Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas.

Título: Desarrollo de la Capa de Vocabulario del Servicio de Terminologías Comunes

Autor:

Randy Ruiz Chantez

Tutores:

Ing. Karel Fernández Cedeño

Ing. Rubén Miranda Silva

La Habana, junio de 2011

"Año 53 de la Revolución"

Datos de Contacto

Ing. Karel Fernández Cedeño:

Graduado en Ingeniería en Ciencias Informáticas en el 2008. Profesor Instructor, ha impartido las Asignaturas Técnicas de Programación, Ingeniería de Software y Práctica Profesional. Actualmente se desempeña como Analista Principal del Centro de Informática Médica.

Correo electrónico: kfernandez@uci.cu

Ing. Rubén Miranda Silva:

Profesor en adiestramiento. Graduado de Ingeniero en Ciencias Informáticas, egresado de la UCI en el 2010. Se desempeña actualmente como Arquitecto Principal del departamento de Tecnologías, Integración y Estándares del Centro de Informática Médica (CESIM), en la UCI.

Correo electrónico: rmsilva@uci.cu

Dedicatoria

A mi mamá, papá, hermana y hermano, por su confianza y a la vez ser ejemplos de sacrificio y consagración.

A mi familia por su apoyo.

A los que de una forma u otra me han ayudado a llegar aquí.

Gracias a todos por estar siempre ahí.

Agradecimientos

A mis padres por brindarme siempre su apoyo y confianza ,por ayudarme a levantar cuando tropezaba, por darme todo lo que pudieron e intentar dar más, no los defraudaré.

A la Universidad de las Ciencias Informáticas por permitirme cursar estudios en ella.

A mis tutores y profesores que me han ayudado en la formación tanto personal como profesional.

A todos los que de una manera u otra hicieron posible el desarrollo de este trabajo.

A mis compañeros con los que compartí estudios, momentos y diversión.

Resumen

El desarrollo de este trabajo surge por la necesidad de establecer una comunicación más eficiente y segura entre diferentes sistemas clínicos. Tiene como objetivo el desarrollo de una Capa de Vocabulario según la especificación para un Servicio de Terminología Común (CTS) del estándar HL7(Nivel 7 de Salud), buscando así mejorar el intercambio de información terminológica entre los sistemas desarrollados para las distintas áreas de atención sanitaria.

Se analizaron sistemas terminológicos existentes internacionalmente, obteniéndose experiencias de los mismos que sirvieron de guía y se identificaron funcionalidades que por su utilidad se pudieran incluir en la implementación. Se realizaron modelos y diagramas que facilitaran el entendimiento de la estructura y los requisitos de desarrollo. Además, se seleccionaron diferentes tecnologías, metodologías y herramientas a emplear en la implementación y desarrollo del trabajo: como son el lenguaje de modelado UML, Enterprise Architect como herramienta de modelado, PostgreSQL como sistema gestor de base datos, Eclipse como entorno de desarrollo integrado y como lenguaje de implementación de negocio Java.

Con la implementación de la Capa de Vocabulario, se facilita el intercambio de información terminológica entre los sistemas desarrollados en el CESIM y sistemas de terceros. Además, se provee de una mejor descripción de las terminologías médicas utilizadas en los sistemas de información lo cual permite un mayor grado de profundidad y entendimiento de los mensajes compartidos. Esto sin dudas se traduce en calidad de vida y bienestar social.

Palabras claves: Terminologías, CTS, HL7, comunicación, sistemas.

Índice

INTRODUCCIÓN

La práctica médica ha evolucionado sin cesar desde la antigüedad. Los avances tecnológicos de los últimos 100 años incrementaron su complejidad y las organizaciones de salud han adoptado las nuevas tecnologías a medida que fueron surgiendo.

La mayoría de los sistemas de salud cuentan con múltiples modelos y métodos de llevar los registros médicos en papel y otros formatos, además de múltiples archivos físicos que conforman los registros médicos. Es por esto que la información asistencial se encuentra fragmentada y muchas veces duplicada cómo hace 150 años. Dos siglos atrás, en 1863, Florence Nightingale, una enfermera que incentivó el inicio de la epidemiología y la estadística, ya expresaba su preocupación por la existencia de datos y la necesidad de optimizar la calidad de la información para la toma de decisiones estratégicas y mejorar la calidad de atención de los pacientes.

Sin embargo, desde hace tan sólo dos décadas se habla de los sistemas de información hospitalario (HIS). Gradualmente el interés se ha ido focalizando en el cuidado del paciente, integrando toda la información que se origina en los múltiples registros dentro de una o varias organizaciones de salud donde fue atendido. Estos aspectos lograron verse en algunos sistemas a finales de los '60 y principios de los 70', pero fallaron debido al insuficiente hardware y software.

En las organizaciones clínicas, los sistemas de información se desarrollaron paulatinamente, pero mayormente no fueron concebidos pensando en integrar la información del paciente. Estos se desarrollaron como solución a problemas puntuales o mejoras a determinados procesos como iniciativas propias, trayendo la dispersión de la información y diferencias en cuanto a estructura y representación. En la actualidad factores como la dispersión geográfica, las nuevas relaciones de interconsultas, la expansión de redes informáticas y la necesidad de mayor control de procesos anuncian la necesidad de reorientar la informatización del sector de la salud hacia una visión global de la gestión clínica.

La Gestión Clínica es la utilización adecuada de los recursos para la mejor atención de los pacientes. La responsabilidad de las decisiones es tomada por el equipo de salud. Su implementación implica una descentralización de la organización y un impulso hacia una nueva forma de trabajo, enfocada hacia la gestión de procesos, la mejora continua y la autoevaluación sistemática. (1)

Esta gestión tiene como requerimiento indispensable el intercambio de datos entre aplicaciones de salud, con el fin de contar con información clínica de alta calidad, contextual y capaz de ser ajustada por riesgos, lo cual implica nuevos desafíos para el diseño y desarrollo de los sistemas de información. (2)

El intercambio de datos es muy importante en escenarios de la salud, ya que la asignación incorrecta de diagnósticos o medicamentos a un paciente durante la atención, puede tener consecuencias en los resultados del proceso y en la salud del individuo. Por ejemplo, la asignación errónea de datos acerca de la positividad de serologías para el virus del VIH a un individuo o alergias medicamentosas que pueden contraindicar equivocadamente un tratamiento.

Actualmente la información sanitaria de un paciente está dispersa entre diferentes organizaciones, principalmente en atención primaria y especializada, además de hospitales y servicios de salud regionales. Como consecuencia, los datos están distribuidos y son heterogéneos, creando una gran diferencia entre el valor potencial y el valor real de la información presente en los sistemas de Historia Clínica Electrónica (HCE). Eliminar esta diferencia haciendo uso de los datos presentes en estos sistemas, permite mejorar significativamente el cuidado, seguridad del paciente y la eficiencia en actividades e investigaciones clínicas.

La HCE es el repositorio de los datos que surgen de la interacción del paciente con los profesionales de la salud. Es un documento médico legal que permite un entendimiento global y longitudinal de la situación del paciente; por eso es de incontable ayuda a la hora de realizar un diagnóstico o instaurar una terapia.

Son varias las ventajas que brindan una HCE: permite consultar la información médica del paciente, dejar registros de las evoluciones diarias, solicitar prácticas e indicaciones médicas, acceder a los resultados e incluso, brinda un sistema de soporte en la toma de decisiones, todo esto en el lugar y momento de la atención. También permite un almacenamiento centralizado y seguro, con consultas distribuidas y además, mediante una codificación previa, brinda capacidad de análisis epidemiológicos, convirtiéndose en una herramienta muy útil para realizar una adecuada gestión de salud. (3)

Sin embargo, a lo largo de los años, han proliferado sistemas diferentes no sólo entre centros de países distintos, sino incluso, entre diferentes centros de una misma zona y diferentes departamentos de un mismo centro. La existencia de sistemas diseñados sin coordinación, implica diferentes modelos para la representación de la información y la enorme amplitud del dominio médico ha dado lugar a una gran cantidad de terminologías diferentes.

Conseguir un historial clínico electrónico que incluya toda la información sanitaria sobre un paciente, con independencia del lugar donde haya sido generada o dónde se encuentre almacenada y mantenida, requiere la interoperabilidad semántica de los sistemas de HCE. La interoperabilidad semántica es la capacidad de los sistemas informáticos de comunicar, incorporar y usar información generada por sistemas externos. (4)

La solución a la interoperabilidad semántica es utilizar una terminología de referencia común con la que todos los sistemas puedan comunicarse, manteniendo al mismo tiempo sus propias terminologías locales intactas y generando mapeos entre ellas.

Para que diferentes sistemas puedan integrar la información de un paciente, se necesita que la misma se transfiera de un sistema a otro. Esta transferencia se realiza a través de interfaces adaptadas y personalizadas, pero la cantidad de interfaces de los sistemas a unir aumenta exponencialmente y resultaría inmanejable.

Un enfoque para resolver este problema de interfaces múltiples es HL7 (Nivel 7 de Salud), un sistema que desarrolla mensajes estandarizados (sintaxis) que viajan a través de una única interfaz. Estos mensajes utilizan a la vez datos estandarizados (semántica), como la identificación del paciente, los datos del laboratorio y valores definidos como posibles para esos campos, basados todos en un vocabulario estándar y controlado. (5)

Una comunicación efectiva requiere que el emisor y el receptor de información compartan un marco de referencia común que permita la interacción. Los estándares proveen ese marco común, promoviendo una uniformidad en la denominación de los componentes del sistema de salud, ya sean objetos, diagnósticos, personas e intervenciones.

La interoperabilidad requiere la creación, aceptación e implementación de estándares para asegurar que los datos, en una parte del sistema de salud, estén disponibles y tengan significado a través de la variedad de escenarios clínicos.

Por tal motivo, el departamento de Tecnología, Integración y Estándares (TIE) del Centro de Informática Médica de la Universidad de las Ciencias Informáticas (UCI), tiene como objetivos estratégicos lograr la integración con otras aplicaciones desarrolladas y estandarizar el uso de las terminologías médicas

utilizadas, y para ello pretende hacer uso de estándares internacionales, como es el Nivel 7 de Salud(HL7).

El estándar HL7, desarrolló un Servicio de Terminología Común (CTS),el cual define una serie de Interfaces de Aplicación (API) y un modelo de datos que permiten integrar y acceder de manera uniforme a diferentes terminologías. El CTS describe la funcionalidad básica necesaria para el acceso al contenido terminológico desde sistemas de información y ofrece un modelo conceptual y de datos para la gestión y almacenamiento de terminologías. (6)

Este servicio posibilitará que la información a que se acceda tenga un significado claro, no ambiguo. Para esto, es necesario describirla mediante anotaciones terminológicas accesibles desde ambos sistemas, el que generó la información y el que la recibe. El enlace terminológico entre los datos y las terminologías clínicas, es un requisito previo para conseguir interoperabilidad semántica y así evitar que en el intercambio de datos exista cualquier posibilidad de error o mala interpretación, asegurando además que la información conserve, al ser recibida, el significado original asignado cuando fue generada.

En la actualidad, los sistemas de información clínica pueden no estar desarrollados sobre una misma arquitectura, tecnología, siguiendo una estructura de datos diferentes, lo que dificulta el entendimiento entre ellos. Otro problema, es la pérdida de información causada por el modelo de datos, ya que en los sistemas gestores de bases de datos se implementan definiciones de modelos de datos individualizados. Igualmente, las ambigüedades del lenguaje médico dificultan la recuperación de información.

A partir del análisis de la situación problemática, se define el siguiente **problema a resolver:** ¿Cómo viabilizar la interpretación de mensajes codificados entre los sistemas desarrollados en el Centro de Informática Médica y sistemas de terceros?

El problema científico se enmarca en el **objeto de estudio:** el proceso de intercambio de información entre los sistemas desarrollados en el Centro de Informática Médica, delimitando el **campo de acción:** en el proceso de intercambio de mensajes codificados entre los sistemas desarrollados en el Centro.

Se define como el **objetivo general:** Desarrollar la Capa de Vocabulario que permita interpretar y traducir diferentes versiones de sistemas de códigos.

Para dar cumplimiento a los objetivos planteados en la investigación, se trazó el siguiente plan de **Tareas de la Investigación**:

- Valorar las tendencias actuales en el mundo de los Servicios de Terminologías Médicas.
- Analizar la arquitectura definida en el departamento Tecnología, Integración y Estándares para la implementación de la Capa de Vocabulario.
- Elaborar los principales artefactos de los flujos de trabajo propuestos por la metodología definida.
- Implementar las funcionalidades correspondientes a la Capa de Vocabulario: Vocabulary Browsing y Vocabulary Runtime.
- Integrar la Capa de Vocabulario desarrollada con la Capa de Mensajes, la de Mapeo y con el Sistema de Administración.
- Realizar pruebas de caja negra a la Capa de Vocabulario.

Con el propósito de desarrollar las tareas planteadas para el desarrollo de la investigación se utilizaron los **métodos de investigación** siguientes:

- De los **métodos teóricos** se utilizó el **analítico-sintético** para analizar las tecnologías, herramientas y metodologías a utilizar en el desarrollo del trabajo así como examinar la bibliografía consultada que ayudaría a su confección. El **histórico-lógico** para analizar las tecnologías y sistemas desarrollados internacionalmente.
- De los **métodos empíricos** se utilizó la **entrevista** para analizar los problemas de interoperabilidad entre los sistemas sanitarios y lograr una mejor comprensión de la comunicación entre estos.

Como **Resultados** se espera lograr que todos los sistemas desarrollados en el CESIM puedan intercambiar mensajes sin pérdida de información por ambigüedad ó interpretaciones incorrectas. Además, que las instituciones de salud que utilicen el servicio dispongan de un sistema integrado capaz de comunicarse entre sus diferentes componentes y con otras aplicaciones.

Al desarrollar la Capa de Vocabulario se podrán obtener los siguientes **Beneficios**:

- Se logra una solución que contribuya a una comunicación más efectiva entre las aplicaciones sanitarias. Lo que posibilitará una mejor descripción de las terminologías clínicas y evitará con esto la pérdida de datos terminológicos, todo ello será de gran valor para el Sistema Nacional de Salud.

- Se agiliza el flujo de información posibilitando una rápida respuesta a las solicitudes hechas al sistema. Igualmente, se obtendría información detallada de las terminologías clínicas existentes en el servidor.
- Con su integración con las de Mensajería y Mapeo, se garantiza una estandarización en la representación y codificación de los modelos de datos en las comunicaciones entre sistemas. Además, permite un aumento en la calidad de la gestión de la información y posibilita a los especialistas dar un diagnóstico clínico más exacto, lo que será de gran importancia para la atención y cuidado del paciente.

Este trabajo está estructurado por 4 capítulos de la siguiente forma:

Capítulo 1: Fundamentación teórica. En este capítulo se realiza un estudio del estado del arte a nivel nacional e internacional. Se establece un marco conceptual en correspondencia con la información que será manipulada. Además de una breve explicación del ambiente de desarrollo empleado para la realización del sistema.

Capítulo 2: Características del sistema. Este capítulo se centra en la especificación de los requerimientos de software. Se muestran los principales procesos a través de casos de uso del sistema, los actores del sistema que intervienen, sus relaciones y una descripción resumida de cada uno de ellos.

Capítulo 3: Análisis y Diseño. En este capítulo se hace un análisis del ambiente a desarrollar. Se estructuran y obtienen los modelos de análisis y de diseño en base a los requisitos obtenidos. Se hace referencia al modelo de implementación, diagrama de despliegue, la arquitectura recoge los diagramas de componentes y su descripción.

Capítulo 4: Implementación y Prueba. Este capítulo trata los aspectos relacionados con la implementación de la solución propuesta y se realiza el diagrama de componentes. Se describe las acciones para garantizar la seguridad. Además se llevará a cabo pruebas de caja negra para validar la solución e implementación de las funcionalidades previstas. Se presenta la propuesta del sistema para obtener los resultados con la calidad requerida.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Este capítulo, tiene el objetivo de abordar los diferentes aspectos teóricos que son la base conceptual para el desarrollo del trabajo. Además, se abordarán antecedentes y tendencias actuales de los Servicios de Terminologías Comunes, así como un análisis de las tecnologías, metodologías y herramientas de software definidas por el Departamento de Tecnología, Integración y Estándares con las que se llevará a cabo el proceso.

1.1 Estado del Arte

1.1.1 Antecedentes

El primer estándar HL7 (Nivel 7 de Salud) apareció en 1990 y era un modelo de información no explícito. Presentaba una opcionalidad excesiva, causante de muchos problemas, carecía de un lenguaje natural y estaba limitado a una única forma de programación. Además, la integración era compleja y se necesitaba de 2 a 4 meses para implementar un interfaz de comunicación. Entre los beneficios de HL7 se encuentra que es más barato de mantener a medio y largo plazo, y que facilita la integración de sistemas heterogéneos. (7)

Este estándar ha estado en constante evolución y desarrollo desde su surgimiento, la asociación que lo desarrolla está acreditada por el ANSI (American National Standard Institute) desde 1994. Su principal logro ha sido la creación de un estándar que se aplica básicamente en tres líneas de acción, como se muestra en la figura1:

Figura 1.1. Estándares aplicables a la práctica clínica.

- En lo **conceptual** el Reference Information Model (HL7 RIM) provee un marco para describir los datos clínicos y el contexto circundante: Quién, qué, cuándo, dónde y cómo.

- Para la **documentación** el Clinical Document Architecture (HL7 CDA) éste indica que tipo de información está incluida en un documento y dónde puede ser hallada.
- Para las **aplicaciones** el Clinical Context Object Workgroups (HL7 CCOW) éste determina el modo en que las reglas de negocio son implementadas y su interacción con los sistemas de software.(8)

El estándar HL7 se basa en el modelo de referencia de información y es general en su estructura. La representación de la información en este modelo depende de la disponibilidad de los recursos terminológicos, siempre que es posible hace referencia a recursos de un estándar externo, en vez de crear nuevos recursos en la misma norma.

Los contenidos y la estructura de estos recursos están expuestos a considerables cambios. HL7 puede identificar las características comunes que poseerá cada término para que pueda ser utilizado. La versión 3 de HL7 ha desarrollado un Modelo de Información de Referencia (RIM) que es la base del intercambio de información del nuevo estándar. La representación de la información dentro de este modelo depende de la disponibilidad de los recursos terminológicos que se pueden utilizar para rellenar las propiedades del modelo con contenido semántico.

El estándar puede identificar un conjunto mínimo de características que cualquier recurso terminológico debe poseer si se va a utilizar en un entorno de mensajería HL7. Un enfoque para esta tarea sería la de especificar una estructura de datos común que todos los recursos terminológicos deben poseer.

El Servicio de Terminología Común (CTS) fue desarrollado como alternativa para una estructura de dato común. HL7 lo ha elegido para identificar las características funcionales comunes que una terminología debe ser capaz de proporcionar. A modo de ejemplo, un servicio de terminología tendrá que determinar si un código de un concepto determinado es válido. Actualmente está publicada la especificación CTS HL7 y una implementación de referencia de la especificación.

La especificación de HL7 describe un CTS Application Programming Interface (API) de llamada que tiene como objeto describir la funcionalidad básica que necesitará la implementación del software, para consultar y acceder al contenido terminológico. Se presenta como un API con el objetivo de dar libertad a los desarrolladores sobre como almacenar la información terminológica. De este modo no se esfuerza a

migrar datos o reescribir código si se pretende, por ejemplo, integrar CTS en un sistema ya implementado.(9)

El CTS define una serie de interfaces de aplicaciones y un modelo de datos, que permite integrar y acceder de manera uniforme a diferentes terminologías. Este se divide en capas, la Capa de Mensajes que es la superior, la cual se comunica con un software de mensajería en términos de dominios de vocabulario y la Capa de Vocabulario que es la más baja, la cual se comunica con el software de almacenamiento de terminología utilizando un modelo conceptual propio, como se muestra en la figura 1.2.

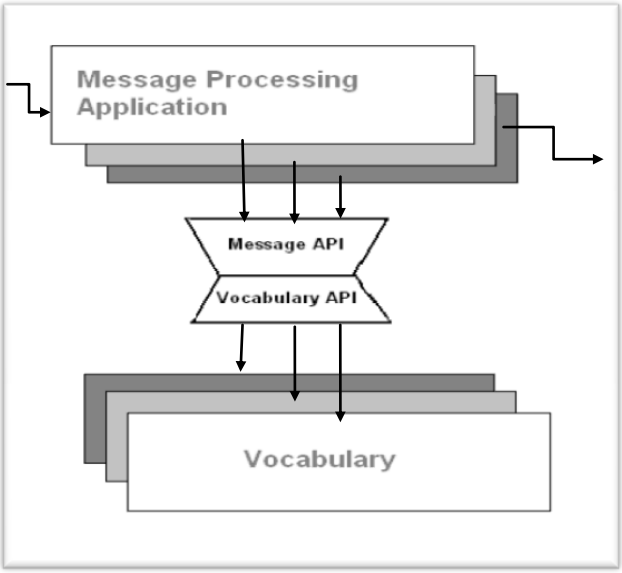


Figura 1.2. Capa de Mensajería y de Vocabulario.

La API de vocabulario está diseñada para ser genérica y fácilmente utilizable en diferentes entornos, no sólo desde la capa de mensajes. Este vocabulario trabaja sobre el supuesto de que existe una base de datos que contiene al menos una terminología o un sistema de codificación. (10)

1.1.2 Sistemas existentes en la actualidad

En la actualidad, diversas herramientas son construidas en las instituciones con la finalidad de ayudar y hacer más eficiente el trabajo diario. A continuación se ejemplifican algunas de las herramientas que fueron estudiadas por su relación con el dominio de la investigación:

SvmSoft

Es la primera sección de términos médicos, con la capacidad de añadir y editar sus propias abreviaturas, acrónimos, términos y definiciones. Es una gran fuente de términos médicos y definiciones de gran utilidad para las actividades diarias de médicos, asistentes, técnicos y todas las personas vinculadas con la información.

Está caracterizado por:

- Una lista completa de más de 16.000 términos y definiciones médicas, así como más de 8.000 acrónimos y abreviaturas.
- Búsqueda por siglas o términos.
- Etiqueta los términos de favoritos.
- Comparte sus términos con los demás por correo electrónico.
- Agrega y edita sus términos y definiciones. (11)

Diccionario médico XTerm (XTerm Medical Dictionary)

Es una aplicación útil y fácil de usar, que incluye un motor de búsqueda con comodines o faltas de ortografía. También es posible buscar las definiciones que contienen una o dos palabras que desee unidos por un operador lógico. La base de datos de términos médicos es actualizada semanalmente en el sitio web del editor y el software incluye entre sus características, una función para descargar y agregar estas actualizaciones fácilmente. En la base de datos de términos médicos se recogen las definiciones, abreviaturas, siglas, símbolos, mnemónicos de condiciones médicas, las estructuras anatómicas, procedimientos, mecanismos de drogas, productos químicos, las asociaciones médicas, unidades de medida, entre otras.

Algunas de las características de esta aplicación:

- Examinar las cabeceras.
- Comodín de la búsqueda.
- Búsqueda en las definiciones.

- Permite actualizar sus terminologías.(12)

SerVoMed

Es un servidor de vocabulario médico que permite establecer relaciones semánticas entre términos médicos y correspondencias entre varios vocabularios biomédicos: “Unified Medical Language System” (UMLS), “Gene Ontology” (GO) y “Human Genome Nomenclature Consortium” (HGNC). Resulta útil para desarrolladores de aplicaciones médicas que necesiten trabajar con un vocabulario médico controlado, ya que define una posible interfaz entre la aplicación y un servidor de vocabulario. (13)

SNOMED CT (Systematized Nomenclature of Medicine--Clinical Terms)

Es una colección procesable sistemáticamente organizada de la computadora de terminología médica, cubre la mayoría de las áreas de la información clínica, ejemplo: enfermedades, resultados, procedimientos, microorganismos, productos farmacéuticos, entre otros. Permite una manera constante de poner en un índice, de almacenar, de recuperar y de agregar datos clínicos a través de especialidades y de sitios del cuidado. También las ayudas organizan el contenido de expedientes médicos, reduciendo la variabilidad en los datos, de esta manera se captura, se codifica y se utiliza para el cuidado clínico de pacientes y de la investigación.

Características:

La ventaja de la información de grabación en una terminología estándar tal como SNOMED CT se liga a las ventajas del expediente electrónico del cuidado y a las ventajas de la información clínica de la grabación en una forma estructurada.

- Proporciona una terminología constante a través de todos los dominios del cuidado.
- Permite la grabación exacta de la información clínica y tiene una estructura inherente.
- Es un estándar internacional que se convierte.

Componentes:

- Conceptos: Unidad básica del significado señalada por un código numérico único, un nombre único (nombre completamente especificado), y descripciones, incluyendo un término preferido y unos o más sinónimos.

- Descripciones: Términos o nombres (sinónimos) asignados a un concepto.
- Jerarquías: 19 jerarquías de alto nivel; cada uno tiene secundario-jerarquías.
- Relaciones: Ligue los conceptos dentro de una jerarquía o a través de jerarquías.
- Subconjuntos. (14)

LexGrid

El modelo LexGrid es la propuesta de la Clínica Mayo para el almacenamiento de nivel de vocabularios controlados y ontologías. El modelo LexGrid define cómo los vocabularios deben ser formateados y representados mediante programación, y está destinado a ser lo suficientemente flexible como para representar con precisión una amplia variedad de vocabularios y otros recursos léxicos. El modelo también define los diferentes mecanismos de almacenamiento de servidor y un formato XML.(15)

El modelo LexGrid permite caracterizar las partes fundamentales de las terminologías de manera que todas puedan descomponerse y conceptualizarse para ser traducidas. Una vez que la información de vocabularios se encuentre representada bajo el mismo estándar, es posible construir repositorios compartidos para contenido terminológico e interfaces comunes y herramientas para acceder y gestionar el contenido. (16)

A partir de los sistemas estudiados, las funcionalidades que se pudieran implementar en el sistema propuesto, en el presente trabajo de diploma son: del **SvmSoft** la búsqueda por siglas o términos, y las correspondencias entre varios vocabularios biomédicos. También, del **Diccionario médico Xterm**, la búsqueda con deletreo incorrecto y que en su base de datos de términos médicos se recogen las definiciones, abreviaturas, siglas, símbolos, procedimientos, mecanismos de drogas, productos químicos, las asociaciones médicas y unidades de medida.

Algunas de estas aplicaciones contienen funcionalidades que no aportan al cumplimiento de los objetivos planteados, como es el caso de la aplicación **SvmSoft** que da permisos para agregar y editar sus términos y definiciones. También el **Diccionario médico XTerm**, el cual es desarrollado como una aplicación web, incluyendo una función para descargar y agregar actualizaciones. Otros como **Términos Clínicos SNOMED** y **LexGrid** se parecen en su estructura a lo que se pretende desarrollar, por lo que se puede utilizar como referencia para el desarrollo del trabajo de diploma.

En el territorio nacional no se conoce del desarrollo de servicios terminológicos, lo que implicará diseñar un sistema que responda a las necesidades de las terminologías médicas para obtener una clasificación integradora de los términos empleados por los profesionales de la salud. Lo que ayudará en la toma de decisiones clínicas en consultorios médicos y otras instituciones hospitalarias. Para concebir de manera integrada los datos generados en los distintos niveles de atención en el Sistema Nacional de Salud (SNS)se requiere de un proceso de captura, registro, procesamiento, validación y análisis de la información inestimable, lo cual incrementa su consistencia, veracidad y oportunidad.

Lo anterior redunda en el mejoramiento de la actividad administrativa, asistencial, docente y de investigación. Ello obliga a que cada subsistema hospitalario cubano de control de registros médicos, no se diseñe en forma aislada y con criterio propio, sino de manera normalizada con el objetivo de enlazar posteriormente con facilidad cada uno de ellos en una red nacional.

Después de haber analizado las soluciones existentes en el ámbito nacional e internacional, se puede observar que actualmente en Cuba los sistemas de información de salud no cuentan con un Servicio de Terminología Común que permita dar solución a los problemas planteados. Pero algunos de los sistemas implementados a nivel internacional cumplen o tienen cierta similitud con el servicio que se necesita, pero los tipos de licencia y altos costos impulsan el desarrollo de un CTS para lograr una calidad superior en la integración y desarrollo de los sistemas informáticos para la atención sanitaria en el país.

1.2 Tecnologías y Tendencias

1.2.1 Tecnologías

La informática médica, define el apoyo que brindan las tecnologías de la información y las comunicaciones (TICs) al sector de la salud, arribando al concepto de e-Salud. Los servicios de e-Salud incluyen aplicaciones asistenciales (Tele-consulta, Tele-diagnóstico y Tele-monitorización), las relacionadas con la administración y gestión de pacientes (continuada asistencial, integración de niveles asistenciales) y las de información y formación a distancia para usuarios y profesionales.

También mediante los arquetipos se pueden lograr facilidades en el intercambio y utilización de información clínica, como es hacer consultas de la historia clínica del paciente, desde cualquier sección de cualquier hospital que la requiera. Para esto se dispone actualmente de la plataforma ArchMS, que

permite la edición, anotación, consulta y transformación de arquetipos clínicos. Los arquetipos son un mecanismo de representación formal de conceptos clínicos que son automáticamente procesables por un sistema informático. (17)

Tendencias

Las tendencias de los sistemas de información en la atención sanitaria, será la compilación de los diferentes lenguajes documentales que permitan al profesional, no sólo navegar a través de una historia clínica para consultar datos clínicos del paciente, sino también acceder a bases de datos, bibliografías y herramientas de ayuda para la toma de decisiones.

1.3 Metodologías y estándares de calidad

El software puede ser considerado como producto o como servicio, lo que ha motivado a nivel mundial generar un conjunto de modelos para estimar su calidad. Estos modelos responden a las necesidades de garantizar productos de calidad. Para lograr un buen desarrollo en un software, se tiene que tener en cuenta la utilización de las metodologías y estándares de calidad.

En el mundo existen diversas metodologías y estándares que permiten evaluar y regir el desarrollo de un producto, entre las que se encuentran: CMMI y RUP.

1.3.1 El Modelo de Madurez y Capacidad Integrado (CMMI)

El Modelo de Madurez y Capacidad Integrado (CMMI) se utiliza para mejorar la calidad del desarrollo y mantenimiento de software, basado en las mejoras de los procesos de la organización. El modelo CMMI aplica conceptos de gerencia de procesos y mejora de calidad al desarrollo y mantenimiento de software, y describe los estados a través de los cuales las organizaciones de software evolucionan a medida en que definen, implementan, miden, controlan y mejoran sus procesos de software. Se basa en el concepto de la evolución de la madurez de los procesos.

El modelo CMMI aplica conceptos de gerencia de procesos y mejora de calidad al desarrollo y mantenimiento de software, y describe los estados a través de los cuales las organizaciones de software evolucionan a medida en que definen, implementan, miden, controlan y mejoran los procesos de software.

CMMI define 5 niveles mediante los cuales se describen los distintos grados de madurez de una organización.

Nivel 1: Inicial. Estado inicial donde el desarrollo se basa en la heroicidad y responsabilidad de los individuos.

Nivel 2: Administrado. Se normalizan las buenas prácticas en el desarrollo de proyectos.

Nivel 3: Definido. La organización entera participa en el proceso eficiente de proyecto de software.

Nivel 4: Administrado cuantitativamente. Se puede seguir con indicadores numéricos la evolución de los proyectos, las estadísticas son almacenadas para aprovechar su aportación en siguientes proyectos y los proyectos se pueden pedir cuantitativamente.

Nivel 5: Optimizado. En base a criterios cuantitativos se pueden determinar las desviaciones más comunes y optimizar procesos. (18)

1.3.2 Proceso Unificado de Desarrollo (RUP)

El RUP puede verse como una metodología adaptable, es decir, se puede modificar para adaptarlo al sistema concreto que se va a desarrollar en cada momento. Por otra parte, es una técnica para elaborar modelos que se adapta especialmente a UML. Su objetivo es producir un software de calidad. Por definición, PU utiliza buenas prácticas de desarrollo, siendo adaptable a un amplio rango de aplicaciones y sistemas. Este proceso no sólo considera aspectos de desarrollo de un sistema, sino también los de gestión del mismo.

Entre sus características se encuentra que soporta técnicas orientadas a objeto, por lo que se basa en los conceptos de clase y objeto y las relaciones entre ellos, usando UML como notación común. Es una metodología que sigue un proceso iterativo e incremental. Propone una descomposición incremental del problema a través de refinamientos sucesivos y una producción incremental de la solución, a través de la realización de varios ciclos. Esta filosofía es lógica cuando se aplica a sistemas grandes ya que “no se puede abarcar todo a la vez”. El PU tiene 4 fases o incrementos y en cada uno se consideran distintos flujos de trabajo (workflow) o modelos que suponen mayor o menor número de horas de trabajo dependiendo de la fase incremental. (19)

Luego de la investigación realizada se llegó a la conclusión de utilizar esta metodología, ya que brinda ventajas importantes al proporcionarle énfasis a los requerimientos y al diseño, basándose en las mejores prácticas que se han probado en el campo de la informática. Con esta decisión se espera una mejor calidad del software a crear, y contar con la menor cantidad de vulnerabilidades al final de la construcción.

1.3.3 Lenguaje de Modelado (UML 2.1)

El desarrollo de sistemas es una actividad humana. Sin un sistema de notación fácil de comprender, el proceso de desarrollo tiene una gran cantidad de errores. El UML es la notación de diseño que los analistas, desarrolladores y clientes aceptan como pauta debido a la necesidad de diseños sólidos.

El UML permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas. UML proporciona el proceso de diseño de tal forma que los analistas, clientes, desarrolladores y otras personas involucradas en el desarrollo del sistema lo comprendan y convengan con él.

Está constituido por un conjunto de diagramas y es necesario contar con todos esos diagramas dado que cada uno se dirige a cada tipo de persona implicada en el sistema. Un modelo UML indica que es lo que supuestamente hará el sistema, más no como lo hará. (20)

Se empleará para el modelado de la aplicación, ya que proporciona a los desarrolladores un mejor entendimiento del proyecto, disminuye el tiempo invertido en desarrollo de la arquitectura y agiliza la detección y corrección de errores. Teniendo en cuenta la arquitectura definida por el departamento de Tecnología, Integración y Estándares, se utilizará el Enterprise Architect como herramienta para el modelado.

1.4 Herramientas y tecnologías de desarrollo

1.4.1 Herramienta de Modelado (Enterprise Architect 7.0)

Enterprise Architect combina el poder de la última especificación UML 2.1 con alto rendimiento e interfaz intuitiva, para traer modelado avanzado al escritorio y para el equipo completo de desarrollo e implementación.

Es una herramienta comprensible de diseño y análisis UML, que cubre el desarrollo de software desde el paso de los requerimientos, a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Está diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad.

Enterprise Architect provee trazabilidad completa desde el análisis de requerimientos hasta los artefactos de análisis y diseño, a través de la implementación y el despliegue. Soporta generación e ingeniería inversa de código fuente para muchos lenguajes populares, incluyendo C++, C#, Java, Delphi, VB.Net, Visual Basic y PHP. Presenta una documentación de alta calidad, es intuitivo y simple de usar además de tener un bajo costo de licencias. (21)

Para definir, crear y mantener la base de datos, se decidió utilizar el sistema gestor de base de datos PostgreSQL 8.4.

1.4.2 Sistema Gestor de Base de Datos (PostgreSQL 8.4)

PostgreSQL 8.4 es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. Es una derivación libre (Open Source) del proyecto POSTGRES y utiliza el lenguaje SQL92/SQL99. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...) y cadenas de bits. Incorpora una estructura de datos array e incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes. Permite la declaración de funciones propias, así como la definición de disparadores. Soporta el uso de índices, reglas y vistas además de permitir la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

Entre sus ventajas se encuentra la instalación ilimitada, permitiendo que nadie pueda demandarlo por violar acuerdos de licencia, puesto que no hay costo asociado a la licencia del software. Es multiplataforma, ya que está disponible en casi cualquier Unix (34 plataformas en la última versión estable), y una versión nativa de Windows está actualmente en estado de pruebas. Permite ahorros

considerables en costos de operación ya que ha sido diseñado y creado para tener un mantenimiento y ajuste mucho menor que los productos de los proveedores comerciales, conservando todas las características, estabilidad y rendimiento. (22)

1.4.3 Máquina Virtual de Java (JVM)

La JVM simula un "Chip Java", el cuál permite ejecutar el conjunto de instrucciones de la máquina Java. Fue diseñada para proveer ejecutables en un tipo de plataforma ejecutable estándar y su arquitectura se basa en el concepto de una implementación que no es específica de una máquina. La JVM es una entidad autónoma e única que ejecuta los archivos de clases y se separa en cinco unidades de funcionalidad distintas, las que se dedican a la tarea de ejecutar los archivos de clases, estas son: Registros, Pila, "Montículo" de Colección de "desperdicios", Área de almacenamiento de métodos y Conjunto de Instrucciones de la JVM. (23)

Con el fin de lograr un mejor funcionamiento y seguridad en el sistema se precisó por el departamento de Tecnología, Integración y Estándares utilizar Java como plataforma para el desarrollo de aplicaciones.

1.4.4 Lenguaje de desarrollo (Java)

Java es un lenguaje de desarrollo de propósito general, y como tal es válido para realizar todo tipo de aplicaciones profesionales. Una de las características más importantes es que los programas “ejecutables”, creados por el compilador de Java, son independientes de la arquitectura. Se ejecutan indistintamente en una gran variedad de equipos con diferentes microprocesadores y sistemas operativos.

El lenguaje Java es robusto. Las aplicaciones creadas en este lenguaje tienen pocos errores, principalmente porque la gestión de memoria y punteros es realizada por el propio lenguaje y no por el programador. Además, el lenguaje contiene estructuras para la detección de excepciones (errores de ejecución previstos) y permite obligar al programador a escribir código fiable mediante la declaración de excepciones posibles para una determinada clase reutilizable.

Entre sus características se encuentra que es orientado a objetos, funciona perfectamente en red y aprovecha características de la mayoría de los lenguajes modernos evitando sus inconvenientes, en particular los del C++. Igualmente, tiene una gran funcionalidad gracias a sus librerías y genera aplicaciones con pocos errores posibles. (24)

Java es muy similar al C# y su sintaxis es amigable, además de que aporta características importantes que simplifican grandemente las labores de implementación. Se decidió utilizar el IDE de desarrollo Eclipse.

1.4.5 Entorno de Desarrollo Integrado (Eclipse Galileo)

Eclipse es un Entorno de Desarrollo Integrado(IDE) de código abierto multiplataforma, para desarrollar lo que el proyecto llama “Aplicaciones de Cliente Enriquecido”, opuesto a las aplicaciones “Cliente-liviano” basadas en navegadores. Provee al programador, frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software y aplicaciones web.

Entre sus características se encuentran:

- Editor de texto
- Resaltado de sintaxis
- El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadata en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.
- Compilación en tiempo real.
- Pruebas unitarias con JUnit.
- Control de versiones con CVS(Concurrent Versions System).
- Integración con Ant(Apache Ant).
- Asistentes: para creación de proyectos, clases, tests.
- Refactorización. (25)

Para desarrollar ciertas aplicaciones con una arquitectura cliente-servidor, se hace necesario hacer uso de los servicios web(Web Services). Para ello, se necesita dotar al servidor de los servicios necesarios para

poder recibir dichas peticiones así como publicar los servicios web desarrollados por lo que se utilizará Axis2 1.4.1 como motor de servicios web.

1.4.6 Apache Axis2

La nueva arquitectura en la que se basa en Axis2 es más flexible, eficiente y configurable. Apache Axis2 no sólo es compatible con SOAP 1.1 y SOAP 1.2, también tiene soporte integrado para el estilo REST muy popular en los servicios Web. Apache Axis2 es más modular y más orientado a XML. Está cuidadosamente diseñado para soportar la fácil adición de plug-in de "módulos" que extienden su funcionalidad para funciones tales como la seguridad y fiabilidad. Los módulos actualmente disponibles o en desarrollo incluyen: WS-ReliableMessaging,WS-Coordination,WS-AtomicTransaction,WS-Security,WS-Addressing.

Algunas de las principales características que ofrece Axis2 son los siguientes:

- Velocidad, soporte MEP, flexibilidad, estabilidad, orientado a componentes de implementación, soporte WSDL, composición y extensibilidad.

La versión 1.4 de Apache Axis2 tiene mejoras de rendimiento y correcciones como son:

Modelo de programación:

- Cliente centrado en XML API como WSDL y el apoyo de políticas.
- Apoyo a los servicios jaxws estilo y clientes.
- Apoyo a los servicios POJO y Spring y los clientes.
- Soporte para cualquier patrón de intercambio de mensajes.
- Modelo de despliegue de servicios archivados soportando toda la encapsulación del servicio con el apoyo de versiones.
- WSDL 2.0

Especificaciones compatibles

- SOAP 1.1 y 1.2.

- WSDL 1.1, incluyendo SOAP y enlaces HTTP.
- WS-Policy,SAAJ 1.1,transportes,HTTP,SMTP, JMS,TCP.

Herramientas incluidas en esta versión

- Aplicación Web Axis2 (aplicación web).
- WSDL2WS-plug-in Eclipse / versión de línea de comandos / IntelliJ IDEA plug-in/Maven2 WSDL2Code Plug-in.
- Servicio de Archivo Asistente-plug-in Eclipse / IntelliJ IDEA plug-in / Maven2 Plug-TCA. (26)

Con el desarrollo de este capítulo se realizó un estudio sobre el Nivel 7 de Salud(HL7) y su Servicio de Terminología Común (CTS), generando con esto una visión de los conceptos fundamentales que rodean al proceso para el intercambio de información en los sistemas desarrollados en el Centro de Informática Médica(CESIM). Se analizaron las ventajas que le brindarían al sistema propuesto algunas funcionalidades de sistemas terminológicos ya existentes en el ámbito internacional para lograr un mejoramiento en sus servicios. Se expusieron las herramientas, metodologías y tecnologías que serán utilizadas para lograr una mejor organización, exactitud y acceso al flujo de conocimientos e información que permitirá facilitar y mejorar las formas de trabajo.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

En el presente capítulo se realiza la descripción de los aspectos fundamentales relacionados con el objeto de estudio, el flujo actual de los procesos involucrados en el campo de acción, y se describen los procesos que son objeto de automatización. Debido a la no existencia de una definición clara de los procesos del negocio en los que tiene lugar la configuración del sistema, se decide desarrollar un Modelo de Dominio. Este abarca las definiciones asociadas a los conceptos encontrados en el entorno donde está enmarcado el sistema, así como las relaciones existentes entre ellos. Se muestra la especificación de los requerimientos del software (los requerimientos funcionales y no funcionales), además de la descripción de los casos de uso resultantes del flujo de trabajo de Requerimientos.

2.1 Modelo de Dominio

Es una representación visual estática del entorno real objeto del proyecto. Es un diagrama con los objetos que existen (reales) relacionados con el proyecto que se realizará y las relaciones que hay entre ellos. Ayuda a comprender los conceptos que utilizan los usuarios, los conceptos con los que trabajan y con los que deberá trabajar la aplicación. (27)

2.2 Conceptos fundamentales del dominio

Para brindar una mejor comprensión del diagrama del modelo de dominio a continuación se realiza una breve descripción de los conceptos encontrados en el problema.

2.2.1 Sistema de código

La Capa de Vocabulario trabaja sobre el supuesto de que existe una base de datos que contiene al menos una terminología o sistema de codificación(CodeSystem) que puede contener teóricamente un número ilimitado de conceptos codificados (CodedConcepts)

2.2.2 Versión del sistema de código

Un Sistema de Codificación (CodeSystem) puede representar opcionalmente una versión concreta (CodeSystemVersion) de una terminología en cualquier momento dado.

2.2.3 Concepto codificado

Representan una clase o tipo dentro de un dominio particular, es único dentro del sistema de código(CodeSystem) que lo define y debe tener al menos un nombre que lo designe (ConceptDesignation). Una vez definido, el significado del concepto codificado, que no sólo depende de la definición textual sino de su posición en la jerarquía de conceptos en el sistema de codificación y de sus relaciones explícitas con otros conceptos, no puede cambiar: es invariable. Igualmente, los conceptos existentes pueden ser retirados y pueden añadirse conceptos nuevos, pero una vez definido, el significado de un concepto debe permanecer estático.

2.2.4 Designación

Una designación o nombre de concepto (ConceptDesignation) es el nombre u otro símbolo textual que representa los conceptos codificados y representan el nombre de uno o más conceptos codificados, siendo equivalentes a las descripciones (Nombre Preferido, Nombre Completo y Sinónimos). Estos nombres dependen del lenguaje.

2.2.5 Propiedad del concepto

Los conceptos pueden, opcionalmente, ser caracterizados por propiedades (ConceptProperties) y una propiedad (ConceptProperty) es un atributo, faceta o cualquier otra característica que puede representar o definir al menos un concepto.

2.2.6 Relación de conceptos

Representa las relaciones sobre el conjunto de conceptos definido en un sistema de código y pueden ser relaciones binarias sobre el conjunto de conceptos definido en un sistema de codificación. Un concepto puede ser fuente y/u origen de relaciones con otros conceptos (ConceptRelationships).

2.3 Diagrama del modelo de dominio

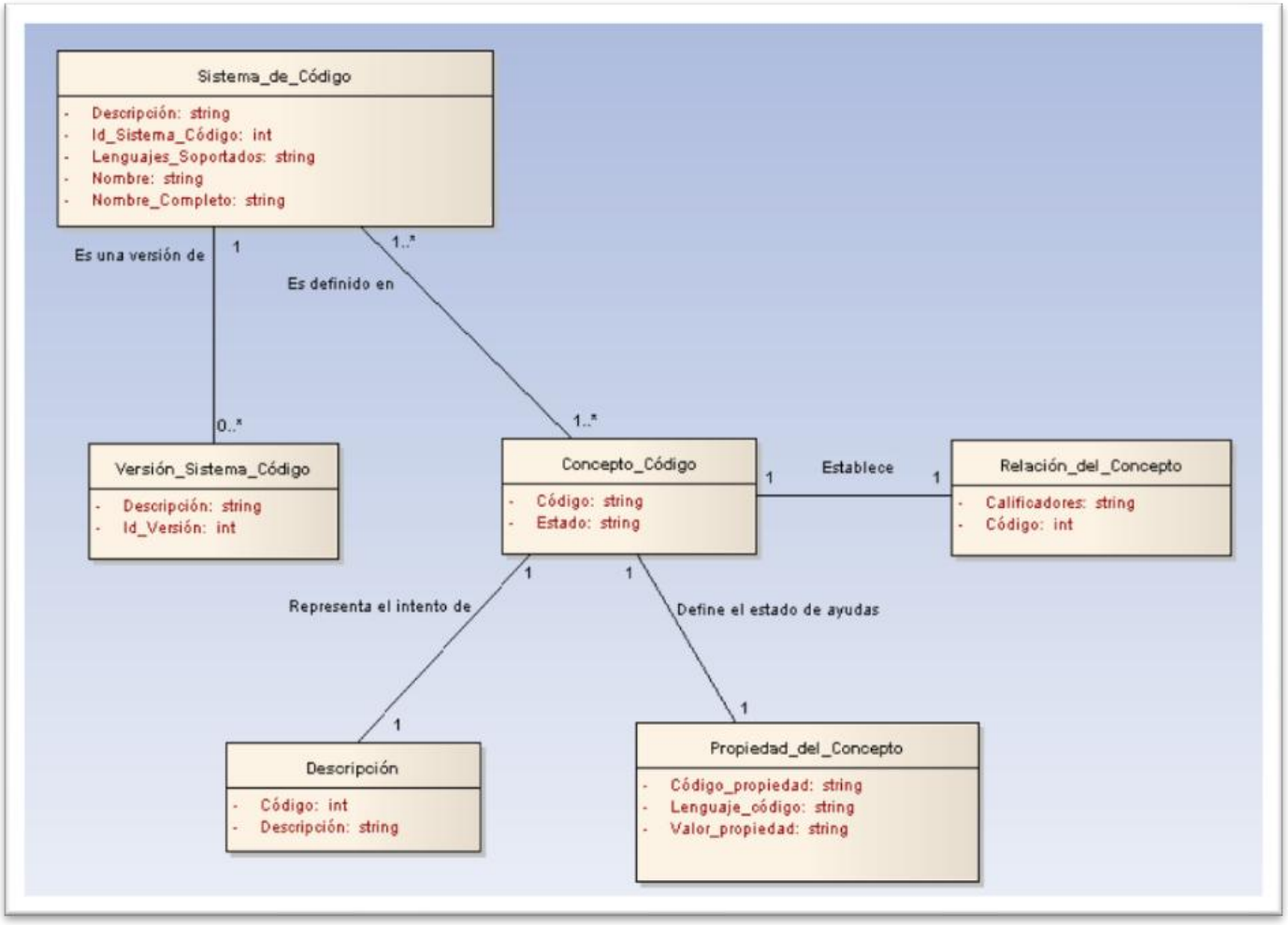


Figura 2.1. Modelo de Dominio.

2.4 Objeto de informatización

Con el presente trabajo se desarrolla un módulo que se comunica con el software de almacenamiento de terminología utilizando un modelo conceptual propio y que a la vez, habilita el manejo de terminologías clínicas. Entre los procesos de manejo de terminologías se encuentra: devolver información sobre algún sistema de código y código de concepto, las relaciones entre estos, así como las definiciones y propiedades que son más apropiadas para cada código de concepto. Además, se posibilitará buscar los

calificadores, tipos y lenguajes para cada código de concepto que pertenezca a un sistema de código específico.

De este modo, con el desarrollo de este módulo y su integración con los demás componentes del Servicio de Terminología Común(CTS), se lograría un puente de comunicación entre las aplicaciones sanitarias permitiendo que cada sistema pueda entender de manera correcta los mensajes de comunicación generados por otros sistemas que implemente estándares internacionales.

2.5 Información que se maneja

Es la referente a las terminologías médicas, como son los sistemas de código, códigos de concepto, definiciones, relaciones y otras entidades específicas de la terminología.

2.6 Propuesta del sistema

El Servicio de Terminología Común (CTS) se desarrolla como una alternativa a una estructura de datos común, el cual define una serie de interfaces de aplicaciones y un modelo de datos que permite integrar y acceder de manera uniforme a diferentes terminologías. El servicio que se quiere implementar está conformado por tres capas: la Capa de Mensajería, la de Vocabulario y la de Mapeo de códigos, además de un sistema de administración.

La Capa de Mensajes permite a una variedad de aplicaciones, crear, validar y traducir datos de una manera consciente y reproducible. Esta, se comunica con el software de mensajería en términos de dominios de vocabulario, es decir, contextos, conjunto de valores, atributos codificados y otros artefactos del modelo de mensajes. Además, sirve de puente entre las aplicaciones y la Capa de Vocabulario, además de utilizar la interfaz de vocabulario.

Un ejemplo de esta comunicación es cuando un sistema solicita un mensaje de tiempo de ejecución, el mensaje consiste en llenar los detalles de un atributo derivado. El servicio, hace varias llamadas a la interfaz de vocabulario con el fin de obtener las designaciones, conceptos de códigos, los nombres de todos los sistemas de códigos implementados y las versiones de los mismos. En este momento, es que entra a jugar un papel clave la Capa de Vocabulario, la cual está diseñada para ser genérica y fácilmente utilizable en diferentes entornos, es decir, no solo desde la Capa de Mensajes, sino desde cualquier

sistema. Esta se comunica con el software de almacenamiento el cual habilita el manejo de sistemas de código, códigos de conceptos, definiciones, relaciones y otras entidades específicas de la terminología.

La Capa de Mapeo cumple con las funcionalidades específicas de la Capa de Vocabulario y se encarga de traducir los mensajes enviados. Traduce el sistema de código que contiene la relación entre los conceptos, las descripciones y el código. Los mapeos permitirán reutilizar la información para fines actuales y futuros. Por último un Sistema de Administración, el cual se comunica con el software de almacenamiento permitiendo hacerle configuraciones a la fuente de información del CTS y ver la información del mismo. También brinda la facilidad de realizar actualizaciones según vayan surgiendo.

La realización del CTS, brindará la posibilidad de tener una mejor integración entre los sistemas de salud, ya que implementa funcionalidades que un recurso terminológico debe poseer. Además, proporciona una interfaz común y un modelo de referencia para la comprensión del cliente. El servicio será compatible con la nomenclatura, el modelo y el enfoque expresado por el vocabulario de HL7, por las versiones del Modelo de Referencia de Información (RIM) y sus estructuras derivadas.

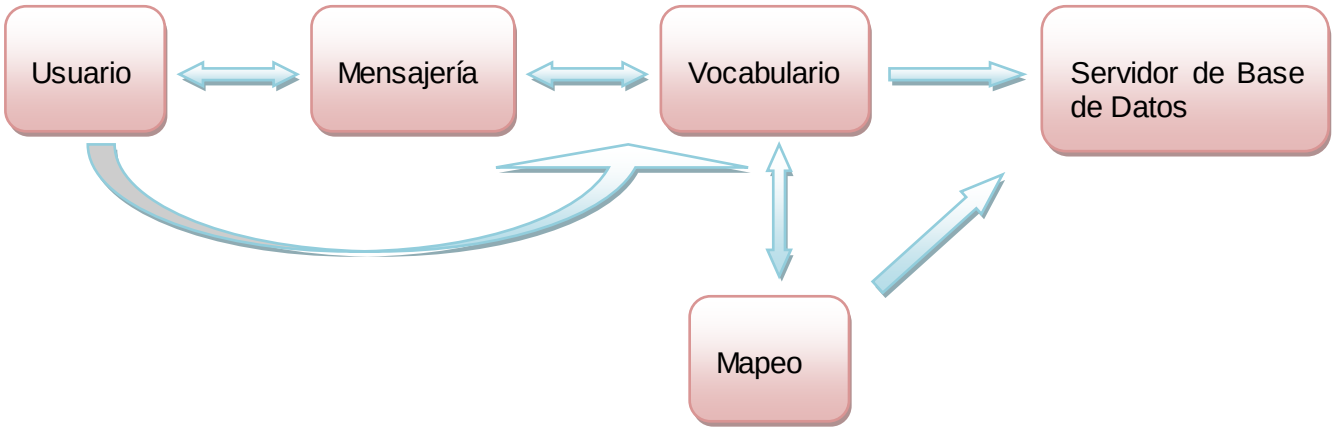


Figura 2.2. Ejemplo del sistema.

2.7 Especificación de los requerimientos del software

Los requerimientos de un sistema son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requerimientos reflejan las necesidades de los clientes de un sistema que

ayude a resolver algún problema como el control de un dispositivo, hacer un pedido o encontrar información. (28)

Los requerimientos de software tienen varias clasificaciones entre las que se encuentran los funcionales y no funcionales.

2.7.1 Requerimientos funcionales

Son declaraciones de los servicios que deben reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. Los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. (29)

Luego de ser conocido algunos de los conceptos fundamentales que encierra el objeto de estudio, se analiza qué debe hacer el sistema para dar cumplimientos a esos objetivos planteados. Dentro de los requerimientos se tienen, las acciones ocultas que deben realizar el sistema y las condiciones extremas a determinar por el sistema.

A continuación se muestran el listado de los requerimientos funcionales:

RF_1. Obtener el nombre del servicio (getServiceName).

RF_2. Obtener la versión del servicio (getServiceVersion).

RF_3. Obtener la descripción del servicio (getServiceDescription).

RF_4. Obtener la versión del CTS (getCTSVersion).

RF_5. Obtener los sistemas de códigos soportados (getSuportedCodeSystems).

RF_6. Buscar una información detallada de un sistema de código (lookupCodeSystemInfo).

RF_7. Verificar si el identificador del concepto es válido (isConceptIdValid).

RF_8. Buscar la designación apropiada para un sistema de código (lookupDesignation).

RF_9. Verificar si los códigos están relacionados según las condiciones (areCodesRelated).

RF_10. Obtener los algoritmos soportados (getSupportedMatchAlgorithms).

RF_11. Buscar los códigos de concepto por designación (lookupConceptCodesByDesignation).

RF_12. Buscar los códigos de concepto por propiedad (lookupConceptCodesByProperty).

RF_13. Buscar una información detallada del concepto de código (lookupCompleteCodedConcept).

RF_14. Buscar las designaciones de un sistema de código (lookupDesignations).

RF_15. Buscar las propiedades de un sistema de código (lookupProperties).

RF_16. Buscar las relaciones de un concepto de código (lookupCodeExpansion).

RF_17. Buscar la expansión del código que puede expandirse (expandCodeExpansionContext).

2.7.2 Requerimientos no funcionales:

Son restricciones de los servicios o funciones ofrecidos por el sistema. Incluye restricciones de tiempo, sobre el proceso del desarrollo y estándares. A menudo se aplican al sistema en su totalidad. Normalmente, apenas se aplican a características o servicios individuales del sistema.(30) Con el propósito de responder la necesidad de intercambiar información entre los sistemas desarrollados en el Centro, se definió un conjunto de propiedades que debe cumplir el sistema, las cuales se describen a continuación.

Fiabilidad:

RNF_1. Estará disponible todo el tiempo permitiendo el trabajo de los sistemas cada vez que lo requieran.

RNF_2. Debe ser estable, fiable y la velocidad de respuestas será rápida durante la utilización del mismo.

RNF_3. El tiempo medio de reparación si existe una falla en el sistema no debe exceder las 4 horas.

Eficiencia:

RNF_4. Dará una respuesta a la hora de consultar alguna información en la base de datos sin exceder el tiempo de espera del usuario en 2 segundos.

RNF_5. Garantiza la rapidez de respuestas antes peticiones de los clientes, al igual que la velocidad de procesamiento de la información.

Soporte:

RNF_6. Para su correcta comprensión debe poseer una documentación del código, necesario para el aprendizaje sobre el uso del sistema.

RNF_7. Podrá ser compatible con sistemas desarrollados en el Centro de Informática Médica (CESIM).

Portabilidad:

RNF_8. Podrá ser accedida sin importar el navegador que se esté utilizando o sobre el sistema operativo que se esté trabajando.

Usabilidad:

RNF_9. Fácil acceso a las funcionalidades.

Seguridad:

RNF_10. Se podrá acceder a toda la información disponible pero no se permitirá modificar ni eliminar dicha información.

Diseño e Implementación:

RNF_11. Se utilizará Eclipse como plataforma de desarrollo y Java como lenguaje de programación.

Software:

RNF_12. Sistema Operativo Linux.

RNF_13. Servidor de Base de Datos PostgreSQL 8.4.

Hardware:

Requisitos mínimos:

- Servidor de aplicaciones:

RNF_14. Computador Pentium a 2GHz, memoria RAM de 1GB, 20 GB de espacio libre del disco duro y una tarjeta de red.

- Servidor de base de datos:

RNF_15. Computador Pentium a 3GHz, memoria RAM de 2GB, 60 GB de espacio libre del disco duro y una tarjeta de red.

2.8 Modelo de casos de uso del sistema

Los casos de uso del sistema permiten que los desarrolladores y a los clientes llegar a un acuerdo sobre las condiciones y posibilidades que debe cumplir el sistema. Estos casos de usos se utilizan para modelar cómo funciona el sistema y proporcionan la entrada fundamental para el análisis, diseño e implementación en el desarrollo del sistema de software.

Definición de actores

Actores	Objetivo
Sistemas externos.	Es el que realiza la solicitud al servicio para obtener alguna información que le sea necesaria conocer.

Tabla 2.1: Actores del sistema.

2.8.1 Diagrama de casos de uso

Sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con otros sistemas. Es un diagrama que muestra la relación entre los actores y los casos de uso en un sistema. Los diagramas de casos de uso se utilizan para ilustrar los requerimientos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo. (31)

Se encuentran divididos en paquetes con el objetivo de facilitar el entendimiento y comprensión del mismo. Los paquetes son un mecanismo de propósito general para organizar elementos en grupos. (32) Se definen 3 paquetes encapsulando en cada uno de ellos las funcionalidades del sistema. A continuación se muestra cada diagrama:

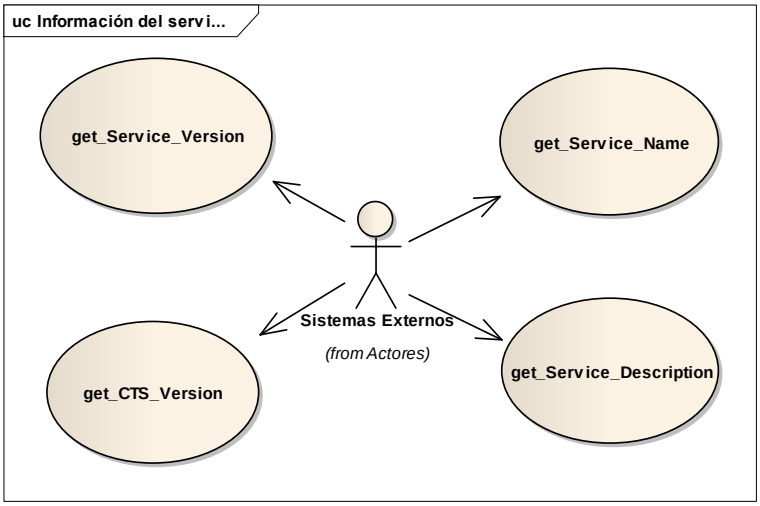


Figura 2.3. Diagrama de casos de uso del sistema - Paquete información del servicio.

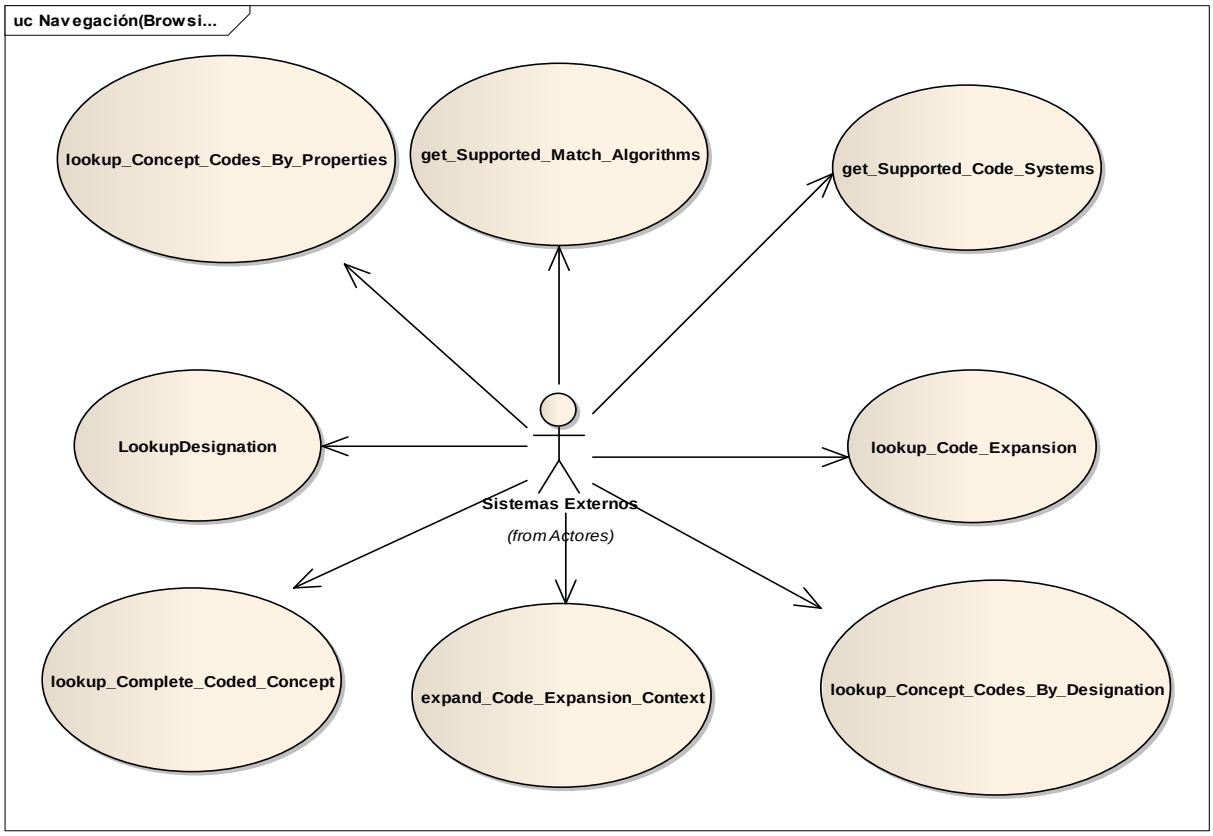


Figura 2.4. Diagrama de casos de uso del sistema - Paquete de Navegación (Browsing).

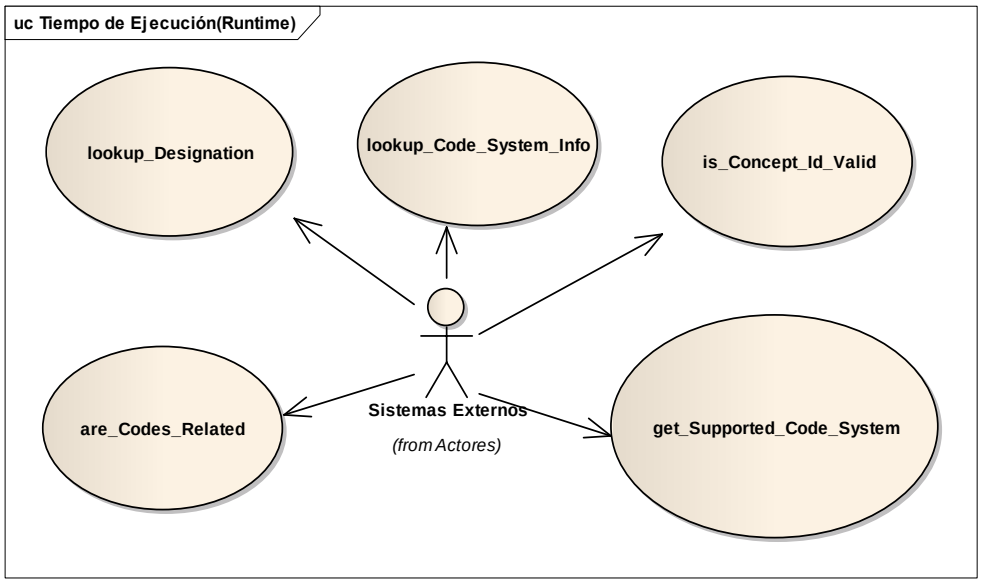


Figura 2.5. Diagrama de casos de uso del sistema - Paquete Tiempo de Ejecución (Runtime).

2.8.2 Descripción de los casos de uso

CU-1	Obtener el nombre del servicio (getServiceName).
Actor	Sistemas Externos
Descripción	Devuelve el nombre asignado a este servicio por el proveedor de servicios.
Referencia	RF 1

Tabla 2.2: Descripción del caso de uso getServiceName.

CU-2	Obtener la versión del servicio (getServiceVersion).
Actor	Sistemas Externos.
Descripción	Devuelve la versión actual del software de servicio.
Referencia	RF 2

Tabla 2.3: Descripción del caso de uso getServiceVersion.

CU-3	Obtener la descripción del servicio (getServiceDescription).
Actor	Sistemas Externos.
Descripción	Devuelve una descripción de la función de servicio que está implementado.
Referencia	RF 3

Tabla 2.4: Descripción del caso de uso getServiceDescription.

CU-4	Obtener la versión del CTS (getCTSVersion).
Actor	Sistemas Externos.
Descripción	Devuelve la versión de CTS que implementan estos servicios.
Referencia	RF 4

Tabla 2.5: Descripción del caso de uso getCTSVersion.

CU-5	Obtener los Sistemas de Códigos soportados (getSuportedCodeSystems).
Actor	Sistemas Externos.
Descripción	Devuelve el identificador, nombre y las versiones liberadas de todos los sistemas de códigos que son soportados por el servicio.
Referencia	RF 5

Tabla 2.6: Descripción del caso de uso getSuportedCodeSystems.

CU-6	Búsqueda de información del Sistema de Código (lookupCodeSystemInfo).
Actor	Sistemas Externos.
Descripción	Devuelve información detallada sobre un sistema de código específico.
Referencia	RF 6

Tabla 2.7: Descripción del caso de uso lookupCodeSystemInfo.

CU-7	Concepto de identificación válido (isConceptIdValid).
Actor	Sistemas Externos.
Descripción	Determine si el código de concepto es válido, dado un sistema de código específico.
Referencia	RF 7

Tabla 2.8: Descripción del caso de uso isConceptIdValid.

CU-8	Busca la designación (lookupDesignation).
Actor	Sistemas Externos.
Descripción	Devuelve la designación preferida para el código de concepto en el idioma suministrado.
Referencia	RF 8

Tabla 2.9: Descripción del caso de uso lookupDesignation.

CU-9	Códigos relacionados (areCodesRelated).
Actor	Sistemas Externos.
Descripción	Determine si la relación nombrada existe entre la fuente y el código objetivo.
Referencia	RF 9

Tabla 2.10: Descripción del caso de uso areCodesRelated.

CU-10	Obtener los algoritmos soportados (getSupportedMatchAlgorithms).
Actor	Sistemas Externos.
Descripción	Devuelve los algoritmos soportados por el servicio.
Referencia	RF 10

Tabla 2.11: Descripción del caso de uso getSupportedMatchAlgorithms.

CU-11	Busca los códigos de concepto por designación (lookupConceptCodesByDesignation).
Actor	Sistemas Externos.
Descripción	Devuelve una lista de códigos de concepto cuyas designaciones se correspondan con la cadena de texto buscada, en el lenguaje introducido (si dichos códigos existen).
Referencia	RF 11

Tabla 2.12: Descripción del caso de uso lookupConceptCodesByDesignation.

CU-12	Busca los códigos de concepto por propiedad (lookupConceptCodesByProperty).
Actor	Sistemas Externos.
Descripción	Devuelve una lista de códigos de concepto que tienen propiedades que se encuentran en los criterios suministrados.
Referencia	RF 12

Tabla 2.13: Descripción del caso de uso lookupConceptCodesByProperty.

CU-13	Búsqueda completa del código de concepto (lookupCompleteCodedConcept).
Actor	Sistemas Externos.
Descripción	Devuelve una descripción completa del código de concepto suministrado.
Referencia	RF 13

Tabla 2.14: Descripción del caso de uso lookupCompleteCodedConcept.

CU-14	Búsqueda por designaciones (lookupDesignations).
Actor	Sistemas Externos.
Descripción	Devuelve de todas las designaciones para el código de concepto suministrado que coinciden con los criterios suministrados.

Referencia	RF 14

Tabla 2.15: Descripción del caso de uso lookupDesignations.

CU-15	Búsqueda por propiedades (lookupProperties).
Actor	Sistemas Externos.
Descripción	Devuelve las propiedades de la identificación del código del sistema dado o código de tipo conceptual que coinciden con los criterios suministrados.
Referencia	RF 15

Tabla 2.16: Descripción del caso de uso lookupProperties.

CU-16	Búsqueda de expansión de códigos (lookupCodeExpansion).
Actor	Sistemas Externos.
Descripción	Recursivamente lista los códigos de concepto que se relaciona con el concepto facilitado, incluido la designación preferida para los códigos.
Referencia	RF 16

Tabla 2.17: Descripción del caso de uso lookupCodeExpansion.

CU-17	Expande el contexto de expansión de códigos (expandCodeExpansionContext).
Actor	Sistemas Externos.
Descripción	Devuelve una lista expandida de códigos, a ésta ser devuelta después de una búsqueda de expansión de contextos.
Referencia	RF 17

Tabla 2.18: Descripción del caso de uso expandCodeExpansionContext.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

El desarrollo de este capítulo permitió abordar en una propuesta del sistema el proceso de integración de la Capa de Vocabulario con el resto de las capas del Servicio de Terminología Común(CTS),logrando así un mejor entendimiento del sistema a desarrollar. Se realizó un Modelo de Dominio donde se especificaron las relaciones entre los sistemas de códigos, conceptos de código y sus características, propiedades y descripciones, posibilitando una mejor comprensión del entorno de trabajo. Se detallaron los requerimientos funcionales y no funcionales del sistema, se identificaron los actores que intervienen en el mismo; así como todos los casos de uso, brindando una especificación de las funcionalidades recogidas en los requerimientos.

CAPÍTULO 3. ANÁLISIS Y DISEÑO

Después de tener un resultado del flujo de trabajo Requerimientos donde se obtiene una vista externa del sistema, se abordará los aspectos relacionados con el flujo de trabajo Análisis y Diseño. Se detallan los casos de uso identificados permitiendo reflejar una vista interna de lo que sería el sistema y se determinan las clases necesarias para llevar a cabo funcionalidades contenidas en ellos. Se representan los diagramas de secuencia, de clases del diseño, con las descripciones de las clases identificadas y el diseño de la base de datos a través del modelo de datos, con la descripción de las tablas correspondientes.

3.1 Modelo de Diseño

El diseño es el centro de atención al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Esto contribuye a una arquitectura estable y sólida, creando un plano del modelo de implementación. En el diseño se modela el sistema y se encuentra su forma para que soporte todos los requerimientos, incluyendo los no funcionales y las restricciones que se le suponen. Una entrada esencial en el diseño es el resultado del análisis, que proporciona una comprensión detallada de los requerimientos. Además, impone una estructura del sistema, que debe conservarse lo mejor posible cuando se de forma al sistema.

Es además, un modelo físico, no genérico, específico para una implementación. Da forma al sistema mientras que intenta preservar la estructura definida por el modelo de análisis lo más posible. Es un manifiesto del diseño del sistema, incluyendo su arquitectura y debe ser mantenido durante todo el ciclo de vida del software. (33)

3.1.1 Diagrama de clase del diseño

Muestra un conjunto de clases, interfaces y colaboraciones, así como sus relaciones. Los diagramas de clases se utilizan para modelar la vista de diseño estático de un sistema. Principalmente, incluye modelar el vocabulario del sistema, las colaboraciones o los esquemas.(34)

(Remitirse a: Expediente. Proyecto\1. ingeniería\1.2 arquitectura y diseño, al documento TIE_ST_0121_MDI)

3.2 Diagrama de Iteración (Secuencia)

Los Diagramas de Colaboración y Secuencia, conocidos como Diagramas de Interacción se utilizan para representar los aspectos dinámicos del sistema.(35) Los primeros se ocupan de mostrar las relaciones entre los objetos y los mensajes que intercambian; mientras que los segundos se encargan de las interacciones entre los objetos a través del tiempo. En el diseño es preferible realizar Diagramas de Secuencia, pues uno de los intereses de esta etapa es encontrar secuencias de interacciones detalladas y ordenadas en tiempo.

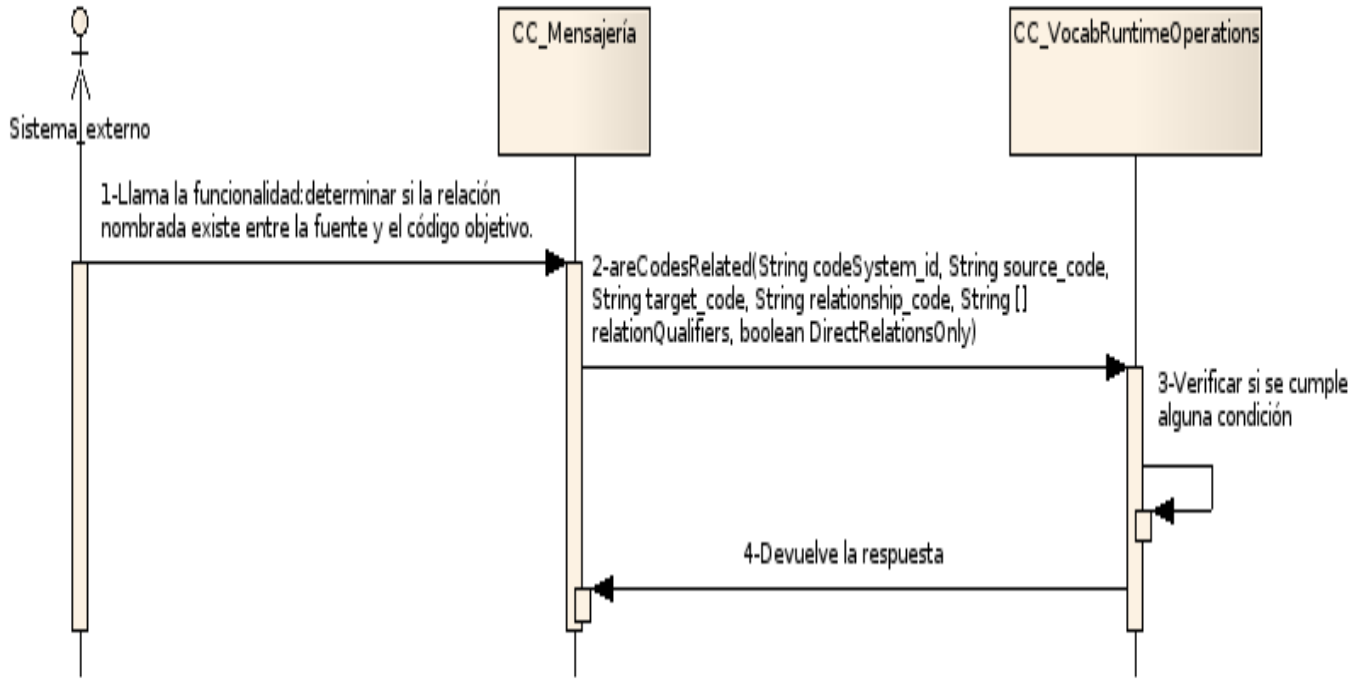


Figura 3.3. Diagrama de secuencia del diseño del caso de uso areCodesRelated.

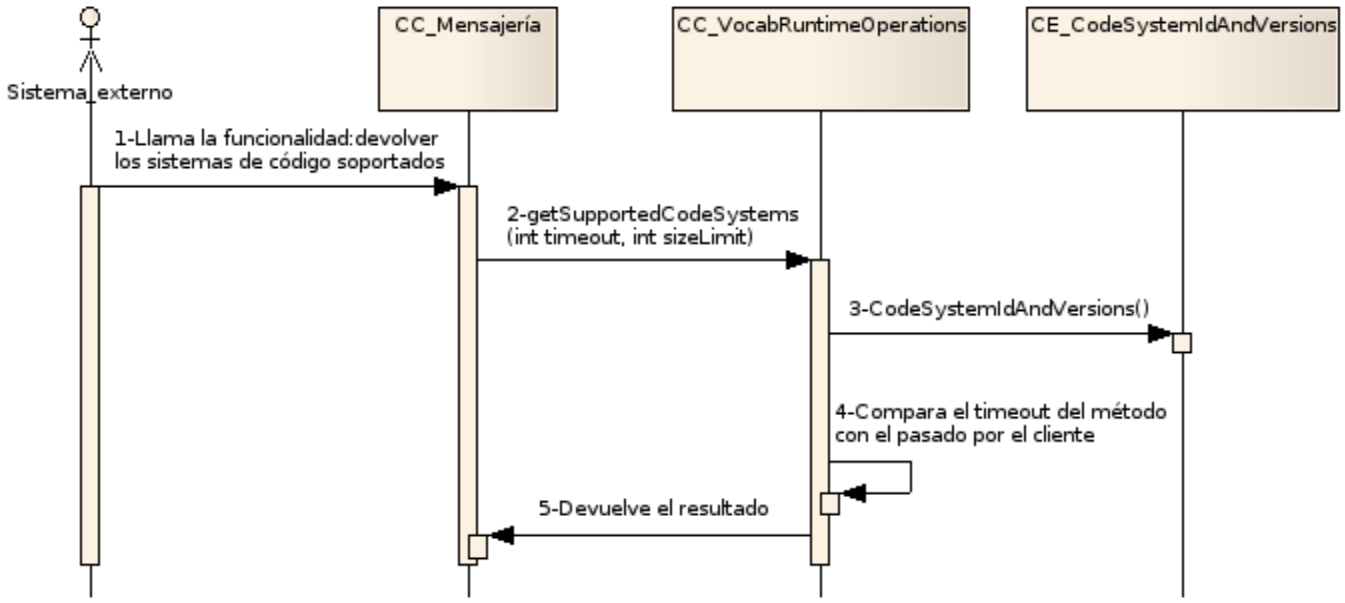


Figura 3.4. Diagrama de secuencia del diseño del caso de uso getSupportedCodeSystems.

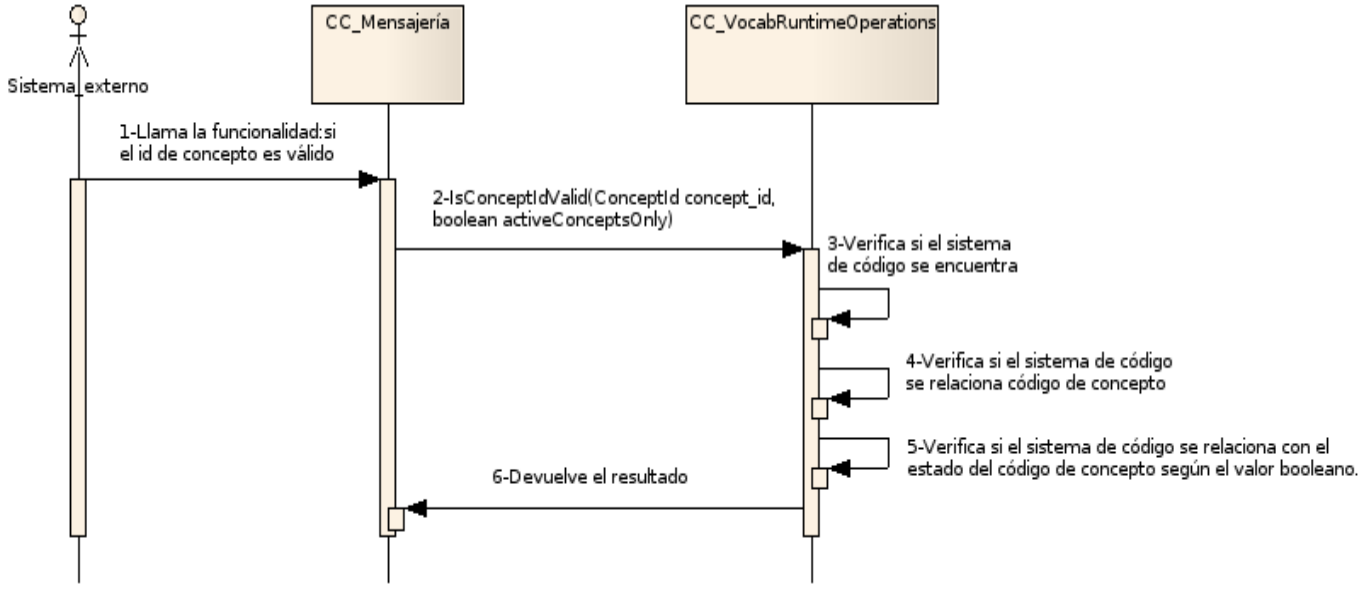


Figura 3.5. Diagrama de secuencia del diseño del caso de uso isConceptIdValid.

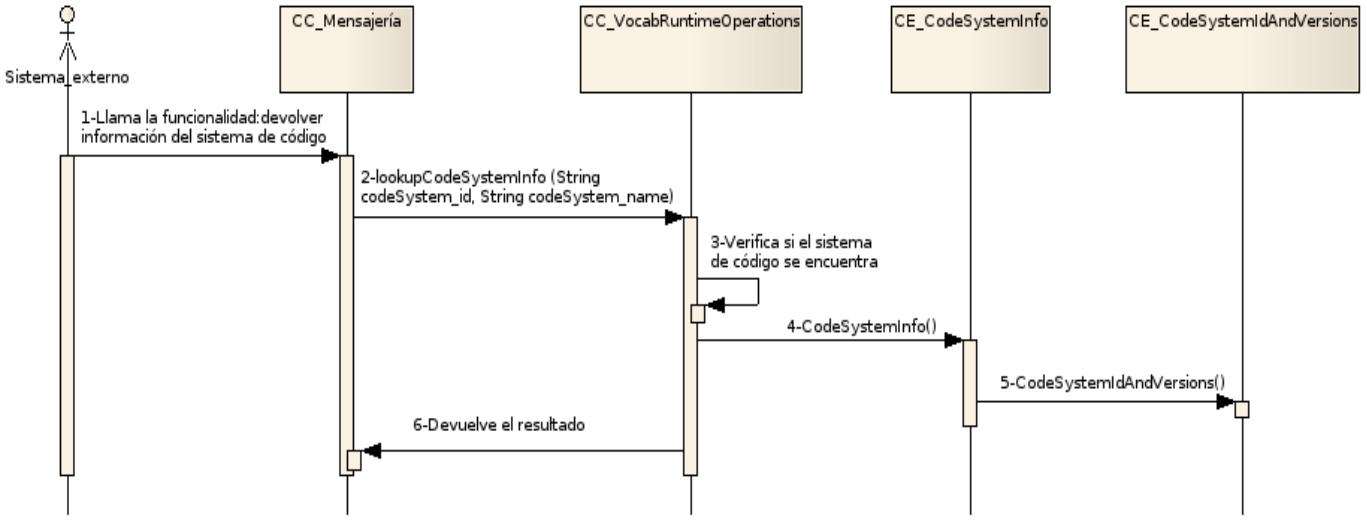


Figura 3.6. Diagrama de secuencia del diseño del caso de uso lookupCodeSystemInfo.

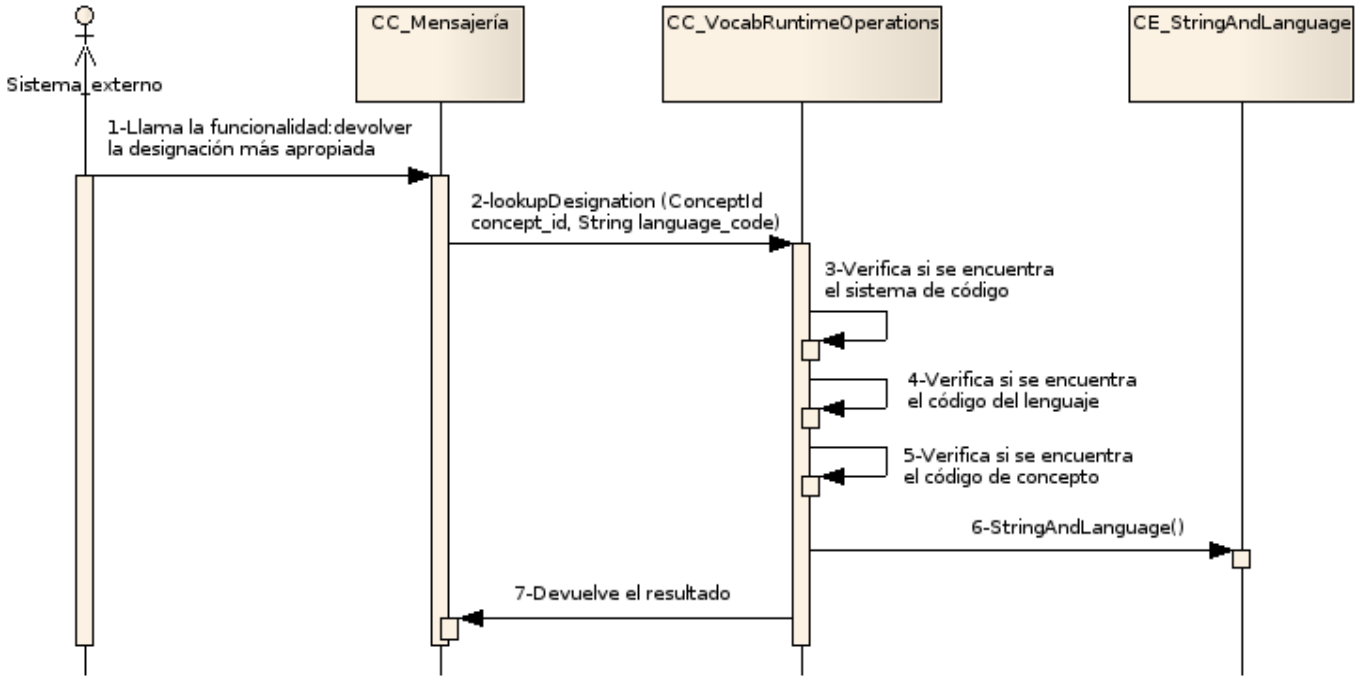
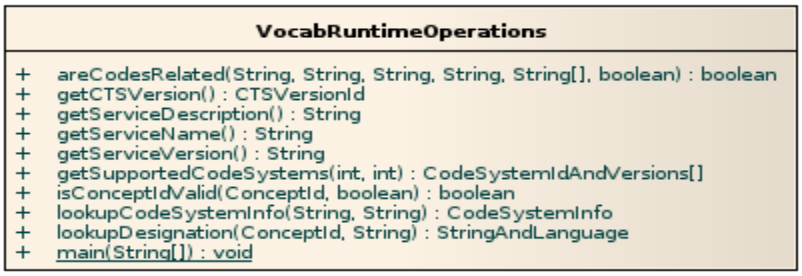


Figura 3.7. Diagrama de secuencia del diseño del caso de uso lookupDesignation

3.3 Descripción de las clases y sus atributos

A continuación se realiza la descripción de las principales clases que han sido identificadas en el diseño para su futura implementación, con el objetivo de lograr una comprensión más amplia del sistema a implementar.

<table>
<tr><td colspan="2">Nombre: VocabRuntimeOperations</td></tr>
<tr><td colspan="2">Tipo de clase: controladora.</td></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><td colspan="2">Para cada responsabilidad:</td></tr>
<tr><td>Nombre:</td><td>getCTSVersion():string</td></tr>
<tr><td>Descripción:</td><td>Retorna la versión de CTS implementada en el servicio.</td></tr>
<tr><td>Nombre:</td><td>getServiceDescription() : string</td></tr>
<tr><td>Descripción:</td><td>Retorna una descripción del servicio.</td></tr>
<tr><td>Nombre:</td><td>getServiceName() : string</td></tr>
<tr><td>Descripción:</td><td>Retorna el nombre asignado a este servicio por el proveedor de servicios.</td></tr>
<tr><td>Nombre:</td><td>getServiceVersion() : string</td></tr>
<tr><td>Descripción:</td><td>Retorna la versión actual del software de servicio.</td></tr>
<tr><td>Nombre:</td><td>areCodesRelated(): boolean</td></tr>
<tr><td>Descripción:</td><td>Determina si la relación nombrada existe entre la fuente y el código objetivo.</td></tr>
</table>

Nombre:	getSupportedCodeSystems(): CodeSystemIdAndVersion[ ]
Descripción:	Retorna el identificador, nombre y las versiones liberadas de todos los sistemas de códigos que son soportados por el servicio.
Nombre:	IsConceptIdValid(): boolean
Descripción:	Determina si el código de concepto es válido con el sistema de código especificado.
Nombre:	LookupCodeSystemInfo(): CodeSystemInfo
Descripción:	Retorna información detallada sobre un sistema de código específico.

Tabla 3.1: Descripción de la clase controladora VocabRuntimeOperations.

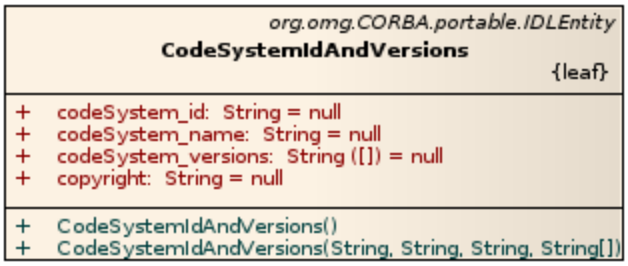
Nombre: CodeSystemIdAndVersion.	
Tipo de clase: entidad.	
	
Atributo	**Tipo**
codeSystem_id	string
codeSystem_name	string
codeSystem_versions	string
copyright	string

Tabla 3.2: Descripción de la clase CodeSystemIdAndVersion.

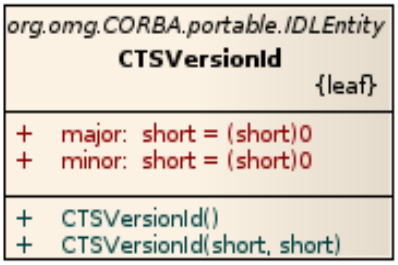
Nombre: CTSVersionId	
Tipo de clase: entidad	
	
Atributo	**Tipo**
major	short
minor	short

Tabla 3.3: Descripción de la clase CTSVersionId.

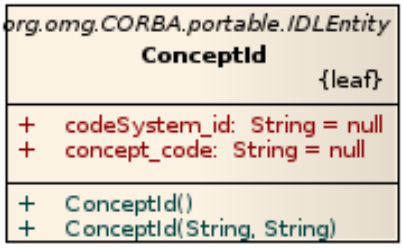
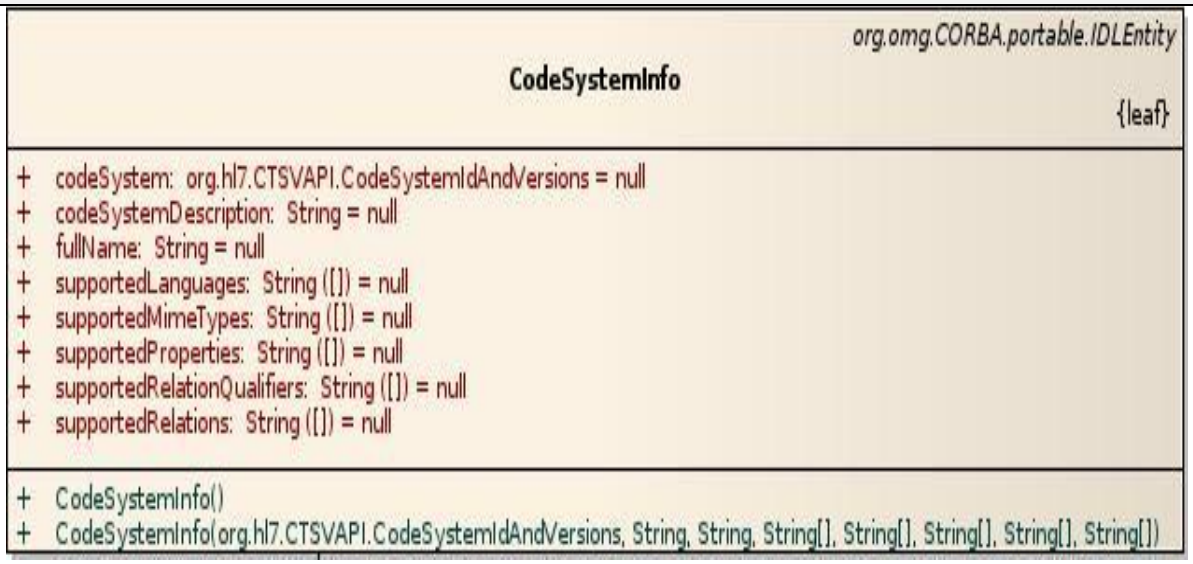
Nombre: ConceptId	
Tipo de clase: entidad	
	
Atributo	**Tipo**
codeSystem_id	string
concept_code	string

Tabla 3.4: Descripción de la clase ConceptId.

Nombre: CodeSystemInfo
Tipo de clase: entidad



Atributo	**Tipo**
codeSystem	CodeSystemIdAndVersion.
codeSystemDescription	string
fullName	string
supportedLanguages	string[ ]
supportedMimeTypes	string[ ]
supportedProperties	string[ ]
supportedRelationQualifiers	string[ ]
supportedRelations	string[ ]

Tabla 3.5: Descripción de la clase CodeSystemInfo.

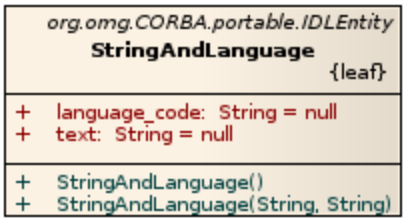
Nombre: StringAndLanguage	
Tipo de clase: entidad	
	
Atributo	**Tipo**
language_code	string
text	string

Tabla 3.6: Descripción de la clase StringAndLanguage.

3.4 Diseño de la base de datos

3.4.1 Modelo de datos

Un modelo de datos es un lenguaje utilizado para la descripción de una base de datos. Permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base). Describe los elementos que intervienen en una realidad o en un problema dado y la forma en que se relacionan dichos elementos entre sí.(36)

(Remitirse a: Expediente. Proyecto\1. ingeniería\1.2 arquitectura y diseño, al documento TIE_ST_0120_ArqSWv1.0.doc)

Las descripciones de las tablas del modelo de datos están en el expediente de proyecto que se encuentra en el repositorio del proyecto.

(Remitirse a: Expediente. Proyecto\1. ingeniería\1.2 arquitectura y diseño, al documento TIE_ST_0120_ArqSWv1.0.doc)

CAPÍTULO 3. ANÁLISIS Y DISEÑO

En este capítulo se ha completado el flujo de análisis y diseño, obteniéndose el modelo de diseño con el que se logra un acercamiento a la implementación. Se presentan los Diagramas de Secuencia, los que describen el proceso para dar solución a las solicitudes realizadas al sistema, brindando un mayor entendimiento para el desarrollo. Igualmente, se describieron las clases fundamentales del sistema y sus funcionalidades, puntualizando la forma en que estarán organizadas y estructuradas para la implementación. Quedaron asentadas las bases para la implementación de la Capa de Vocabulario del Servicio de Terminología Común.

CAPÍTULO 4. IMPLEMENTACIÒN Y PRUEBA

Se centra fundamentalmente en la implementación del sistema mediante los resultados obtenidos durante el diseño en términos de componentes, con el objetivo de dar solución a los requerimientos especificados desarrollando el modelo de componentes. También se expondrá el modelo de pruebas, por lo que se realizará una caracterización de las pruebas de caja negra, al ser este tipo de método el que se ha decidido utilizar, definiéndose y describiéndose los casos de pruebas definidos.

4.1 Implementación

El flujo de trabajo de implementación se empieza con el resultado del diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. (37)

4.1.1 Modelo de componentes

Describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos de software que entran en la fabricación de aplicaciones informáticas, pueden ser archivos, paquetes, bibliotecas cargadas dinámicamente, entre otros.(38) A continuación se presenta una vista general del modelo de implementación expresado en un diagrama de componentes.

(Remitirse a: Expediente. Proyecto\1. ingeniería\1.2 arquitectura y diseño, al documento TIE_ST_0120_ArqSWv1.0.doc)

4.2 Modelo de despliegue

El modelo de despliegue describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. (39)

De manera general el diagrama de despliegue de la Capa de Vocabulario para el CTS quedaría de la siguiente manera:

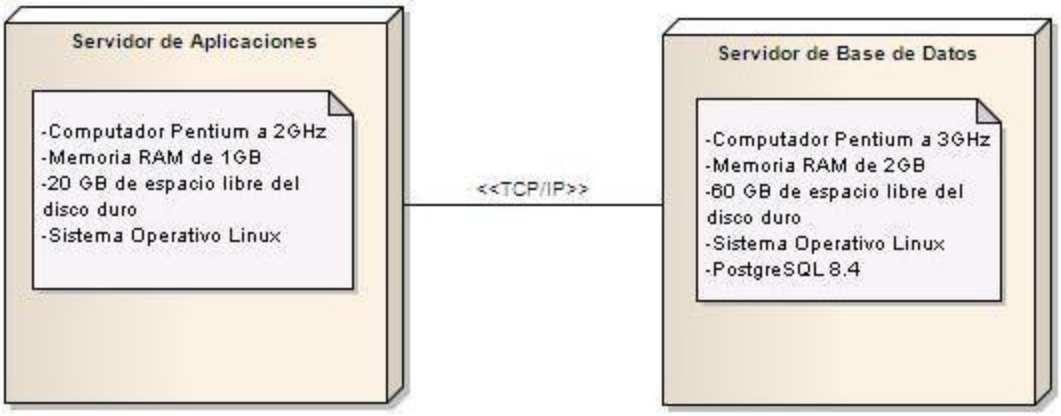


Figura 4.1. Diagrama del Modelo de Despliegue.

4.3 Estándares de codificación

Los lenguajes de programación tienen una serie de normas de estilo, para ser el código más elegante, comprensible, reutilizable, fácil de depurar y mantener. Para la codificación del servicio se tuvo en cuenta la especificación de HL7.

4.4 Tratamiento de excepciones

Durante el tiempo de ejecución de un sistema pueden fracasar diferentes rutinas, es a esto a lo que comúnmente se le llama excepción. Las excepciones son situaciones anómalas que requieren un tratamiento especial. No necesariamente son errores. Si se consigue dominar su programación, la calidad de las aplicaciones que se desarrollen aumentará considerablemente. (40)

En la implementación de la Capa de Vocabulario se tuvo en cuenta todas las facilidades que brinda la especificación HL7 para el tratamiento de excepciones. Para cada fragmento de código donde se espera una situación anómala, se definen las excepciones correspondientes. Estas excepciones son manejadas internamente y resueltas al momento en que se producen, sin ser lanzadas a instancias superiores, de forma tal que se haga la mejor acción posible. De esta forma se evita una salida de mensajes de error probablemente incomprensibles para el cliente.

```
        else
        {
            throw new TimeoutError();
        }
    }
     catch (TimeoutError e) {

        throw e;
    }
    catch (Throwable e) {

        UnexpectedError ue = new UnexpectedError();
        ue.setStackTrace(e.getStackTrace());
        ue.possible_cause = e.getMessage();

        throw ue;
    }
```

Figura 4.2. Ejemplo de tratamiento de excepciones

4.5 Fase de prueba

Las pruebas son un proceso de ejecución de programas con la intención de descubrir sus errores, las cuales tiene éxito si descubre un error no detectado hasta entonces. Los buenos casos de prueba son aquellos que tienen una alta probabilidad de mostrar un error no descubierto hasta entonces.(41)

El principal objetivo de las pruebas que se efectuarán, son evaluar la calidad del producto desarrollado a través de las diferentes fases por las que transita mediante la aplicación de pruebas concretas para validar las suposiciones hechas en el diseño, así como que los requerimientos se estén cumpliendo satisfactoriamente.

Para verificar que el producto funcione como se diseñó y que los requerimientos son cumplidos cabalmente, se realizaron pruebas de caja negra, las cuales tienen como propósito verificar las relaciones de entrada y salida de una unidad. Además, se centra en los requisitos funcionales del software y su objetivo es verificar que hace la unidad, pero sin averiguar como lo hace. (42)

Se realizó un diseño de caso de prueba para definir una forma de probar el sistema, incluyendo entradas de datos, resultados esperados, y las condiciones bajo las cuales se probará. En la primera iteración, para realizarles pruebas a cada una de las funcionalidades, se montó un ambiente de prueba que consistió en la creación de una clase que contaría con cada funcionalidad. Aquí se verificaron que sus tiempos de respuesta fueran lo menor posible, se visualizaron para probar que cada funcionalidad devolviera los

resultados debidamente y cumplieran con sus especificaciones. Además, se le entraron datos incorrectos para comprobar que se lanzaran las debidas excepciones, en todos los casos se obtuvo una buena respuesta de las funcionalidades al lanzar dichas excepciones, lo que asegura una correcta implementación de cada uno de estos algoritmos.

4.6 Seguridad del sistema

Con el propósito de garantizar la integridad de la información terminológica, se llevarán a cabo normas de seguridad que protejan los datos referentes a enfermedades o padecimientos, ya sean pasados, presente o previsiblemente. Igualmente, para proteger datos meramente administrativos que estén ubicados físicamente en las dependencias sanitarias, o no estándolo ahí, dependan de esta.

La seguridad de la aplicación será responsabilidad del servicio en general (CTS), en el cuál se tendrán en cuenta una serie de características entre las que se encuentran, ejecución del código, comprobación de la seguridad del código, compilación y demás servicios del sistema. Los requerimientos de seguridad serán iguales para todos los sistemas. Si fallara la seguridad podría traer consigo la alteración, pérdida, tratamiento o acceso no autorizado a los datos.

En el código se programa el tratamiento de las excepciones para evitar una salida de mensajes de error o fallos en la aplicación probablemente incomprensibles para el cliente y así aumentar la calidad del sistema. Las excepciones permitirán un seguimiento hasta guardar información acerca del lugar dónde se produjo el error y de los parámetros de configuración del sistema que lo provocaron. Se validan los datos recibidos de los sistemas externos para evitar posibles fallos en el sistema y disminuir el tiempo de ejecución del mismo. Los dispositivos conectados al sistema deberán estar físicamente ubicados en lugares que garanticen la confidencialidad de los datos.

Deberá existir un plan de copias de seguridad, que debe cumplir requisitos como son hacer una copia total cada quince días y que en cada equipo donde residen datos, sólo un usuario privilegiado ("root") tendrá acceso a los datos copiados procedentes de ese equipo. Además, las copias de seguridad deben permitir hacer recuperaciones totales de la información. El equipamiento para la realización de las copias de seguridad y los soportes externos usados deberán estar ubicados en un local diferente al de los equipos de tratamiento de datos.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA

En la realización de este capítulo se llevó a cabo el flujo de trabajo implementación, considerando la arquitectura definida por el departamento de Tecnología, Integración y Estándares, logrando así la solución final para la Capa de Vocabulario del CTS. Se obtuvo el diagrama de componentes para representar los elementos relacionados con la implementación y las relaciones que existen entre cada uno de ellos. La distribución física del sistema pudo ser visualizada a través del diagrama de despliegue así como los requerimientos no funcionales de hardware con que debe ser ejecutado el servicio. Se llevó a cabo el proceso de pruebas, donde fue evaluado el sistema de acuerdo a los diseños de casos de pruebas definidos, logrando así que se le dieran cumplimiento a los requisitos.

CONCLUSIONES

Una vez finalizado el desarrollo de la Capa de Vocabulario para el Servicio de Terminología Común(CTS), se comprobó el cumplimiento de los objetivos trazados durante las diferentes etapas de desarrollo de software.

- Se realizó un estudio de los sistemas existentes a nivel nacional e internacional identificando características que pudieron ser útiles para el desarrollo del trabajo.
- Se realizó un análisis de las tendencias, tecnologías y herramientas en el desarrollo de software de salud a nivel nacional e internacional que permitió corroborar como adecuada la arquitectura definida por el centro para el desarrollo de la capa.
- El modelado de cada uno de los flujos de trabajo propuestos por la metodología de desarrollo utilizada permitió conformar un expediente de proyecto que contiene todos los elementos del desarrollo que servirán de base para la continua evolución del producto.
- Al finalizar la primera iteración de implementación el sistema cuenta con funcionalidades que pueden ser accedidas desde cualquier sistema externo para consultar las terminologías disponibles en el servicio de manera efectiva.
- Al finalizar la primera fase de implementación se realizó de forma efectiva la integración de la Capa de Vocabulario con la Capa de Mensajería y el Sistema de Administración.

RECOMENDACIONES

Con el objetivo de mejorar la solución planteada se agregan las siguientes recomendaciones:

- Estudiar la especificación de CTS 2.0 de HL7 para identificar mejoras en las funcionalidades ya implementadas y nuevas funcionalidades que se pudieran incorporar a la solución actual.
- Se propone realizar una interfaz donde se visualicen datos, objetivos y descripciones de las funcionalidades que le permitan al usuario obtener información sobre estas.

REFERNCIAS BIBLIOGRÁFICAS

1. **Etcheverry, Graciela.** itaes.org. *GESTIÓN CLÍNICA: una herramienta para la gestión de la calidad en laboratorios clínicos.* [En línea] [Citado el: 18 de octubre de 2010.] http://www.itaes.org.ar/biblioteca/Gestionclinica.pdf.

2. *Curso de HL7Abierto a la comunidad iberoamericana.* [En línea] [Citado el: 18 de octubre de 2010.] http://www.telemedicina.buap.mx/Archivos/Estandar_3.pdf.

3. Ídem a referencia 2.

4. **Portolés Sánchez, MaJesús.** Proyecto de Fin de Carrera. *Búsqueda Semántica en Repositorios de Conceptos Biomédicos Estandarizados:C.T.Hunter.* [En línea] Universidad Politécnica de Valencia, 2010. [Citado el: 6 de noviembre de 2010.]

5. Ídem a referencia 2.

6. Ídem a referencia 4.

7. Informática y Salud. *Innovación y Salud.* [En línea] IX Jornadas de Informática Sanitaria en Andalucía, 2003. [Citado el: 21 de octubre de 2010.] http://www.conganat.org/seis/is/is42/andaluciamesa6.htm.

8. **Díaz Espinoza, Diego.** *ESTÁNDARES DE COMUNICACIÓN EN TECNOLOGíAS DE LA INFORMACIÓN EN SALUD.* Chile : Departamento de Ciencias de la Computación, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, 2008.

9. Ídem a referencia 4.

10. Ídem a referencia 4.

11. svmsoft.com. [En línea] svmsoft. [Citado el: 2 de noviembre de 2010.] http://svmsoft.com/index.php?option=com_content&view=article&id=47:medicalterms&catid=34:medicalapps&Itemid=54.

12. **Kpiachem A. S. Ivan, MD.** medical-dictionary. [En línea] [Citado el: 3 de noviembre de 2010.] http://www.medical-dictionary.ro/medical-dictionary-overview.html.

13. www.gib.fi.upm.es Servidor de vocabulario y ontologías [En línea] Grupo de Informática Biomédica [Citado el: 5 de noviembre de 2010.] http://www.gib.fi.upm.es/es/node/29

14. WorldLingo Translations LLC. [En línea] worldlingo.com. [Citado el: 5 de noviembre de 2010.] http://www.worldlingo.com/ma/enwiki/es/SNOMED_CT.

15. [En línea] NCI caBIG® Knowledge Center Web Infrastructure, 1 de marzo de 2011. [Citado el: 6 de noviembre de 2010.] https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexBig_and_LexEVS.

16. Ídem a referencia 4.

17. [En línea] [Citado el: 21 de febrero,2011]

http://archivo.revistaesalud.com/index.php/revistaesalud/article/viewArticle/308/641

18. **Huacoto, Nancy E. Concha.** *Propuesta para implantar CMMI en una empresa con múltiples unidades desarrolladoras de software.* [En línea] UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS.PERÚ, 2005. [Citado el: 8 de noviembre de 2010.] http://sisbib.unmsm.edu.pe/bibvirtualdata/Tesis/Basic/concha_hn/concha_hn.pdf.

29. buenastareas. *Rup (Rational Unified Process) O Pu (Proceso Unificado).* [En línea] buenastareas.com. [Citado el: 10 de noviembre de 2010.] http://www.buenastareas.com/ensayos/Rup-Rational-Unified-Process-o-Pu/509647.html.

20.**Schmuller, Joseph.** www.scribd.com. [En línea] [Citado el: 18 de febrero de 2011.] http://www.scribd.com/doc/38392190/UML-en-24-Horas.

21. www.sparxsystems.com.ar. *Enterprise Architect - Herramienta de diseño UML.* [En línea] [Citado el: 21 de febrero de 2011.] http://www.sparxsystems.com.ar/products/ea.html.

22. scribd. [En línea] scribd.com. [Citado el: 17 de noviembre de 2010.] http://es.scribd.com/doc/36570462/postgreSQL-investigacion.

23. **Bastías, Luis.** www.dcc.uchile.cl. *Máquina Virtual de Java.* [En línea] [Citado el: 14 de febrero de 2011.] http://www.dcc.uchile.cl/~rbaeza/cursos/proyarq/lbastias/JVM.html#instruccion.

24. *El lenguaje de Programación Java.* [En línea] [Citado el: 2 de febrero de 2011.] http://bill.cl/libros/programacion/java/El_Lenguaje_de_Programacion_Java.pdf.

25. **Rivera Rivera, Leyla Dinora, Colato Rodríguez, Hugo Miguel y Tesorero, Nelson Antonio.** es.scribd.com. *Sistema WDS para la Administración remota deservidores.* [En línea] UNIVERSIDAD FRANCISCO GAVIDIA. [Citado el: 4 de febrero de 2011.] http://es.scribd.com/doc/51495744/8/Eclipse.

26. Welcome to Apache Axis2/Java [En línea] [Citado el: 29 de enero de 2011.] http://axis.apache.org/axis2/java/core/index.html

27. Entorno Virtual de Aprendizaje. *Modelo_de_Dominio.* [En línea] [Citado el: 18 de noviembre de 2010.] http://eva.uci.cu/mod/resource/view.php?id=21011/Modelo_de_Dominio.doc.

28. clases3gingsof.wetpaint.com. *¿Qué es un requerimiento?* [En línea] [Citado el: 25 de enero de 2011.] http://clases3gingsof.wetpaint.com/page/%C2%BFQu%C3%A9+es+un+requerimiento%3F.

29. **MIZRAIM, CERVANTES RAMÍREZ ÁNGEL.** *INGENIERÍA DE SOFTWARE.* [En línea] UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO.FACULTAD DE INGENIERÍA., noviembre de 2008. [Citado el: 20 de noviembre de 2010.] http://es.scribd.com/doc/36878980/apuntesIngSW.

30. Ídem a referencia 29.

31. www.mastermagazine.info. *Definición de Casos de uso.* [En línea] [Citado el: 6 de febrero de 2011.] http://www.mastermagazine.info/termino/4184.php.

32. barranquillo.ucaldas.edu.co. *EL LENGUAJE DE MODELADO UNIFICADO UML, Presentación de UML.* [En línea] [Citado el: 8 de febrero de 2011.] http://barranquillo.ucaldas.edu.co/rgarcia/ingsoft/UML.htm.

33. *Diseno.* [En línea] Entorno Virtual de Aprendizaje . [Citado el: 16 de enero de 2011.] http://eva.uci.cu/mod/resource/view.php?id=14070.

34. **Caro Piñeres, Manuel Fernando y otros.** *DISEÑO DE SOFTWARE EDUCATIVO BASADO EN COMPETENCIAS.* [En línea] junio de 2009. [Citado el: 20 de enero de 2011.] http://www.umng.edu.co/www/resources/onj_articulo_5_19_1.pdf.

35. www.slideshare.net. *Diagramas de Interaccion de Objetos.* [En línea] [Citado el: 10 de febrero de 2011.] http://www.slideshare.net/javi2401/diagramas-de-interaccion-de-objetos-presentation.

36. www.MiTecnologico.com. [En línea] MiTecnologico. [Citado el: 28 de enero de 2011.] http://www.mitecnologico.com/Main/Dise%F1oDeModulos.

37. **Eusebio Rojas, Oswaldo.** *Rational Unified Process .* [En línea] [Citado el: 10 de febrero de 2011.] http://www.slideshare.net/genesyss/diseo-de-sistemas.

38. [En línea] [Citado el: 13 de febrero de 2011.] http://www.slideshare.net/wdavidslideshare/cap5-diseo-de-sistemas.

39. www.kybele.urjc.es, *Tema 9: Diseño de Software, Departamento de Lenguajes y Sistemas Informáticos II,* [En línea] Universidad Rey Juan Carlos [Citado el: 24 de marzo de 2011.] http://www.kybele.etsii.urjc.es/docencia/IS_GII_V/2010-2011/Material/GII-V-IS-2010-11.9.DisenoDeSoftware.pdf

40. *Tratamiento de excepciones* [En línea] [Citado el: 21 de febrero de 2011.] http://elvex.ugr.es/decsai/builder/intro/6.html

41. www.slideshare.net. *Presentacion Pruebas.* [En línea] [Citado el: 12 de febrero de 2011.] http://www.slideshare.net/dajigar/presentacion-pruebas-presentation.

42. Ídem a referencia 41.

BIBLIOGRAFÍA

- archives.postgresql.org. [En línea] PostgreSQL Global Development Group. [Citado el: 22 de febrero de 2011.] http://archives.postgresql.org/pgsql-es-fomento/2009-07/msg00000.php.
- barranquillo.ucaldas.edu.co. *EL LENGUAJE DE MODELADO UNIFICADO UML, Presentación de UML.* [En línea] [Citado el: 8 de febrero de 2011.] http://barranquillo.ucaldas.edu.co/rgarcia/ingsoft/UML.htm.
- **Bastías, Luis.** www.dcc.uchile.cl. *Máquina Virtual de Java.* [En línea] [Citado el: 14 de febrero de 2011.] http://www.dcc.uchile.cl/~rbaeza/cursos/proyarq/lbastias/JVM.html#instruccion.
- **Breis Fernandez, Jesualdo Tomas.** [En línea] Universidad de Murcia. [Citado el: 21 de enero de 2011.] http://webs.um.es/~%20jfernand/miwiki/doku.php?id=ofertapfc#servicios_de_explotacion_de_terminologias_clinicas_en_la_plataforma_archms.
- buenastareas. *Rup (Rational Unified Process) O Pu (Proceso Unificado).* [En línea] buenastareas.com. [Citado el: 10 de noviembre de 2010.] http://www.buenastareas.com/ensayos/Rup-Rational-Unified-Process-o-Pu/509647.html.
- C.E.L.I.T. S.A. [Citado el: 15 de noviembre de 2010.] http://www.trabajoline.com.ar/CursosOnLine/JavaPrincipiante/programacionJava1.htm.
- **Catalani, Exequiel.** *ARQUITECTURA Modelo/Vista/Controlador* [En línea] [Citado el: 16 de febrero de 2011.] http://exequielc.wordpress.com/2007/08/20/arquitectura-modelovistacontrolador/
- clases3gingsof.wetpaint.com. *¿Qué es un requerimiento?* [En línea] [Citado el: 25 de enero de 2011.] http://clases3gingsof.wetpaint.com/page/%C2%BFQu%C3%A9+es+un+requerimiento%3F.
- *Curso de HL7Abierto a la comunidad iberoamericana.* [En línea] [Citado el: 18 de octubre de 2010.] http://www.telemedicina.buap.mx/Archivos/Estandar_3.pdf.
- **Día, Oscar. 2008.** www.doctorsandmanagers.com. *Calidad e interoperabilidad entre los actores de un sistema de salud.* [En línea] doctors & managers, 2008. [Citado el: 13 de enero de 2011.] http://www.doctorsandmanagers.com/item.php?id=86&lang=1.
- **Díaz Espinoza, Diego.** *ESTÁNDARES DE COMUNICACIÓN EN TECNOLOGíAS DE LA INFORMACIÓN EN SALUD.* Chile : Departamento de Ciencias de la Computación, Facultad de Ciencias Físicas y Matemáticas, Universidad de Chile, 2008.
- *Diseno.* [En línea] Entorno Virtual de Aprendizaje . [Citado el: 16 de enero de 2011.] http://eva.uci.cu/mod/resource/view.php?id=14070.
- *El lenguaje de Programación Java.* [En línea] [Citado el: 2 de febrero de 2011.] http://bill.cl/libros/programacion/java/El_Lenguaje_de_Programacion_Java.pdf.
- Entorno Virtual de Aprendizaje. *Modelo_de_Dominio.* [En línea] [Citado el: 18 de noviembre de 2010.] http://eva.uci.cu/mod/resource/view.php?id=21011/Modelo_de_Dominio.doc.

- **Etcheverry, Graciela.** itaes.org. *GESTIÓN CLÍNICA: una herramienta para la gestión de la calidad en laboratorios clínicos.* [En línea] [Citado el: 18 de octubre de 2010.] http://www.itaes.org.ar/biblioteca/Gestionclinica.pdf.
- **Eusebio Rojas, Oswaldo.** *Rational Unified Process .* [En línea] [Citado el: 10 de febrero de 2011.] http://www.slideshare.net/genesyss/diseo-de-sistemas.

  http://archivo.revistaesalud.com/index.php/revistaesalud/article/viewArticle/308/641
- **Huacoto, Nancy E. Concha.** *Propuesta para implantar CMMI en una empresa con múltiples unidades desarrolladoras de software.* [En línea] UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS.PERÚ, 2005. [Citado el: 8 de noviembre de 2010.] http://sisbib.unmsm.edu.pe/bibvirtualdata/Tesis/Basic/concha_hn/concha_hn.pdf.
- Informática y Salud. *Innovación y Salud.* [En línea] IX Jornadas de Informática Sanitaria en Andalucía, 2003. [Citado el: 21 de octubre de 2010.] http://www.conganat.org/seis/is/is42/andaluciamesa6.htm.
- **Kpiachem,Ivan.** medical-dictionary. [En línea] [Citado el: 3 de noviembre de 2010.] http://www.medical-dictionary.ro/medical-dictionary-overview.html.
- kubuntu-es.org. [Citado el: 17 de noviembre de 2010.] http://www.kubuntu-es.org/wiki/desarrollo-programacion/programas-desarrollo-libres.
- **Caro Piñeres, Manuel Fernando y otros.** *DISEÑO DE SOFTWARE EDUCATIVO BASADO EN COMPETENCIAS.* [En línea] junio de 2009. [Citado el: 20 de enero de 2011.] http://www.umng.edu.co/www/resources/onj_articulo_5_19_1.pdf.
- **MIZRAIM, CERVANTES RAMÍREZ ÁNGEL.** *INGENIERÍA DE SOFTWARE.* [En línea] UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO.FACULTAD DE INGENIERÍA., noviembre de 2008. [Citado el: 20 de noviembre de 2010.] http://es.scribd.com/doc/36878980/apuntesIngSW.
- NCI caBIG® Knowledge Center Web Infrastructure, 1 de marzo de 2011. [Citado el: 6 de noviembre de 2010.] https://cabig-kc.nci.nih.gov/Vocab/KC/index.php/LexBig_and_LexEVS.
- **Pakoz, Zuñiga.** mitecnologico. *Procesos De La Ingenieria De Requerimientos.* [En línea] Instituto Tecnologico de Villahermosa. [Citado el: 22 de enero de 2011.] http://www.mitecnologico.com/Main/ProcesosDeLaIngenieriaDeRequerimientos.
- **Portolés Sánchez, MaJesús.** Proyecto de Fin de Carrera. *Búsqueda Semántica en Repositorios de Conceptos Biomédicos Estandarizados:C.T.Hunter.* [En línea] Universidad Politécnica de Valencia, 2010. [Citado el: 6 de noviembre de 2010.]
- **Rivera Rivera, Leyla Dinora, Colato Rodríguez, Hugo Miguel y Tesorero, Nelson Antonio.** es.scribd.com. *Sistema WDS para la Administración remota deservidores.* [En línea] UNIVERSIDAD FRANCISCO GAVIDIA. [Citado el: 4 de febrero de 2011.] http://es.scribd.com/doc/51495744/8/Eclipse.

- **Schmuller, Joseph.** www.scribd.com. [En línea] [Citado el: 18 de febrero de 2011.] http://www.scribd.com/doc/38392190/UML-en-24-Horas.
- scribd. [En línea] scribd.com. [Citado el: 17 de noviembre de 2010.] http://es.scribd.com/doc/36570462/postgreSQL-investigacion.
- **Stusser Beltranena, Rodolfo J. y Rodríguez Díaz, Alfredo .** bvs.sld.cu. *La informatización de la atención primaria de salud.* [En línea] [Citado el: 7 de diciembre de 2011.] http://bvs.sld.cu/revistas/mgi/vol22_4_06/mgi12406.htm.
- svmsoft.com. [En línea] svmsoft. [Citado el: 2 de noviembre de 2010.] http://svmsoft.com/index.php?option=com_content&view=article&id=47:medicalterms&catid=34:medicalapps&Itemid=54.
- synergix.wordpress.com . *Modelo de Dominio.* [En línea] [Citado el: 28 de enero de 2011.] http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.
- *Tratamiento de excepciones* [En línea] [Citado el: 21 de febrero de 2011.] http://elvex.ugr.es/decsai/builder/intro/6.html
- Welcome to Apache Axis2/Java [En línea] [Citado el: 29 de enero de 2011.] http://axis.apache.org/axis2/java/core/index.html
- WorldLingo Translations LLC. [En línea] worldlingo.com. [Citado el: 5 de noviembre de 2010.] http://www.worldlingo.com/ma/enwiki/es/SNOMED_CT.
- www.datacentersystems.org. *Base de Datos SQL PostgreSQL.* [En línea] [Citado el: 23 de febrero de 2011.] http://www.datacentersystems.org/datacenter/cmsms/index.php?page=base-de-datos-sql-postgresql.
- www.european-hospital.com. *LA GESTIÓN CLÍNICA: UNA PRIORIDAD PARA LA SANIDAD EN ESPAÑA.* [En línea] European hospital. [Citado el: 7 de enero de 2011.] http://www.european-hospital.com/en/article/2293.html.
- www.gib.fi.upm.es Servidor de vocabulario y ontologías [En línea] Grupo de Informática Biomédica [Citado el: 5 de noviembre de 2010.] http://www.gib.fi.upm.es/es/node/29
- www.kybele.urjc.es, *Tema 9: Diseño de Software, Departamento de Lenguajes y Sistemas Informáticos II,* [En línea] Universidad Rey Juan Carlos [Citado el: 24 de marzo de 2011.] http://www.kybele.etsii.urjc.es/docencia/IS_GII_V/2010-2011/Material/GII-V-IS-2010-11.9.DisenoDeSoftware.pdf
- www.mastermagazine.info. *Definición de Casos de uso.* [En línea] [Citado el: 6 de febrero de 2011.] http://www.mastermagazine.info/termino/4184.php.
- www.MiTecnologico.com. [En línea] MiTecnologico. [Citado el: 28 de enero de 2011.] http://www.mitecnologico.com/Main/Dise%F1oDeModulos.
- www.slideshare.net [Citado el: 13 de febrero de 2011.] http://www.slideshare.net/wdavidslideshare/cap5-diseo-de-sistemas.

- www.slideshare.net. *Diagramas de Interaccion de Objetos.* [En línea] [Citado el: 10 de febrero de 2011.] http://www.slideshare.net/javi2401/diagramas-de-interaccion-de-objetos-presentation.
- www.slideshare.net. *Presentacion Pruebas.* [En línea] [Citado el: 12 de febrero de 2011.] http://www.slideshare.net/dajigar/presentacion-pruebas-presentation.
- www.sparxsystems.com.ar. *Enterprise Architect - Herramienta de diseño UML.* [En línea] [Citado el: 21 de febrero de 2011.] http://www.sparxsystems.com.ar/products/ea.html.

GLOSARIO DE TÉRMINOS

ANSI: Instituto Nacional de Normas de Estados Unidos.

API: Interfaz de Programación de Aplicaciones (Application Programming Interface), es un conjunto de funciones y procedimientos que ofrece una cierta biblioteca para ser utilizado por otro software.

BD: Base de Datos.

CAP: Colegio de Patólogos Americanos (College of American Pathologists).

CESIM: Centro de Informática Médica.

CTS: Servicio de Terminologías Médicas.

GO: Gene Ontology.

HCE: Historia Clínica Electrónica. Se define como el conjunto de documentos surgidos de la relación entre el médico y el paciente y, a partir de la segunda mitad del siglo XX, como el registro de la relación que se establece entre los usuarios y el hospital o la atención primaria. La HC se considera como el único documento válido desde los puntos de vista clínico y legal a todos los niveles de atención en salud.

HGNC: Human Genome Nomenclature Consortium.

HL7: Nivel 7 de salud, organización que desarrolla especificaciones de estándares para el intercambio electrónico de información médica.

ISO: Organización Internacional de Estándares (International Organization for Standarization).

OOP: Programación Orientada a Objeto.

RIM: Modelo de Información de Referencia.

RUP: Proceso Unificado de Desarrollo de Software.

SNOMED: terminología que abarca un amplio espectro del dominio clínico.

TIE: Departamento de Tecnología, Integración y Estándares.

UCI: Universidad de las Ciencias Informáticas.

UML: Lenguaje unificado de modelado.

GLOSARIO DE TÉRMINOS

UMLS: Unified Medical Language System. Es un proyecto que se ocupa de reunir las terminologías médicas más utilizadas, enlazarlas y distribuirlas, con el objetivo de contribuir al desarrollo de herramientas capaces de sacar partido al conocimiento formalizados en ella.

XML: Permite definir la gramática de lenguajes específicos. Es una manera de definir lenguajes para diferentes necesidades. Se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas.