen, tipo de relacion=Mayor, valor limite=80]], funcion Objetivo:recolectar, Elementos=[[Elemento=peso, tipo de relacion=Mayor, valor limite=500]]]

2016-08-25 17:48:59 INFO Logger:63 - Se guardan los contextos en un archivo xml

2016-08-25 17:48:59 INFO Logger:63 - Se guarda el archivo xml

2016-08-25 17:56:51 INFO Logger:63 - Para el elemento peso se carga el valor:400

2016-08-25 17:57:04 INFO Logger:63 - Para el elemento peso se carga el valor:520

2016-08-25 17:57:04 INFO Logger:63 - Se llama a la funcion objetivo Recolectar()

2016-08-25 17:57:04 INFO Logger:63 - Se envia correo a la recolectora

2016-08-25 17:57:47 INFO Logger:63 - Para el elemento volumen se carga el valor:70

2016-08-25 17:58:06 INFO Logger:63 - Para el elemento volumen se carga el valor:90

2016-08-25 17:58:06 INFO Logger:63 - Se llama a la funcion objetivo Recolectar()

2016-08-25 17:58:06 INFO Logger:63 - Se envia correo a la recolectora

7. Conclusiones y trabajo a futuro

Al inicio del proyecto se definieron objetivos, en este capítulo se presentan las conclusiones y resultados, así como también se plantean posibilidades de mejoras de cara al futuro.

7.1 Conclusiones

El desarrollo de este proyecto permitió cumplir los objetivos formulados inicialmente, por lo cual llegamos a las siguientes conclusiones:

- Comprender que es un contexto, y su importancia:

  La gran cantidad de información que forma los distintos tipos de contextos, lleva a que existan diversas definiciones sobre los mismos, que si bien tienen puntos en común, todas se enfocan según el perfil que el autor desea darle, o el tipo de contexto al que refiere. A los efectos del presente proyecto también realizamos nuestra definición, que si bien puede verse incluida en otras, se enfoca particularmente en la tecnología.

  Respecto a la importancia quedo muy claro que el contexto juega un rol fundamental en la vida de las personas, ya que todos cambiamos las decisiones en base al contexto en el que nos encontramos. También cumple un rol fundamental en las aplicaciones sensibles al contexto, particularmente en la computación ubicua.

- Realizar un estudio del estado del arte de las herramientas existentes utilizadas para la representación de los contextos:

  Se realizó dicho estudio, se analizaron las arquitecturas, funcionalidades de las mismas, y se intentó utilizar para modelar los contextos, sin tener éxito, ya que las únicas que nos ofrecieron información concreta al respecto de cómo forma el modelo, fueron GAIA y SOCAM, pero ninguno de las dos nos resultó útil. Esto se debe a que su foco no está en modelar el contexto, sino ser herramientas para utilizarlas con otros fines como muestra su documentación. También se analizó la posibilidad de hacerlo combinando herramientas, pero no fue posible, y por ellos se decidió implementar una herramienta que cuente con esa posibilidad.

- Encontrar una forma de representar genéricamente los contextos:

  Se encontró una forma, de esta manera y con el modelo encontrado se logra abstraer al usuario de la complejidad del entorno, por lo cual es un elemento muy útil para el desarrollo de aplicaciones sensibles al contexto. Esto también contribuye con la computación ubicua, el cual era otro de los objetivos planteados.

- Implementar un Framework que modele de forma genérica los contextos:

  Utilizando el modelo mencionado, se desarrolló la herramienta “Context Modeling Framework” la cual como ya vimos ofrece una capa de abstracción para tomar los valores de los distintos elementos del contexto, interpretarlos, evaluarlos y actuar en caso de que así lo quiera el usuario cuando se da alguna relación.

  De esta manera los usuarios pueden canalizar sus esfuerzos en lo que realmente quieren implementar, sin tener que preocuparse por modelar el contexto, o por evaluar las distintas fuentes de entrada.

  Con esta herramienta se logró cumplir y demostrar el objetivo de contribuir con las aplicaciones sensibles al contexto, y también con la computación ubicua.

Se pudieron cumplir los objetivos propuestos en cuanto al desarrollo de la herramienta, los cuales consistían en:

- Modelar, interpretar y monitorear contextos, realzando acciones en caso de que el contexto lo requiera.

  Se demostró mediante pruebas prácticas que la herramienta puede modelar un contexto, y luego interpretar las fuentes de entrada para luego ejecutar las funciones objetivo en caso de que el contexto así lo requiera.

  Para monitorearlo se ofrece la posibilidad de hacerlo de distintas maneras, como la consola y exportación del log, que muestra una foto del momento en que se imprime, además de guardar los datos pasados el tiempo que el usuario lo requiera.

Además, en el transcurso de la evolución del Context Modeling Framework, nos trazamos nuevos objetivos, que consideramos pueden ser un aporte importante para el crecimiento de la computación ubicua:

- No limitar el uso solo para usuarios desarrolladores:

  El Context Modeling Framework está implementado de tal forma que no es necesario ser un desarrollador para crear un contexto y verlo modelado. Para cumplir con esto, se implementó la ya mencionada web de administración, que cuenta con un Wizard para la creación y edición de contextos, la cual permite hacerlo siguiendo los pasos indicados, y por lo tanto cualquiera puede utilizar la herramienta

- Brindar escalabilidad a la herramienta:

  Como se menciona en el capítulo 5, para el desarrollo de esta herramienta se utilizó una estructura MVC, lo que permite a cualquier desarrollador localizar las vistas, controladores y modelos fácilmente, y así poder extenderlo para lo que deseé.

Además, como lo indica la documentación de la herramienta, la misma puede utilizarse como parte de otro proyecto, o también importarlo y tomarlo como punto de partida para un nuevo proyecto.

- Context Modeling Framework como Software as a service:

Una característica que tiene la herramienta implementada, es que se puede utilizar directamente desde la nube, sin necesidad de contar con infraestructura para instalarla. Esto trae ventajas y desventajas:

  - Ventajas:
    - Al no ser necesario instalarla, no es necesario contar con infraestructura para la herramienta, lo que permite un ahorro considerable en recursos y tiempo.
    - No es necesario que el cliente cuente con un área especializada de soporte para el sistema, por lo que se reducen sus costes y riesgo de inversión.
    - Se puede utilizar de forma distribuida con distintos clientes, es decir manipular los contextos desde distintos lugares.
  - Desventajas:
    - Al depender de la conexión a internet, tiene una latencia considerable, por lo que no es recomendable utilizarla desde la nube en caso de que se trate de contextos que requieran acciones inmediatas, ya que puede tomar unos segundos invocar la función objetivo o ingresar el valor.
    - Al depender de la conexión a internet, si esta se cae no se puede utilizar la herramienta.
    - Al estar en la nube, el usuario no puede extender las funcionalidades, como si puede hacerlo en caso de tenerla instalada localmente o en su propio servidor.

Por lo ya expuesto, podemos concluir también que tanto el modelo encontrado como la herramienta pueden colaborar a distintos niveles con la evolución de la computación ubicua.

Una motivación planteada al desarrollar el Framework, consiste en contribuir con la mejora en la calidad de vida de los seres humanos, y consideramos que se puede lograr en las distintas áreas:

- **Medio ambiente:**
  "Contenedor Inteligente": El sistema puede avisar cuando un contenedor o un determinado grupo de contenedores (barrio, calle, etc) están llegando a su límite, para así mandar un camión a recolectar los residuos, evitando contenedores desbordados, consumo excesivo e innecesario de combustible por ir a recolectar cuando los contenedores están vacíos, etc.

- **Salud**:
  El Framework puede formar parte de un sistema de salud que avise cuando se está por agotar un medicamento, o si a una persona le subió la temperatura, aumento el ritmo cardiaco, etc.

Salas climatizadas, ya sean de operación o para el cuidado de los pacientes regulando la temperatura como se mostró anteriormente.

- **Transito**:
  Los accidentes de tránsito se dan por imprevistos, ya que suceden cosas que los conductores no pudieron prever. Con el Context Modeling Framework se pueden mostrar a los conductores problemas que se avecinan, ayudando así a prever los "imprevistos" más comunes. La inversión en las herramientas necesarias para que un Framework como este colabore con la regulación del tránsito puede ayudar considerablemente a evitar estos problemas. Incluso se podría probar en tramos de las rutas más problemáticas, como para hacer un plan piloto y comprobarlo.

- **Calidad de vida**:
  Las casas inteligentes hoy en día no son ninguna novedad, pero hoy en día solo las personas con mucho dinero pueden acceder a ellos. La colaboración del Framework en este tema es rapidez con la que se podrían implementar estos sistemas, haciéndolo llegar a una mayor proporción de la población, y no solo a los que tienen mayor potencial económico.

- **Otros aspectos donde consideramos que puede colaborar:**
  Puede colaborar en el área de robótica, pilotos automáticos, controles de estabilidad de aviones, etc. La idea es que se instale y se deje corriendo la herramienta, configurándole las variables a considerar y los valores limites, para que luego se ejecute la función correspondiente.

7.1.1 Comparación entre el Context Modeling Framework y las herramientas estudiadas

A Continuación se presenta una comparación de las principales características entre la herramienta desarrollada y las herramientas investigadas en capítulo 2.

Nombre	Modela	Interpreta	Exportar Modelo	Monitoreo	Funciona localmente	Funciona en la nube	Disponible
Context Toolkit	X	X		X	X		X
TCMF	X	X		X	X		
Aura	X	X			X		
Gaia	X	X			X		
Socam	X	X			X		
Cobra	X	X	X		X		X
Context Modeling Framework	X	X	X	X	X	X	X

Tabla 2- comparativa de herramientas estudiadas vs Context Modeling Framework

7.2 Posibles Mejoras

De cara al futuro se podría mejorar la herramienta, utilizando algunas de las sugerencias que se detallan a continuación:

7.2.1 Pasos a seguir

El Framework se podría mejorar, si en un futuro se le pudiera agregar ontologías, para la representación de relaciones y propiedades.

Por motivos de tiempo a la hora de desarrollar el prototipo del Framework decidimos ir por XML ya que lo manejamos habitualmente y nos resulta más sencillo, pero luego de la investigación podemos afirmar que las ontologías mejorarían y facilitarían la tarea de resolver las relaciones.

7.2.2 Seguridad

Sería una buena mejora implementar algún sistema de seguridad sobre los servicios web (tanto para el ingreso de datos, como para las funciones objetivo), entre los que se podría utilizar están los siguientes:

7.2.2.1 WS Security

WS Security (15) , es un protocolo de comunicación que suministra un medio para aplicar seguridad a los servicios web.

Define una especificación que implementa una serie de mejoras a SOAP con el objetivo de mejorar la protección de los mensajes.

7.2.2.2 SSL (Secure Sockets Layer)

SSL (16), es un protocolo diseñado para para dar seguridad a la transmisión de datos en transacciones, utilizando criptografía de llave pública.

Permite que los datos que se transfieren entre un cliente y un servidor permanezcan privados.

7.2.2.3 WS-Addressing

WS Addressing (17), ofrece seguridad extremo a extremo a la mensajería SOAP.

7.2.3 Funcionalidades extra

Si bien el Framework es abierto, por lo que cualquier desarrollador puede extenderlo a su gusto, se detectaron algunas funcionalidades que podrían ser de utilidad:

- **Categorizar los límites de las relaciones:**
  Se podría poner un checkbox "critico" para indicar cuando un límite es crítico, y a estas relaciones, cuando se cumple el limite además de ejecutar la funcionalidad establecida en la función objetivo, enviar algún tipo de alerta por email, o incluso por SMS utilizando alguna herramienta como Twilio, o similar.

- **Probabilidades de error en las fuentes de datos:**
  En nuestro trabajo, la calidad de la información ingresada por las fuentes de entrada, no es evaluada en cuanto a la calidad de la misma, es decir que se utiliza tal cual ingresa.
  Generalmente en este tipo de aplicaciones se toma la información de sensores, que no siempre dan la información correcta, ya que tienen márgenes de error, los cuales no se tienen en cuenta.
  En el futuro, creemos que puede ser una mejora importante aplicar alguna técnica para mitigar estos errores.

- **Subir el Framework a la nube:**

  Si bien lo publicamos en un servidor nuestro para brindar el servicio en la nube, se podría subir a un servidor dedicado, con mayores recursos para poder brindar un mejor servicio.

  El mismo deberá contar con mayor velocidad de respuesta, y disponibilidad 24*7.

8 Glosario

Computación Ubicua (ubicomp)

Es entendida como la integración de la informática en el entorno de la persona, de forma que los ordenadores no se perciban como objetos diferenciados. Esta disciplina se conoce en inglés por otros términos como Pervasive computing, Calm technology, Things That Think y Everyware. Desde hace unos años también se denomina inteligencia ambiental.

Framework

Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Desde el punto de vista del desarrollo de software, un Framework es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado. Suelen incluir soporte de programas, bibliotecas, lenguaje de scripting, software para desarrollar y unir diferentes componentes de un proyecto de desarrollo de programas. Permiten facilitar el desarrollo de software, evitar los detalles de bajo nivel, permitiendo concentrar más esfuerzo y tiempo en identificar los requerimientos de software.

Situación

La forma en la que se dispone algo en un determinado espacio.

Entorno

Según la real academia española:

1. m. Ambiente, lo que rodea.

2. m. Inform. Conjunto de características que definen el lugar y la forma deejecución de una aplicación.

3. m. Mat. Conjunto de puntos vecinos a otro.

Arquitectura

Forma en que se representa la estructura o estructuras del sistema que consiste en componentes de **software**, las propiedades externas visibles de esos componentes y las relaciones entre ellos.

Software

Conjunto de programas, instrucciones y reglas informáticas paraejecutar ciertas tareas en una computad ora.

Wizard

Aplicación que guía paso a paso a un usuario para realizar una determinada tarea.

Infraestructura

Conjunto de medios técnicos, servicios e instalaciones necesarios para el desarrollo de una actividad o para que un lugar pueda ser utilizado.

Herramienta

Conjunto de instrumentos que se utilizan para desempeñar un oficio o un trabajo determinado.

Middleware:

Es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos.

Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red). El middleware abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, así como de los sistemas operativos y lenguajes de programación, proporcionando una API para la fácil programación y manejo de aplicaciones distribuidas.

9 Bibliografía

1. Schilit. B., Theimer. M. Disseminating Active Map Information to Mobile Hosts. - *Netwrk. Mag. of Global Internetwkg.* Setiembre 1994.

2. Anind K. Dey and Gregory D. Abowd. Towards a Better Understanding of Context and Context-Awareness. - *Handheld and Ubiquitous Computing, First International Symposium.* 1999.

3. Paul Dourish. What We Talk About When We Talk About Context. - *Personal and Ubiquitous Computing.* Febrero 2004.

4. Strang, Thomas - Linnhoff-Popien, Claudia. A Context Modeling Sourvey. - *Workshop Proceedings.* 2004.

5. El contexto y su contexto - Observatorio tecnológico. https://lacofa.fundaciontelefonica.com/2010/07/15/el-contexto-y-su-contexto/. [Ultima visita Agosto 2016]..

6. The Context Toolkit. 2015. http://contexttoolkit.sourceforge.net/. [Ultima visita Agosto 2016].

7. SourceForge. https://sourceforge.net/projects/contexttoolkit/. [Ultima visita Agosto 2016].

8. Korpipaa, P. and Mantyjarvi, J. and Kela, J. and Keranen, H. and Malm, E.-J. Managing context information in mobile devices. - *IEEE Pervasive Computing.* Julio de 2003.

9. João Pedro Sousa and David Garlan. Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments. - *Proceedings of the IFIP 17th World Computer Congress - TC2 Stream / 3rd IEEE/IFIP Conference on Software Architecture: System Design, Development and Maintenance.* 2002.

10. Gaia. http://gaia.cs.illinois.edu/. [Ultima visita Agosto 2016].

11. Tao Gu,Xiao Hang Wang,Hung Keng Pung,Da Qing Zhang. A service-oriented middleware for building context-aware services . - Journal of Network and Computer Applications. Enero 2005

12. CHEN, HARRY and FININ,TIM and JOSHI,ANUPAM. An ontology for context-aware pervasive computing environments. - *The Knowledge Engineering Review.* Setiembre 2003.

13. OWL. https://www.w3.org/TR/2003/CR-owl-features-20030818/. [Ultima visita Agosto 2016].

14. XML - w3c. https://www.w3.org/TR/REC-xml/. [Ultima visita Setiembre 2016].

15. WS-Security. http://cxf.apache.org/docs/ws-security.html. [Ultima visita Agosto 2016].

16. SSL. http://info.ssl.com/article.aspx?id=10241. [Ultima visita Agosto 2016].

17. WS-ADDRESSING. https://www.w3.org/Submission/ws-addressing/. [Ultima visita Agosto 2016].

10 Anexo I

10.1 Ejemplos

A continuación se presentan dos ejemplos y su comportamiento en el Framework.

Ejemplo 1

Se quiere mejorar el tránsito en las autopistas, para esto se implementa una “ruta inteligente”

Descripción:

Existen 2 carriles, el de la izquierda no permite ir a menos de 110 km/h (carril ágil), y el de la derecha es para circular a menos de 110 km/h.

La siguiente tabla muestra los elementos considerados por el Framework:

Elemento	Descripción:
Entorno:	Ruta inteligente, sensores de velocidad, autos.
Fuentes de datos:	Sensores de velocidad, sensor de movimiento que detecta por cual carril va el vehículo.
Objetivo:	Mantener orden en el tránsito, llevando los autos que van a menos de 110 km/h hacia el carril derecho.
Usuario:	Vehículos.
Límite:	110 km/h.

Funcionamiento del intérprete:

La ruta mide la velocidad de los autos:

- Auto que va por la izquierda y baja de 110 km/h lo acelera, y en caso de no poder subirle la velocidad lo obliga a pasarse al carril de la derecha
- Auto que va por la derecha a más de 110 km/h lo mueve al carril de la izquierda.

XML Generado:

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
	<context name="Ruta Inteligente" date="01/02/2016 00:34">
	<entrys>
		<entry id="1">
		<name>Sensor Velocidad</name>
		<type>Entero</type>
		<typeWS>consume</typeWS>
		<urlWS />
		</entry>
			<entry id="2">
		<name>Carril</name>
		<type>Entero</type>
		<typeWS>consume</typeWS>
		<urlWS />
		</entry>
	</entrys>
	<relations>
		<relation>
			<funcObj>Mover_A_Carril_Derecho</funcObj>
			<limitEntrys>
				<limitEntry>
				<entryId>1</entryId>
				<typeRelation>Menor</typeRelation>
				<valueLimit>110</valueLimit>
				</limitEntry>
					<limitEntry>
				<entryId>2</entryId>
				<typeRelation>Igual</typeRelation>
				<valueLimit>1</valueLimit><!-Carril Izquierdo ->
				</limitEntry>
			</limitEntrys>
		</relation>
		<relation>
		<funcObj>Mover_A_Carril_Izquierdo</funcObj>
		<limitEntrys>
		<limitEntry>
		<entryId>1</entryId>
		<typeRelation>Mayor</typeRelation>
		<valueLimit>110</valueLimit>
		</limitEntry>
			<limitEntry>
			<entryId>2</entryId>
			<typeRelation>Igual</typeRelation>
			<valueLimit>0</valueLimit><!-Carril Derecho ->
		</limitEntry>
```

```
            </limitEntrys>
            </relation>
      </relations>
      </context>
</contexts>
```

Ejemplo2

Estabilizador de un avión.

Descripción:

Se busca mantener la estabilidad de un avión, es decir no permitir que sus alas se inclinen para ningún lado más de 45°.

La siguiente tabla muestra los elementos considerados por el Framework:

Elemento	Descripción:
Entorno:	Alas del avión, nubes, horizonte, etc.
Fuentes de datos:	Sensor de inclinación de las alas.
Objetivo:	Mantener el avión entre -45° y +45° de inclinación, en caso de que se exceda el objetivo es inclinarlo hacia el otro lado manteniendo la estabilidad.
Usuario:	Avión.
Límite:	Límite izquierda - 45°C Límite derecha - 45°C

Funcionamiento del intérprete:

El intérprete monitorea la inclinación de las alas, en caso de que estas se muevan si:

- Baja el ala izquierda a menos de 45°, entonces invoca la función objetivo “levantar ala izquierda”
- Baja el ala derecha a menos de 45°, entonces invoca la función objetivo “levantar ala derecha”

XML Generado:

```
<?xml version="1.0" encoding="UTF-8"?>
<contexts>
	<context name="Estabilizador" date="01/02/2016 00:33">
		<entrys>
			<entry id="1">
				<name>Sensor Inclinacion</name>
				<type>Entero</type>
				<typeWS>consume</typeWS>
				<urlWS />
			</entry>
		</entrys>
		<relations>
		<relation>
			<funcObj>SubirAlaIzquierda</funcObj>
			<limitEntrys>
				<limitEntry>
				<entryId>1</entryId>
				<typeRelation>Menor</typeRelation>
				<valueLimit>-45</valueLimit>
			</limitEntry>
			</limitEntrys>
		</relation>
		<relation>
				<funcObj>BajarAlaIzquierda</funcObj>
				<limitEntrys>
					<limitEntry>
					<entryId>1</entryId>
					<typeRelation>Mayor</typeRelation>
					<valueLimit>45</valueLimit>
				</limitEntry>
			</limitEntrys>
		</relation>
		</relations>
	</context>
</contexts>
```

11 Anexo II

11.1 Manual de Instalación

Para instalar el Context Modeling Framework se deben seguir los siguientes pasos:

1- **Descargar Eclipse**

http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/luna/SR2/eclipse-jee-luna-SR2-win32.zip

2- **Descargar jdk 1.6**

http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase6-419409.html#jre-6u45-oth-JPR

3- **Descargar el zip con el proyecto y descomprimirlo en el workspace del eclipse.**

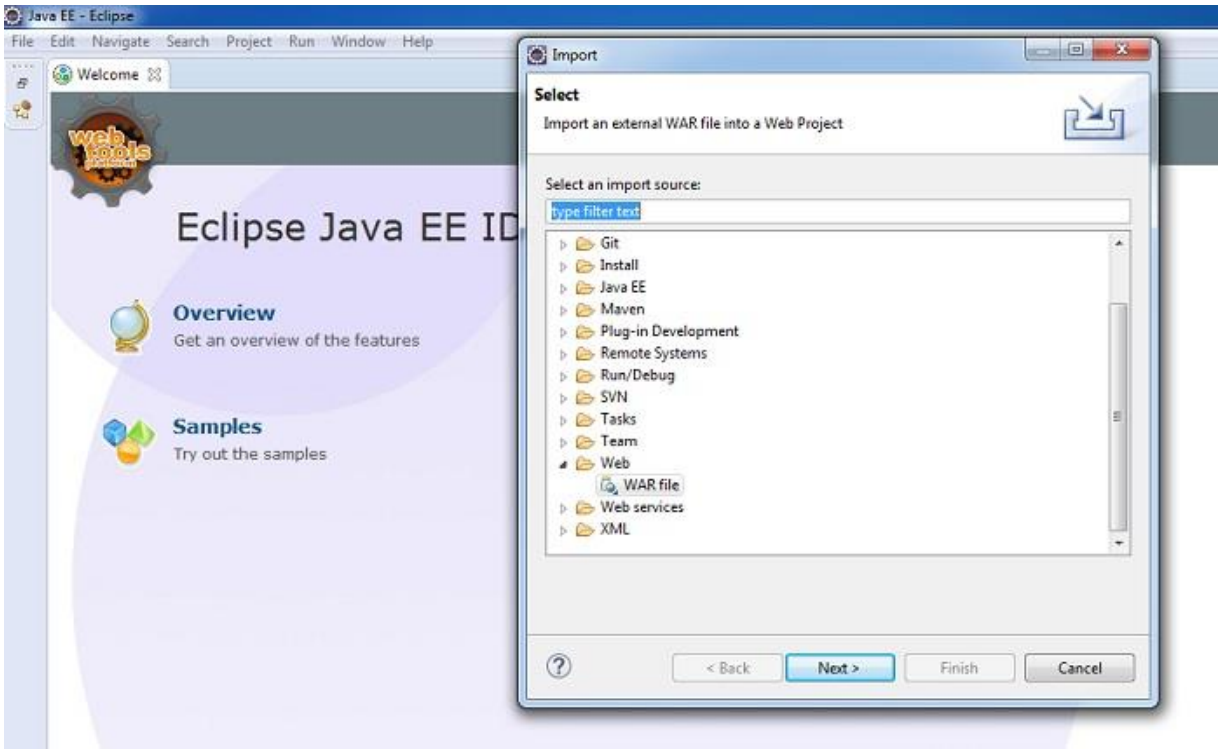
Deben descargar el zip, descomprimirlo dentro del workspace del eclipse.

Una vez descargado y copiado, se debe proceder a importar el proyecto.

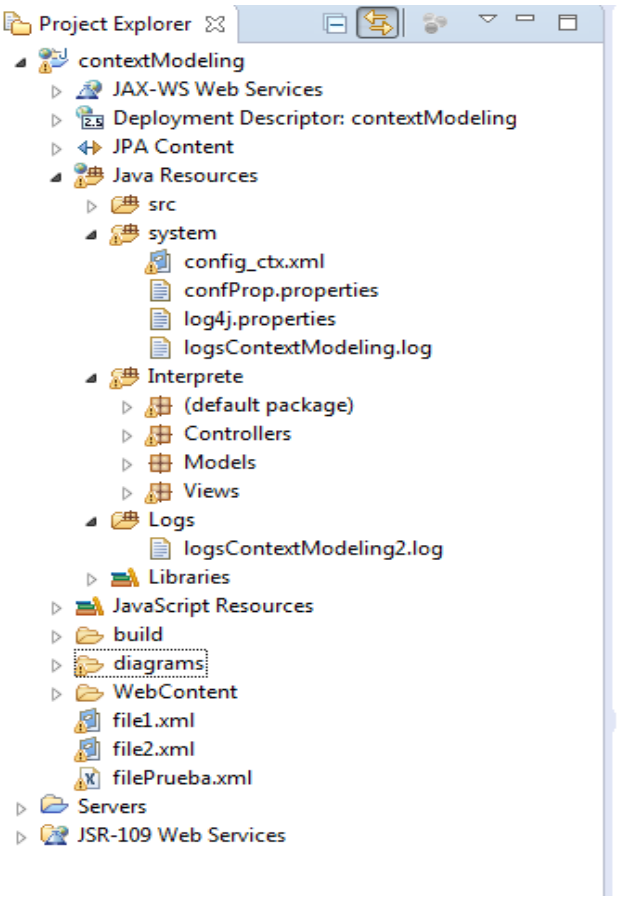
4- **Importar proyecto**

Ejecutar eclipse y una vez abierto importar el proyecto de la siguiente manera:

“File->Import->General->Existing project into workspace”



Ahora ya se puede ver cómo está organizado el proyecto



5- Instalar Apache Tomcat 6.

Para poder ejecutarlo localmente, solo resta instalar un servidor Apache Tomcat.

Se recomienda de la siguiente manera:

Ir a la pestaña “Servers” y seleccionar “Nuevo Servidor”.

Luego seleccionar Apache – Tomcat 6.

A continuación, se debe serleccionar “Download and Install” y listo, está instalado el servidor Apache Tomcat 6.

6- Configurar propiedades

Se deben configurar las rutas donde se guardan los archivos XML, y el Log.

- LOG:

1- Dentro de la carpeta System, se encuentra el archivo “confProp.properties”
Se debe poner la ruta donde se guardara el log en la sección “dirlog=AQUÍ VA LA RUTA”
2- La misma ruta se debe configurar en:
System -->“log4j.properties” en la sección “log4j.appender.file.File=AQUÍ VA LA RUTA”

- XML: En el mismo archivo se debe cambiar la ruta donde se guardará el XML: en la sección “dir=AQUÍ VA LA RUTA”

7- Ejecutar el Context Modeling Framework

Para ejecutar el context Modeling Framework, se le da click derecho “run”.

luego ya pueden utilizarlo entrando a la siguiente url:

http://localhost:8080/contextModeling/faces/login.xhtml