UNIVERSIDAD DE MURCIA

PROYECTO INFORMÁTICO

Adquisición de conocimiento y visualización para abstracción temporal. Una aplicación a medicina y biología.

Autora:
Mª Esperanza Velázquez Castillo

Directores:
Manuel Campos Martínez
José Manuel Juárez Herrero

Resumen

En las sesiones de diagnóstico de las Unidades de Cuidados Intensivos (UCI) es importante hacer un resumen visual de la información temporal de los pacientes. Esta información temporal tiene una naturaleza y unos modos de visualización propios (series, secuencias e historias). Sin embargo, es necesaria una representación unificada para facilitar las tareas inteligentes que hacen uso de esta información.

El problema de resumir la información temporal ya ha sido abordado en trabajos previos a este proyecto dentro del grupo de investigación AIKE, aportando como solución un módulo de abstracción como parte del sistema de ayuda, donde se implementan distintos mecanismos de abstracción.

El propósito de las técnicas de abstracción temporal es la abstracción de conceptos y patrones de alto nivel a partir de colecciones de datos temporales. La abstracción temporal juega un rol importante en los sistemas de ayuda a la decisión clínicos. Estos sistemas deben ser capaces de crear resúmenes de alto nivel de los datos temporales del paciente de una forma temporalmente significativa.

La idea de desarrollar herramientas automáticas para extraer características relevantes de alto nivel a partir de conjuntos de datos marcados temporalmente es especialmente importante cuando las colecciones disponibles de datos crecen constantemente en tamaño. La capacidad de abstracción temporal de datos es particularmente interesante en entornos de monitorización de pacientes, como son las UCI. Sin embargo, las técnicas y herramientas aquí mostradas son genéricas, y aplicables a dominios como la cronobiología, donde se tratan series temporales procedentes también de la adquisición de datos sobre humanos o animales.

Por otra parte se identifican dos problemas importantes asociados a este tipo de sistemas: la adquisición del conocimiento empleado en el proceso de abstracción y la visualización de los datos temporales generados por las abstracciones temporales. Por tanto, éstos serán los problemas abordados a lo largo de este proyecto.

Tabla de contenido

Tabla de ilustraciones

1. Introducción y referencias históricas

Son numerosos los dominios en los que es necesario monitorizar una serie de parámetros que nos permiten conocer la evolución de algunas de las características de un sujeto o elemento. Como resultado de un proceso de adquisición de datos continuado se obtienen colecciones de datos que crecen constantemente en tamaño. Además de por su tamaño, la interpretación de esta colección de datos se puede hacer tediosa por la complejidad de éstos, ya que en cualquier caso pueden estar sujetos a incertidumbre e imprecisión. Así, la interpretación del estado de la variable monitorizada, en base a los datos extraídos, resulta un proceso complejo para el usuario. Este hecho repercute negativamente en el análisis e interpretación de los datos, pudiendo interferir en el establecimiento de un diagnóstico, tratamiento o evaluación de la gravedad de un paciente.

Por tanto, queda de manifiesto la necesidad de obtener, a partir de los datos originales, un resumen de los mismos. De este modo sería interesante que el usuario de estos datos tuviera a su disposición una colección de datos que sea un resumen de los obtenidos por el sistema de adquisición de datos, mediante el uso de un sistema de ayuda a la decisión clínico. La abstracción temporal, como parte esencial del sistema de ayuda, realiza la simplificación mediante un resumen de colecciones de datos temporales, generando datos a un nivel más alto donde es más fácil su interpretación, todo ello basado en un modelo de abstracción. Ejemplos de abstracción son, estados acerca de la presión de la sangre (normal, alto o bajo) y la tendencia del mismo (creciente, decreciente o estacionario) sobre intervalos de tiempo.

En este proyecto, se plantea la abstracción temporal como una parte importante del sistema de ayuda de forma que permita al usuario trabajar con un conjunto de datos de más alto nivel, siendo necesario completar al proceso de abstracción con otras funcionalidades que hagan este sistema de ayuda más eficiente y usable. Ello nos lleva a plantearnos dos problemas, (1) la adquisición de conocimiento como paso previo a la abstracción, donde se requiere una interacción por parte del usuario con la aplicación para definir la información temporal tal como conceptos, abstracciones y contextos, y (2) la posterior visualización de los datos generados por la abstracción temporal, de forma que le sea más fácil al usuario el interpretar los datos, y que le permita conocer qué relación hay entre los datos temporales obtenidos del sistema de adquisición de datos y la colección de datos temporales resultado del proceso de abstracción.

En este proyecto se ha contemplado un paso más en la representación de los datos, así pues la colección de datos temporales producto de la abstracción de los datos obtenidos por el proceso de adquisición de datos se pueden llevar a un nivel aún más alto que simplifique su representación visual. De esta colección de datos podemos hacer agrupaciones o subconjuntos, ya sea para diferenciar la evolución de un concepto temporal, o para visualizar un conjunto de conceptos temporales donde su marca de tiempo depende de otros conceptos, plasmando visualmente la relación que se establece. Con este propósito se han definido una serie de estructuras temporales, vinculadas en gran parte al dominio de los datos que contiene.

Los resultados obtenidos en este proyecto pueden ser útiles en el campo de la medicina para al ayuda de adquisición de datos como, por ejemplo, la UCI, Unidad de

Cuidados Coronarios Intensivos (UCCI), Unidades de Cuidados Postoperatorios, etc. En estos entornos, los expertos manejan grandes volúmenes de datos clínicos evolutivos continuamente, los cuales tienen asociada una connotación temporal. En concreto podría plantearse su aplicación en el ámbito de las hojas de enfermería, donde se registra la evolución en el tiempo de una serie de parámetros que están siendo monitorizados a un paciente. Esta información podrá ser utilizada por distintos usuarios, que deben de interpretarla para la toma de decisiones, por lo que será necesario que los datos sean claros para realizar una evaluación correcta y rápida, ya que el tiempo es importante. Por tanto, parece lógico pensar que el uso de esta versión resumida de los datos temporales, será de ayuda en tiempo y dinero.

1.2. Contexto del trabajo

Este proyecto fin de carrera se enmarca dentro de una serie de proyectos más generales en los que colabora el grupo de investigación AIKE (Artificial Inteligence and Knowledge Engineering) de la Universidad de Murcia para el desarrollo de herramientas y sistemas inteligentes en dominios como el clínico y la cronobiología. Concretamente estos proyectos son:

- Proyecto "Herramientas inteligentes para el control de la calidad asistencial en procesos de seguimiento integrado intra/extrahospitalario de paciente" (TIN2006-15460-C04) del Ministerio de Ciencia y Tecnología. El objetivo principal de este proyecto es el diseño, desarrollo e implementación de un conjunto de herramientas orientadas al control de la calidad asistencial de los pacientes, que es contemplada desde una aproximación basada en guías de práctica clínica.

- Proyecto "Extensión temporal para el razonamiento basado en casos. Una aplicación al diagnóstico en medicina" (PET2007-0033) del Ministerio de Educación y Ciencia.

- Proyecto "Descubrimiento de conocimiento temporal para el establecimiento de marcadores de envejecimiento de los ritmos circadianos en humanos" (08853/PI/08). Este proyecto está financiado por la Fundación Séneca de la Agencia Regional de Ciencia y Tecnología de la Región de Murcia.

En los proyectos anteriores se propone el uso de distintas técnicas de abstracción con distintos objetivos. Así, en el primer y segundo proyecto las técnicas de abstracción se utilizan principalmente para modelado de conocimiento clínico y para visualización de datos complejos, mientras que en el tercer proyecto se usan como elemento fundamental en el preprocesamiento de series temporales con el fin de reducir su tamaño para hacer más eficiente el proceso de descubrimiento de conocimiento.

La propuesta aquí presentada surge como un elemento fundamental para superar uno de los cuellos de botella más importante en el desarrollo de sistemas inteligentes: la adquisición de conocimiento. Aunque el módulo de adquisición de conocimiento para abstracción temporal desarrollado en este proyecto está orientado principalmente al dominio clínico y cronobiológico, el módulo es reutilizable en otros dominios gracias a la definición genérica del modelo teórico subyacente, y a la utilización de las ontologías como formalismo para la descripción de los conceptos de dichos dominios.

El modelo de representación temporal extendido y las técnicas de visualización de información temporal también son un elemento a destacar para su integración en los resultado de los proyectos anteriormente mencionados.

1.3. Referencias comentadas

En esta línea de trabajo, donde interviene la abstracción temporal, encontramos algunos trabajos previos como el modelo de Cheung y Stephanopoulos [Cheung, 1990], que transforma registros de señal en descripciones de tendencia. Representa la evolución de un parámetro mediante su correspondiente historia cualitativa, contenedora de una secuencia de episodios cualitativos. Su principal inconveniente es su excesivo carácter cualitativo, además de que no propone ninguna solución para adquirir el conocimiento experto.

Por otra parte, el modelo de Haimowitz [Haimowitz, 1995] presenta una técnica para identificar patrones sobre la evolución de un conjunto de parámetros mediante un modelo de regresión polinómica, lo cual dificulta la elicitación del conocimiento, requiriendo por parte del experto el conocimiento de los polinomios.

En el modelo de Drakopoulos [Drakopoulos, 1998] para reconocimiento de patrones multivariable, los autores reconocen que el proceso de elicitación se hace muy tedioso por la complejidad del lenguaje propuesto. Se propone una interfaz gráfica de adquisición, basada en los dibujos realizados por el usuario, de la cual no consta implementación.

En el modelo de Steimann [Steimann, 1996] para la identificación automática de tendencias sobre una señal unidimensional, la adquisición de tendencias se realiza a partir de un experto que indica dos intervalos para definir el conjunto borroso trapezoidal. Presenta el inconveniente de que la adquisición del conocimiento se realiza de un modo muy ad hoc, dando una solución muy específica para el problema.

El Modelo de Lowe [Lowe, 1999] para la representación de patrones multivariable inspirándose en las propuestas de Steimann y de Haimowitz, tiene el inconveniente de no proporcionar herramientas para comprobar la consistencia del conocimiento y no hace una propuesta para la adquisición de conocimiento.

En el modelo de Otero [Otero, 2005] de restricciones borrosas para la abstracción y el reconocimiento sobre señales, se plasma mediante una representación computable el conocimiento y realiza razonamiento en lo referente a la evolución de un conjunto de parámetros. Es capaz de representar y gestionar la vaguedad de incertidumbre propia del conocimiento, así como la imprecisión y ruido de los datos. Facilita la tarea adquisición y validación de conocimiento mediante la herramienta TRACE, permitiendo al usuario el modelado de perfiles. No restringe el sistema de estudio, ejemplificando su aplicación al dominio médico y de la robótica móvil.

Entre estos modelos destaca la interfaz inteligente desarrollada por Shahar [Shahar, 2004] para obtener, visualizar y explorar bases de datos clínicas orientadas a datos temporales. Esta interfaz se llama KNAVE (Knowledge-based navigation of abstractions for visualization and explanation) de la que hay un segundo prototipo, KNAVE-II. Entre sus características destaca que se implementa un servicio para que al comienzo de una sesión de exploración el usuario pueda establecer una conexión remota con la base de datos de conocimiento deseada. Además, la interfaz de visualización

muestra a la vez un browser del conocimiento y unos paneles con datos obtenidos de la base de conocimiento o resultado de la abstracción. Permite explorar los datos temporales a diferentes niveles de granularidad. Fomenta la colaboración mediante el almacenamiento de los datos en un archivo de formato especial, para posteriormente poder ser enviado.

2. Análisis de objetivos y metodología

En esta sección se proponen los objetivos a alcanzar para el desarrollo del proyecto, partiendo de un objetivo principal del cual se derivan un conjunto de objetivos. Del análisis de dichos objetivos se extraen las distintas tareas que son necesarias realizar para su consecución.

A partir de las tareas a realizar habrá que plantear qué herramientas son necesarias y adecuadas para su ejecución. Además, se comenta la metodología a seguir a lo largo del desarrollo del proyecto, así como una planificación de cómo se secuenciarán las tareas en el tiempo.

2.1. Objetivos

La abstracción temporal surge como ayuda a la interpretación de datos de un paciente, en el contexto de la resolución de problemas médicos. De ello se deriva el objetivo general de este proyecto, "desarrollo de un entorno de adquisición de conocimiento y visualización de abstracciones temporales", ya que como paso previo será necesario un proceso de adquisición de conocimiento, el cual será utilizado durante el proceso de abstracción. La ontología nos permite representar este conocimiento, como pueden ser relaciones de abstracción entre conceptos del dominio y conocimiento temporal.

Posteriormente a la abstracción, se ha propuesto en este proyecto la visualización de los datos obtenidos como complemento a la finalidad de la abstracción temporal, que es la ayuda a la interpretación de esos datos. Así como el módulo de abstracción temporal con el que se integra este proyecto, pese a estar orientados para su uso en un dominio clínico y biológico, por estar enmarcados en los proyectos anteriormente mencionados, se ha desarrollado para poder ser utilizado para cualquier dominio en el que la adquisición, abstracción y/o visualización de datos temporales sea relevante.

En particular, este proyecto se desglosa en los siguientes objetivos:

O1: Creación de un módulo de adquisición de conocimiento, relativo a la información que contendrá la ontología.

O2: Creación de un modelo extendido de gestión de información temporal

O3: Creación de un módulo de visualización de información temporal.

O4: Integración de los resultados obtenidos en un prototipo que haga uso de datos de dominio médico.

No se incluye en los objetivos de este proyecto, por haber sido ya objeto, y por tanto resuelto, en otros estudios:

- Definición de la ontología del dominio de abstracción.
- Creación de un módulo de abstracción temporal.
- Realización de una Red de Restricciones Temporales Borrosas (FTCN).

Además, queda fuera de los objetivos de este proyecto la integración del módulo de

adquisición con el servidor de ontologías.

La comunicación con los distintos módulos se realizará de forma local, usando bases de conocimiento basadas en XML. En el caso del módulo de visualización de abstracciones temporales sí se ha implementado en el módulo de comunicación la posibilidad de la interacción con una base de datos.

2.2. Tareas

De los objetivos anteriormente definidos se derivan una serie de tareas que permitan su consecución:

T1: Revisión bibliográfica.

T2: Diseñar e implementar el módulo de comunicación con la base de conocimiento contenedora de las ontologías de dominio.

T3: Diseñar e implementar el módulo de adquisición de conocimiento, relativo a la información que contendrá la ontología.

T4: Diseñar e implementar un módulo de comunicación con el sistema de adquisición de datos para la entrada de datos temporales, objeto de visualización y/o abstracción.

T5: Diseñar e implementar un modelo temporal extendido que cree una representación estructurada de la información temporal.

T6: Diseñar e implementar un módulo de visualización de datos temporales.

T7: Integración del módulo de adquisición de conocimiento, el modelo temporal extendido, el módulo de visualización de abstracciones temporales y el módulo de abstracción temporal.

T8: Documentación del proyecto.

Aunque esta última tarea se realiza a lo largo de todo el proyecto, es más intenso una vez finalizadas todas las tareas anteriores.

Como se verá en la planificación del proyecto, cada una de las tareas donde se realiza un proceso de diseño e implementación, podemos identificar tres subtareas: diseño, implementación y pruebas.

2.3. Herramientas y librerías

En esta sección se indican las herramientas y librerías usadas para el desarrollo de este proyecto, así como el por qué de su elección. Se identifican dos grupos de herramientas: de desarrollo y librerías. Las herramientas de desarrollo utilizas son el entorno de programación Netbeans, Subversion, Log4J, XML y UML. Las librerías utilizadas han sido JGraph y HSQLDB.

Además se han utilizado otras 2 librerías, FuzzyTime y temporalAbstraction. Estas librerías son la implementación de la red temporal borrosa y el modelo de abstracción respectivamente. Serán comentadas más adelante en esta sección y posteriormente, en la sección 3.2.8 (Diseño) se mostrará la integración con estas librerías.

2.3.1. Tecnología Java

Al elegir el lenguaje a usar en la implementación del proyecto, se ha tenido en cuenta que sea un lenguaje orientado a objetos, característica que hace que grandes proyectos sean más fáciles de gestionar. Java, como lenguaje orientado a objetos permite generar un software de calidad, reutilizable y extensible.

Algunas de las características de Java que han llevado a elegirlo como lenguaje para el desarrollo de este proyecto son:

- Simple: es un lenguaje potente que elimina características poco usadas y confusas presentes en otros lenguajes. Mediante el garbage collector evita al programador el preocuparse por liberar la memoria, eliminando también el tener que trabajar con punteros.
- Robusto: realiza verificaciones tanto en tiempo de compilación como en tiempo de ejecución, ayudando a la detección de errores. Para asegurar el funcionamiento de la aplicación realiza una verificación de los byte-codes, comprobación de punteros, comprobación de límites de arrays y excepciones.
- Portable: además de la portabilidad básica, Java implementa otros estándares de portabilidad. Construye sus interfaces de usuario a través de un sistema abstracto de ventanas, pudiendo ser implantadas en entornos Unix, Pc o Mac.
- Interpretado: El intérprete de Java puede ejecutar directamente código objeto.
- Dinámico: No conecta los módulos que comprenden la aplicación hasta el tiempo de ejecución, de forma que librerías nuevas o actualizadas no detienen la aplicación.

Además Java posee otras características, multithread, seguro y distribuido, pero que no son relevantes para este proyecto.

La plataforma donde se ejecutará la aplicación no se prevé que sea una en concreto, por lo que un lenguaje que nos limitase la plataforma podría suponer problemas de implantación. El lenguaje Java es multiplataforma, pudiendo ejecutarse en cualquier equipo siempre que contenga la máquina virtual de Java.

2.3.2. XML

XML (Extensible Markup Lenguaje) es un metalenguaje de marcado estándar recomendado por el World Wide Web Consortium. Hablamos de metalenguaje ya que permite definir otros lenguajes mediante etiquetas, donde por la característica de Extensible que tiene XML las etiquetas definidas pueden ser de número ilimitado, fue diseñado para contener datos y no para mostrarlos. Da la confianza de un lenguaje asentado y ampliamente aceptado, ya que su versión 1.0 data de 1998. Juega un rol importante en el intercambio de una amplia variedad de datos en la Web y otros campos.

En el marco de nuestro proyecto, XML es la herramienta para almacenar tanto Ontologías como datos temporales de forma estructurada, haciendo estos datos de fácil acceso, pues tiene un formato de texto muy flexible, cualquier software que pueda tratar texto plano podrá acceder a la información que transporta el documento XML. Puesto que las etiquetas no están predefinidas, permite definir nuestras propias etiquetas

adaptando la estructura del documento a la naturaleza de nuestros datos, conteniendo un escaso número de reglas sintácticas para crear documentos. Ya que para el desarrollo del proyecto se ha elegido el lenguaje Java el cual es independiente de la plataforma, XML no impide la portabilidad de la aplicación, ya que también es independiente de la plataforma.

DTD y XML esquema

Un documento XML será bien formado si cumple con la especificación XML de producción. Sin embargo será válido si el usuario define mediante un DTD o esquema una gramática que ha de cumplir el documento XML. Tanto un DTD como un XML Schema lo que realizan es una comprobación de la integridad de los datos del documento XML, es decir, comprueba si el contenido coincide con lo que ha de tener.

El DTD es un documento de texto que recoge la estructura lógica o gramática de un documento XML. Define qué elementos, relaciones entre ellos, atributos y entidades aparecen en el archivo XML.

El XML Schema surge para evitar el uso de la sintaxis especial de DTD, que aunque puede llegar a ser algo más complicado evita tener que aprender una sintaxis particular. Son estructuras más potentes y expresivas que los DTD, ya que permiten especificar el contenido de los documentos en función del tipo de datos empleado.

2.3.3. HSQLDB

Es un gestor de base de datos relacional escrito en Java y de código abierto, lo cual da la posibilidad a la aplicación de interaccionar con una base de datos de forma rápida y sencilla. Las tablas en él creadas pueden hacerse persistentes tanto en disco como en un archivo de texto.

Sin necesidad de iniciar un servidor, se puede hacer que una aplicación Java se conecte a una base de datos HSQLDB, aunque realmente se esté conectando a un archivo de texto. Se puede acceder a la base de datos para crear tablas, insertar datos o recuperarlos. La interacción con la base de datos no se hará como con un fichero común, sino mediante las clases proporcionadas por la librería java.sql: `Connection`, `Statement` y `Resulset`, siempre usando el lenguaje SQL estándar. Este modo de almacenamiento de la base de datos es muy útil para aplicaciones portables. Se puede crear una aplicación con una base de datos incrustada sin necesidad de instalar ninguna librería.

Para conectar a la base de datos se utiliza el controlador JDBC, con el que se para acceder a la base de datos. Es decir, en nuestra aplicación usaremos el mismo driver `org.hsqldb.jdbcDriver` y el url que usaremos será `jdbc:hsqldb:file:`[ruta a la base de datos].

La gestión de la base de datos permite las operaciones habituales, soportando la sintaxis SQL estándar, haciendo flexible las operaciones con la base de datos.

Se ha optado por esta solución basada en una base de datos local, ya que simplifica el desarrollo. Por una parte es más fácil la creación y modificación de las tablas de la base de datos ya que todo el trabajo se realiza en local. Por otra parte, incide

en la portabilidad de la aplicación ya que tiene la base de datos adherida, evitando los accesos a una base de datos externa. Sin embargo, para futuras versiones en las que se integre con otras herramientas lo adecuado sería trasladar la información de la base de datos local a una base de datos centralizada, accediendo por tanto de forma remota.

2.3.4. Log4j

Durante el proceso de desarrollo de la aplicación se hace necesario el uso de mecanismos de depuración del funcionamiento de la aplicación. El propio entorno de programación permite hacer depuración, aunque la librería open source log4j tiene otras características interesantes, como definir un nivel de prioridad, no en tiempo de compilación como la mayoría de los debuggers, sino en tiempo de ejecución editando los ficheros de configuración externos, para lo cual no es necesario modificar los binarios de la aplicación. Por tanto, se puede definir a criterio de programador que mensajes de log habilita o deshabilita, sin embargo con `System.out.println()` eliminar o filtrar los mensajes se hace muy complicado.

Para que una clase pueda usar este sistema de log, tendrá que contener un objeto estático denominado logger. Otra característica que distingue a esta librería es la capacidad de herencia entre loggers, ayudando así a reducir el volumen de salidas de log y minimizando su coste.

Otra ventaja de este sistema de log, es que modificando sólo el fichero de propiedades se puede cambiar el destino de la salida de log, pudiendo ser a archivo, un `OutputStream`, un `java.io.Writer`, un servidor remoto de log4j, un demonio Syslog remoto de Unix entre otros muchos posibles destinos.

2.3.5. JGraph

Para la implementación del módulo de visualización de información temporal, hay que buscar una librería para la representación de estas gráficas. Principalmente se dibujarán puntos, intervalos y relaciones entre cualquier par de éstos. Por lo tanto la librería que se utilice debe ser capaz de dibujar estas relaciones. Así pues, si analizamos una librería de representación de gráficas como JFreeChart, no se adecúa a las necesidades del módulo, ya que en cualquier caso debemos de poder visualizar los datos temporales como un grafo, no ofreciendo este tipo de librerías esa posibilidad.

La librería JGraph es una librería completa y potente que nos permite visualizar grafos, opensource y escrita en Java, siendo además totalmente compatible con los componentes Swing, pudiendo ejecutar en cualquier sistema que tenga Java (1.4 o posterior).

Aunque no son estrictamente necesarias, esta librería posee otras funcionalidades que abren el abanico de posibilidades que se le ofrecen al usuario haciendo la interfaz con el usuario más atractiva e intuitiva:

- Permite interacción con el grafo.
- Proporciona distintas distribuciones de los elementos del grafo.
- Proporciona otras funcionalidades como zoom, drag and drop, folding, undo, etc.

2.3.6. Entorno de programación NetBeans

El entorno de programación utilizado debe ser necesariamente compatible con Java, que ha sido el lenguaje elegido para implementar la aplicación. Son varios los entornos de programación que podemos encontrar para programar en Java, entre ellos NetBeans, entorno de desarrollo de software open-source que proporciona las herramientas necesarias para crear una aplicación de escritorio en lenguaje Java (entre otros).

NetBeans además de fácil de instalar y ejecutar, es multiplataforma. También posee otras características que lo han hecho adecuado para el desarrollo de este proyecto.

Puesto que gran parte de este proyecto es el diseño de la interfaz gráfica de usuario, esta herramienta se hace adecuada para el desarrollo ya que cuenta con una paleta de componentes Swing y AWT, simplificando el proceso de desarrollo, siendo sólo necesario arrastrar los componentes desde la paleta al canvas, tras lo que se generará el código automáticamente.

Otro punto importante de este entorno es la característica de control de versiones que se comenta a continuación.

Han sido considerados otros entornos de construcción de interfaces y de IDE, como JBuilder y Eclipse. Tras compararlos con NetBeans han sido descartados, ya que por una parte, JBuilder es un entorno comercial y no se dispone de presupuesto para este proyecto. Por otra parte, Eclipse pese a ser gratuito como NetBeans, necesita la instalación de plugins que son inestables.

2.3.7. Subversion

Subversion (svn) es un software de control de versiones, el cual presenta muchas mejoras frente a cvs. Una característica importante es que svn, no introduce un número de revisión independiente para cada archivo a diferencia de cvs. Realiza modificaciones atómicas del repositorio, enviando sólo las diferencias, mientras que cvs envía al servidor archivos completos. Se pueden bloquear archivos, para modificaciones exclusivas de un usuario al mismo tiempo. Permite seguir la historia de los archivos y carpetas, volviendo en caso de que sea necesario a revisiones anteriores. La creación de ramas y etiquetas es más eficiente puesto que tiene un costo de complejidad constante, siendo en cvs la complejidad lineal.

Ofrece varias interfaces, ya sea como un programa independiente o como parte integrada de un entorno de desarrollo. En este caso y puesto que nuestro entorno de desarrollo es NetBeans, como ya ha sido comentado, se tuvo en cuenta el que se pudiera integrar control de versiones, como diff, update y commit que ayuda en los típicos ciclos del desarrollo.

2.3.8. Lenguaje de modelado UML

UML es un lenguaje que permite modelar, construir y documentar los elementos de un sistema software orientado a objetos. UML ha sido ampliamente aceptado, ya que unifica la notación e incorpora las ventajas de cada uno de los métodos particulares en los que se basa. La notación y semántica unificada es tan importante ya que permite el desarrollo de modelos que se puedan intercambiar entre las distintas herramientas

CASE.

En este proyecto se usará UML para modelar mediante diagramas la arquitectura y funcionamiento de la aplicación. Ello permite que para un posterior uso de la aplicación se pueda hacer una comprensión rápida de la estructura de la aplicación, ya que puesto que lo que se implementan son distintos módulos será útil para hacer uso de su API.

En este proyecto, de entre los 13 tipos básicos de diagramas que hay en UML, se usarán diagramas de casos de uso y diagramas de clases. Mediante los diagramas de caso de uso se define el comportamiento del sistema.

2.3.9. FuzzyTime

Es un módulo de razonamiento temporal para la gestión de información heterogénea independiente del dominio. Está estructurado en tres capas, la capa superior es la interfaz de usuario, encargada de traducir los datos introducidos por el usuario en relaciones temporales, la capa intermedia o Mundo Temporal contiene dos tipos de entidades (puntos e intervalos) y relaciones temporales y la capa base es la FTCN independiente del anterior pero con una correspondencia con el mundo temporal. [Campos, 2007]

Permite realizar consultas sobre las relaciones y entidades temporales definidas, basándose en la Teoría de la Posibilidad.

En este proyecto se trabaja directamente con la capa intermedia de la librería, con el objetivo de definir variables temporales y relaciones entre ellas, dentro del Mundo Temporal.

2.3.10. TemporalAbstraction

En esta librería se implementa el modelo desarrollado en la sección 2.6 de este documento.

2.4. Metodología

Para el desarrollo del proyecto se ha seguido un modelo de prototipado incremental, de forma que tras una primera versión del prototipo, se ha ido retroalimentando para corregir funcionalidades y adicionando nuevas funcionalidades. Se ha realizado una validación de cada nueva versión del prototipo.

Ilustración 1: Ciclo de desarrollo

2.5. Planificación

En las ilustraciones se muestra la planificación del proyecto, donde incluyen las tareas que en el apartado 2.1 se han identificado como derivación de los objetivos planteados para el proyecto.

WBS	Nombre	Inicio	Fin	Duración
1	Revisión bibliográfica	oct 1	oct 27	21d
2	Diseño e implementación del módulo de comunicación con la base de conocimiento	oct 27	nov 11	11d
3	Diseño e implementación del módulo de adquisición de conocimiento	nov 11	dic 9	20d
4	Diseño e implementación del módulo de comunicación con sistema de monitorización	dic 9	ene 7	15d
5	Diseño e implementación del framework genérico gestión temporal	ene 7	feb 4	20d
6	Diseño e implementación del módulo de visualización	feb 4	abr 29	60d
7	Documentación	oct 1	jul 1	190d

Ilustración 2: Temporalización de las tareas del proyecto

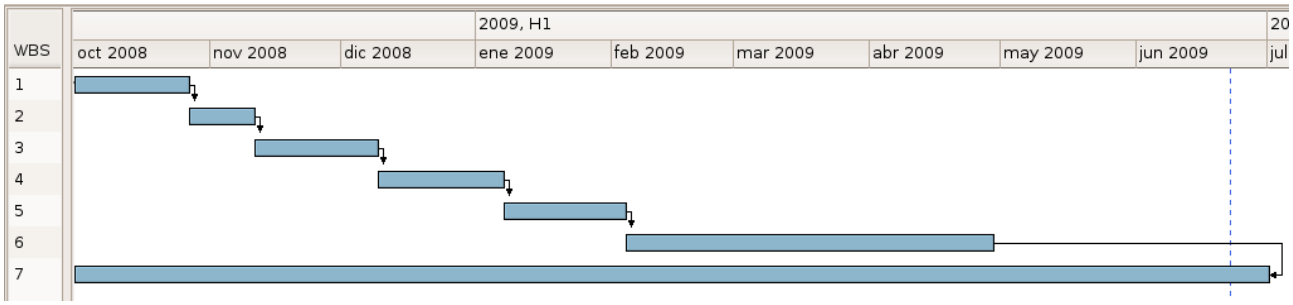


Ilustración 3: Diagrama de Gantt. Planificación

2.6. Modelo de Abstracción

A continuación se expone de manera breve el modelo de abstracción temporal sobre el cual se desarrolla el presente proyecto [Campos, 2007].

2.6.1. Entidades Conceptuales del Dominio

La abstracción temporal está basada en tres tipos de conceptos:

- Conceptos temporales: encontramos dos tipos, observables, cuya evolución temporal se obtiene directamente mediante la adquisición de una señal o por interacción con el usuario. El segundo tipo son parámetros, que se obtiene por derivación de los valores medidos por un observable u otro parámetro. La entidad parámetro es abstracta, por lo que una instancia será un Parámetro de Ocurrencia o un Parámetro de Estado.

  Cada subtipo de concepto temporal está relacionado con un subtipo de concepto histórico, formando dos jerarquías paralelas.

- Conceptos históricos: Describen la evolución temporal de los conceptos

temporales. Los tipos de conceptos históricos definidos son observación, ocurrencia, estado y evento. Una **observación** se obtiene al hacer una medición sobre un observable en un instante. El conjunto de observaciones que se obtienen para un observable se denomina historia. Una **Ocurrencia** es el resultado de abstraer una observación u otra ocurrencia, por lo que su valor también es para un instante discreto de tiempo. Un **Estado** define un intervalo de tiempo máximo durante el que un parámetro tiene un valor cualitativo constante. Se obtiene a partir de la historia de un concepto temporal. Un **Evento** representa un cambio significativo en la historia de estados en un instante de tiempo, pudiendo ser de comienzo o fin de un estado.

Los conceptos históricos reflejan una información temporal que en el caso de observaciones, ocurrencias y eventos harán referencia a un punto en el tiempo, mientras que los estados harán referencia a un intervalo de tiempo.

- Conceptos del dominio: proporcionan la estructura organizacional de las variables manejadas por el sistema. La semántica de los conceptos del dominio no influye en el proceso de abstracción, ya que es un proceso de propósito general. Para una correcta interacción con la ontología del dominio habrá que definir una cierta información como atributos de los conceptos del dominio: concepto, característica, dominio, persistencia, unidad de persistencia, granularidad y unidad de granularidad.

En la ilustración 4 se muestran las jerarquías de estos conceptos y sus subtipos.

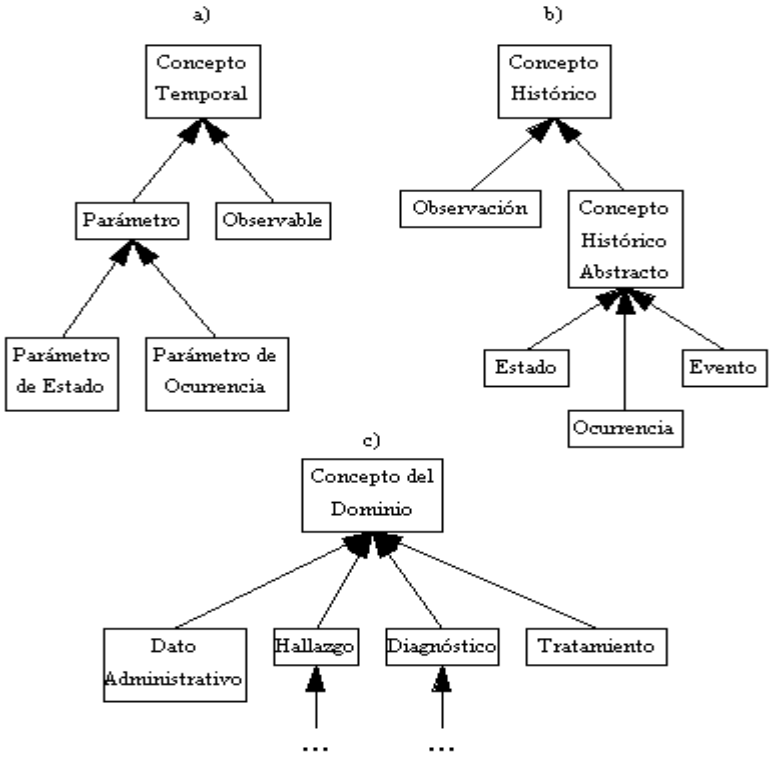


Ilustración 4: Ontologías del dominio

Si tenemos el concepto temporal *Fiebre:Temperatura*, éste contiene una historia con la evolución del concepto. Esta historia está compuesta por conceptos históricos que se corresponden con las mediciones de temperatura. El concepto del dominio temperatura define la información asociada, que será necesaria para el concepto

temporal.

2.6.2. Proceso de Abstracción Temporal

Se considera un proceso abductivo en vez del clásico problema de clasificación ya que no hay medidas densas, y bajo esa condición no se puede asegurar la corrección de la clasificación.

Este problema de abstracción temporal consiste en buscar la historia temporal, es decir, una explicación para las observaciones que sea consistente, cubra todos los datos y sea lo más simple posible.

Se pueden usar Redes de Restricciones Temporales (FTCN) para representar las relaciones temporales entre historias abstraídas y las historias medidas.

2.6.3. Mecanismos de Abstracción Temporal

Hay diferentes tipos de abstracción guiados por el conocimiento, característica que los diferencia de un simple análisis estadístico.

La abstracción atemporal es uno de los tipos más simples, donde el tiempo está implícito y se realiza un mapeo de un dato simple a uno abstracto. Así pues, en este tipo de abstracción el concepto abstraído hereda la referencia temporal del concepto del que se abstrae.

Ejemplos de este tipo son:

- Abstracción cualitativa: abstrae un parámetro en otro parámetro cualitativo abstracto, es decir, mapea los valores del parámetro origen en el parámetro cualitativo abstracto. Para ello se suele definir una tabla de mapeo.
- Abstracción por generalización: una instancia se mapea a una de sus clases. Se basa en taxonomías.
- Abstracción por definición: un dato de una categoría conceptual se mapea con otro dato que es su contrapartida en otra categoría conceptual.

La abstracción temporal implica abstracción de conceptos y abstracción temporal. Ejemplos de abstracción de datos temporales son:

- Abstracción de estado: su objetivo es derivar el intervalo maximal (con cierta granularidad) en el que no cambia el valor de un parámetro. En este caso, hay un conjunto de datos que comparten una propiedad concatenable.
- Abstracción de persistencia: su objetivo es derivar el intervalo maximal de alguna propiedad; para ello se mira hacia atrás en el instante de tiempo discreto en el que se toma la medida.
- Abstracción de tendencia: su objetivo es derivar cambios significativos y la velocidad de cambio en la progresión de algún parámetro. Encadena abstracción de estado y de persistencia para obtener el intervalo en el que el parámetro no sufre ningún cambio. Lo difícil es encontrar los puntos en los que se produce un cambio significativo y en qué dirección se produce el cambio.

Estos tipos de abstracción pueden ser combinados para definir otros tipos de

abstracción más complejos, aumentando así la expresividad del proceso de abstracción. Por tanto la abstracción compuesta tendrá una entrada y la salida de cada mecanismo de abstracción que la compone será la entrada del siguiente.

El orden en que se compongan los mecanismos de abstracción será vinculante con los resultados obtenidos, ya que ordenaciones diferentes generarán distintos resultados.

En la Ilustración 5 se muestra una abstracción de estado de temperatura a estado de temperatura cualitativa

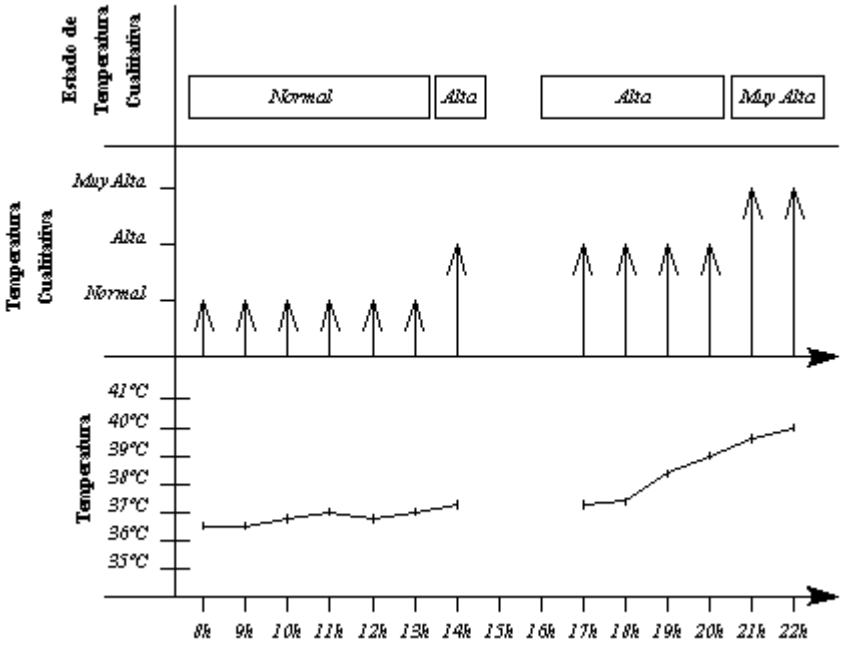


Ilustración 5: Ejemplo de abstracción de estado

2.6.4. Contextos de Abstracción

Las abstracciones temporales se definen únicamente en base al conocimiento del dominio. Sin embargo, podría ocurrir que ciertas abstracciones dependiesen de factores externos. Para ello se define un contexto como un conjunto de reglas que serán aplicables durante todo el proceso de abstracción y que por tanto alterarán el resultado de la abstracción.

Otra opción sería definir un intervalo de tiempo durante el cual el contexto es aplicable, pero ello entraña gran complejidad, por lo que no se contempla en la definición de este modelo de abstracción. Por tanto, suponemos que un contexto será aplicable durante todo el proceso de abstracción.

Por ejemplo, en el dominio de la abstracción clínica puede ocurrir que ciertas relaciones de abstracción dependan de características del paciente, como puede ser la edad o la condición de fumador. Véase el ejemplo de la Ilustración 6.

Generalización / Contexto fumador	No Fumador	Fumador
Alteración Respiratoria	NO	SI

Ilustración 6. Ejemplo de abstracción sensible al contexto

3. Diseño y resolución del trabajo realizado

En esta sección se hace un análisis de la aplicación en base a los casos de usos extraídos. Además se presenta el modelo conceptual, estructura de los módulos, persistencia del modelo, diseño de la base de datos loca, diseño de la base de conocimiento, diseño del sistema de adquisición de conocimiento, integración de los módulos desarrollados y un ejemplo del proceso de adquisición de conocimiento y visualización.

3.1. Casos de Uso

Al analizar el prototipo se han identificado tres casos de uso principales. Como se puede ver en el siguiente diagrama tenemos un caso de uso mediante el que se realiza el proceso de adquisición de conocimiento de la ontología, otro caso de uso realiza la abstracción y finalmente el tercer caso de uso que realiza la visualización de conceptos temporales abstraídos.

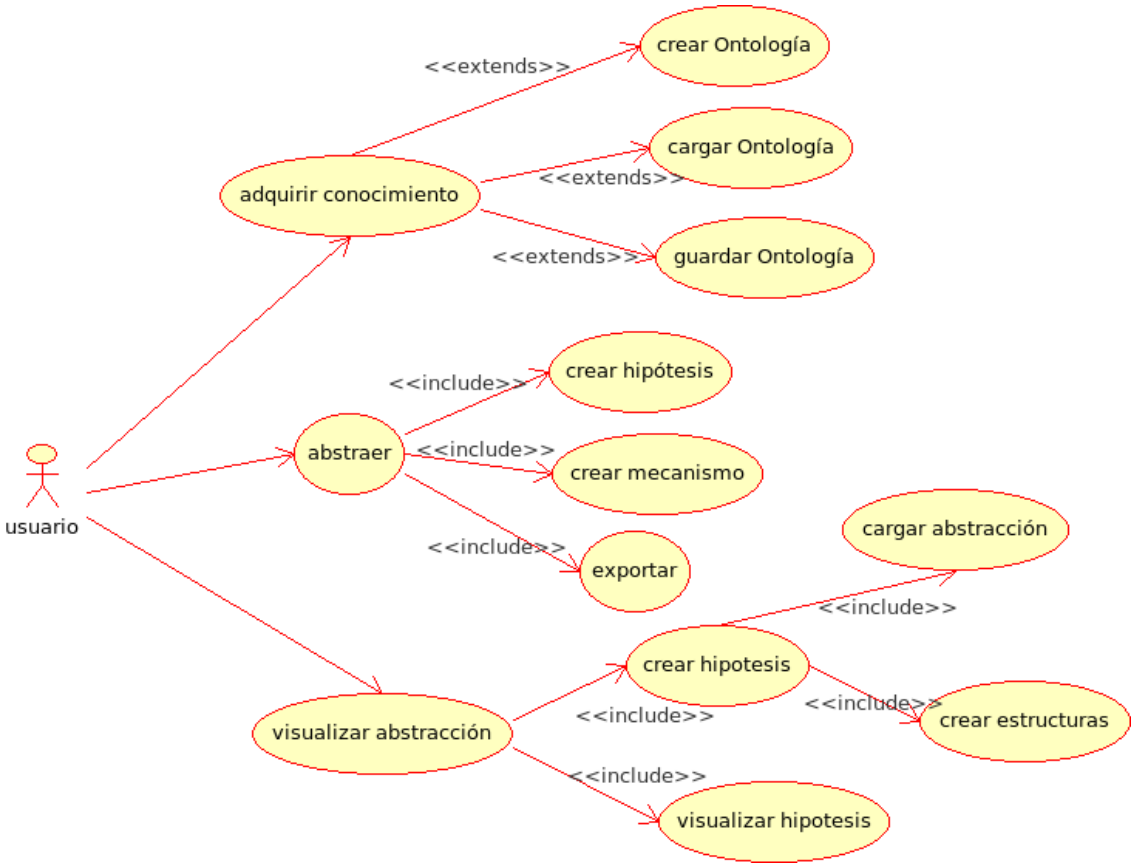


Ilustración 7: Diagrama de casos de uso de la aplicación

A continuación se desarrollan los principales casos de uso identificados en el diagrama anterior. Para ello se usa plantilla extraída de [Larman, 2005], indicando para cada uno su nombre, actor principal, descripción, precondiciones y escenarios principal y alternativo.

Adquirir conocimiento

Este caso de uso describe una de las principales líneas de trabajo de este proyecto, el proceso de adquisición de conocimiento. Se ha implementado a modo de asistente para guiar al usuario y ocultar la complejidad subyacente del módulo de adquisición, de forma que haga el proceso lo más fácil posible al usuario.

Descripción: Permite crear una nueva ontología y almacenarla en un documento XML

Actores: Usuario del prototipo

Precondiciones: Sólo debe haber iniciado la aplicación

Escenario principal

1) El usuario selecciona crear una nueva ontología
2) Definir Parámetros
    1. El usuario selecciona crear nuevo parámetro
    2. El usuario introduce los datos del parámetro
    3. El usuario guardar el parámetro
    4. vuelve al paso 1 para definir otro parámetro
    5. El usuario selecciona un parámetro
    6. El usuario añade el parámetro a la ontología
    7. vuelve al paso 5 para añadir otro parámetro
3) Definir Contextos
    1. El usuario selecciona crear nuevo contexto
    2. El usuario introduce los datos del contexto
    3. El usuario guarda el contexto
    4. vuelve al paso 1 para definir otro contexto
    5. El usuario selecciona un contexto
    6. El usuario añade el contexto a la ontología
    7. vuelve al paso 5 para añadir otro contexto
4) Definir Abstracciones
    1. El usuario selecciona la abstracción
    2. El usuario selecciona los parámetros sobre los que hace abstracción
    3. El usuario selecciona la función de abstracción
    4. Vuelve al paso 1 para añadir otra abstracción
5) Guardar Ontología
    1. El usuario selecciona guardar Ontología
    2. El usuario elige la ruta donde guardar la ontología

Escenario alternativo

1a El usuario selecciona abrir una ontología existente.

1. El sistema muestra los datos de la ontología

2.1a El usuario selecciona cargar parámetro

1. La aplicación muestra la lista de parámetros definidos actualizada

2.1b El usuario selecciona editar parámetro

2.1c El usuario selecciona copiar parámetro

2.1d El usuario selecciona borrar parámetro

3.1a El usuario selecciona cargar contextos

1. La aplicación muestra la lista de contextos definidos actualizada

5.1a El usuario selecciona "guardar como"

5.2a El sistema guarda la ontología en la ruta que tenía.

Abstraer

Descripción: Permite abstraer datos temporales.

Actores: Usuario del prototipo

Precondiciones: Debe haber una ontología abierta

Escenario principal

1) El usuario introduce el nombre a la hipótesis
2) Selecciona la ruta de la fuente de datos del sistema de adquisición de datos
3) Elegir el tipo de abstracción
4) El usuario inicia la abstracción
5) El sistema ejecuta la abstracción y almacena el resultado en un documento XML
6) El usuario elige visualizar la abstracción

Escenario Alternativo

6a El usuario elige no visualizar la abstracción

Visualizar abstracción

Descripción: Permite visualizar una abstracción.

Actores: Usuario del prototipo

Precondiciones: Debe haber abierta una ontología

Escenario principal

1) Seleccionar el origen de abstracción
   1. Selecciona el tipo de fuente (XML o bbdd)
   2. Selecciona la ruta de la fuente de datos

2) Definir como se estructuran los conceptos temporales en la visualización
    1. Seleccionar gráficas separadas

Escenario Alternativo

2.1a Seleccionar todas las gráficas en un grafo

2.1b Definir varios grafos

1. Seleccionar conceptos temporales a añadir al grafo
2. Añadir grafo
3. Volver al paso 1 tantas veces como grafos se quiera definir.

3.2. Modelo Conceptual

El modelo conceptual se divide en dos partes: estructuras temporales y relaciones temporales.

Entidades conceptuales (estructuras temporales)

Una estructura temporal será un objeto contenedor de uno o varios conceptos temporales objeto y/o fruto del proceso de abstracción, cuya representación va en función del dominio de dicha información temporal, puesto que está vinculado al tipo de estructura temporal, pudiendo ser una `Serie`, una `Secuencia` o un `Grafo`.

Estas estructuras las agrupamos en el concepto que hemos denominado modelo temporal extendido, donde se extiende el modelo de entidades conceptuales definidas en el modelo conceptual del módulo de abstracción. Como se muestra en la Ilustración 8, se extiende la hipótesis de abstracción que sigue estando formada por un contexto de abstracción, pero en lugar de contener una colección de conceptos temporales, la nueva hipótesis estará formada por una colección de estructuras temporales. Esta representación simplifica la visualización, al llevar a un nivel superior la representación que tenemos de los datos temporales generados por el proceso de abstracción.

Una estructura temporal independientemente del tipo que sea, siempre contendrá un conjunto de conceptos temporales y un identificador único. Además, tendremos distintas características propias de cada uno de los tres tipos de estructuras temporales. Una serie se caracteriza por contener un único concepto temporal, cuyo dominio será continuo, es decir, su dominio será `VDouble` o `VInteger`. En una secuencia, la lista de conceptos temporales también contendrá un único concepto temporal, siendo en este caso de dominio discreto, es decir, que podrá tomar los valores `VSymbol`, `VTrend`, `VGradient`, `VQualitative` y `VBoolean`. Un grafo se usa en el resto de casos, donde se instanciará este tipo de estructura si se desea visualizar más de un concepto temporal en una misma gráfica, con el objetivo de representar las relaciones temporales establecidas entre ambos conceptos. En un grafo lo que se pretende es contener conceptos históricos que carecen de una marca temporal y que su situación en el tiempo se infiere a partir de una relación temporal respecto a otro concepto. En cualquier caso, estos conceptos históricos podrán ser tanto puntos como intervalos.

Por ejemplo, en el dominio de la cronobiología el concepto temperatura cuyo dominio es `VDouble` se representaría como una estructura temporal de tipo serie, si realizamos una abstracción de estado a un concepto de de dominio `VQualitative` se representaría como una estructura temporal de tipo secuencia. Se podrían incluir ambos

conceptos en un mismo grafo, de forma que al visualizar se representasen las relaciones de abstracción entre ambos conceptos.

En la Ilustración 8 se puede ver cómo se integra el modelo definido en el modelo temporal extendido con el que ya estaba definido en el módulo de abstracción.

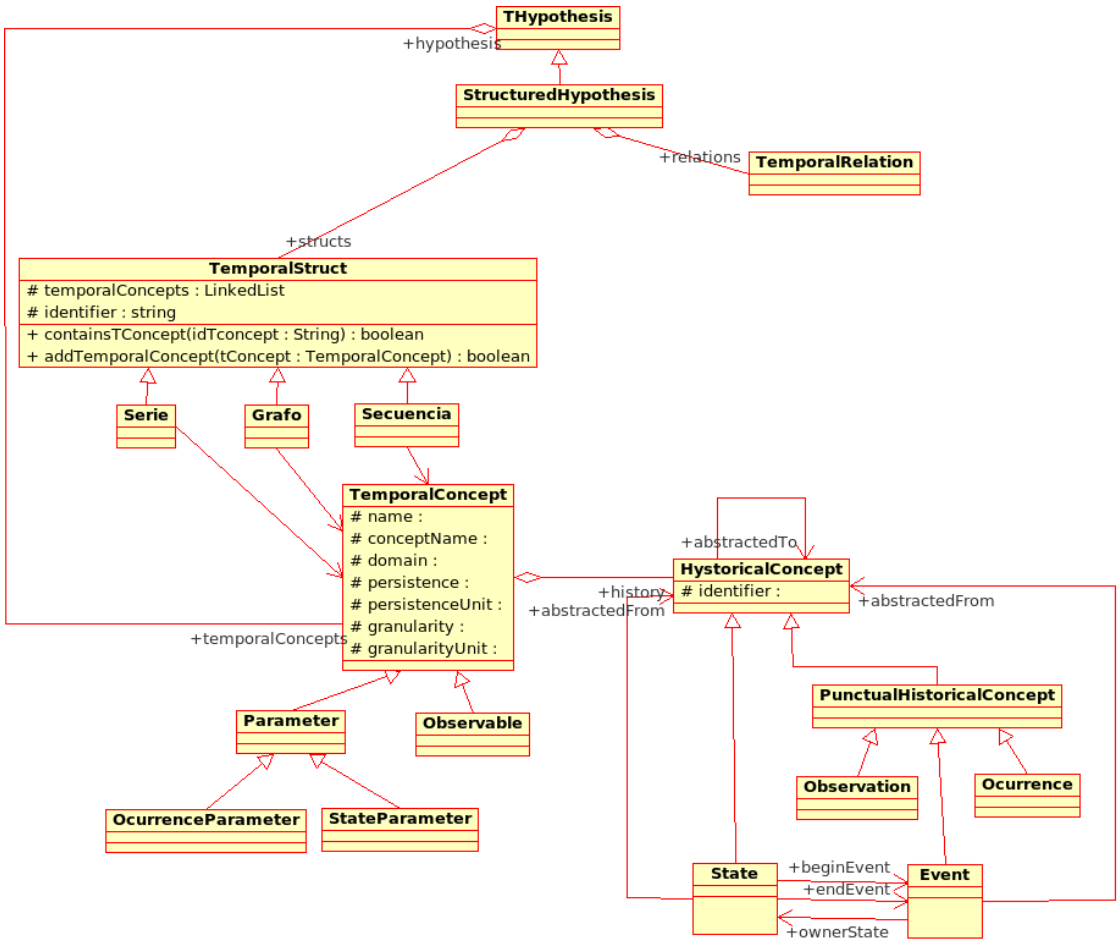


Ilustración 8: Modelo Conceptual. Estructuras temporales

Relaciones temporales

El modelo temporal extendido establece una jerarquía de clases para la representación de las relaciones temporales que se han podido definir en los datos de entrada procedentes del sistema de adquisición de datos.

Con esta jerarquía de clases se intenta desacoplar la definición de relaciones del modelo temporal extendido de los conceptos temporales definidos en `temporalAbstraction`. Así pues, toda relación de esta jerarquía contendrá la instancia de la relación en el mundo temporal, la cual se crea a partir de las entidades del mundo temporal correspondientes al concepto histórico origen y destino de la relación (`fuzzyTIME`).

Los tipos de relaciones que encontramos en el mundo temporal son:

- Relaciones Cualitativas de Punto a Punto (QPP): que pueden ser AFTER, EQUALS y BEFORE
- Relaciones Cualitativas de Punto a Intervalo (QPI): que pueden ser

BEFORE, STARTS, DURING, FINISHES y AFTER.

- Relaciones Cualitativas de Intervalo a Punto (QIP): las inversas de las anteriores.
- Relaciones Cualitativas de Intervalo a Intervalo (QII): que pueden ser OVERLAPS, BEFORE, MEETS, STARTS, DURING, FINISHES, sus inversas y la relación EQUALS.
- Relaciones Cuantitativas Punto a Punto (MPP): expresan la distancia entre dos puntos temporales.

Si llevamos las relaciones temporales al dominio médico, podemos tomar como ejemplo de relación la definición de un procedimiento de actuación en el que se quiere indicar que tras administrar el medicamento A deberían pasar 30 minutos para administrar el medicamento B, esta situación se trasladaría a una representación temporal donde se establece una relación MPP con una duración de 30 minutos entre un punto origen A y un punto destino B.

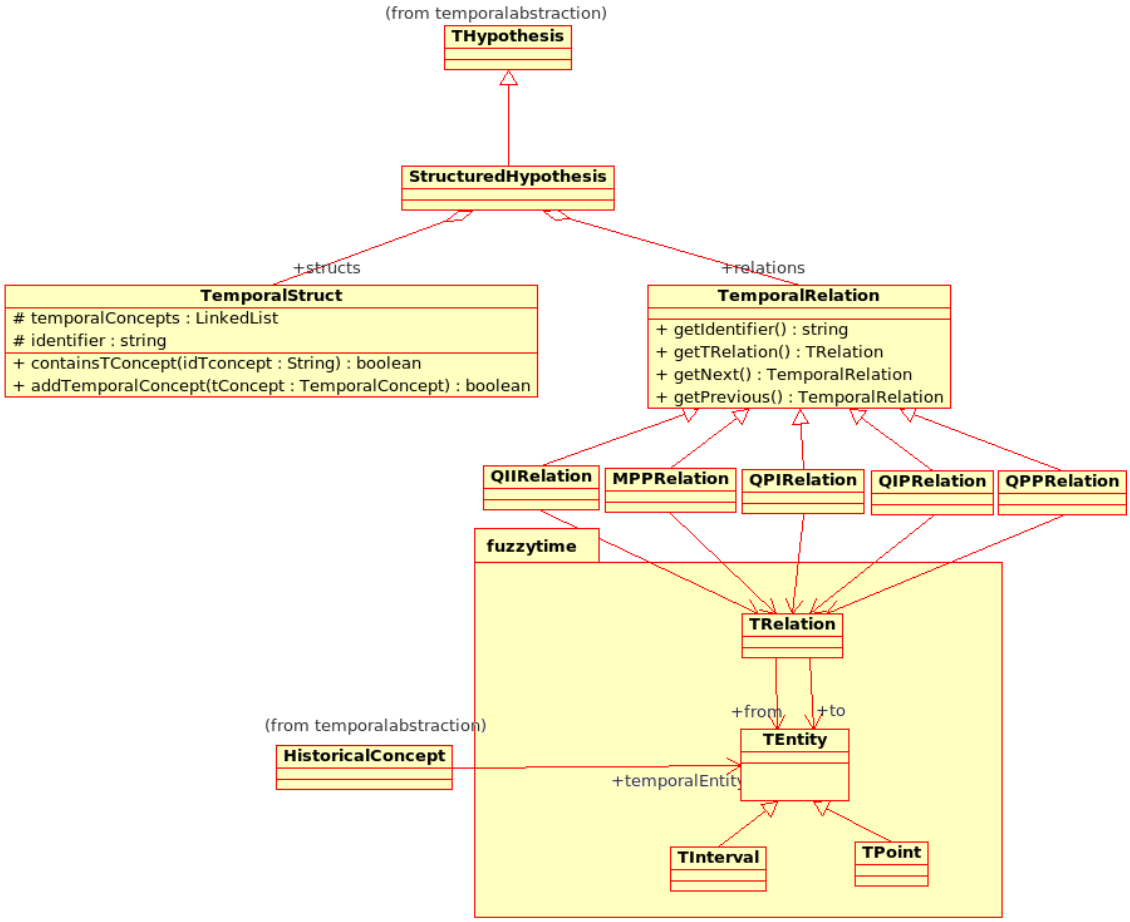


Ilustración 9: Modelo conceptual. Relaciones temporales

Entidades de visualización

En el módulo de visualización se establece una jerarquía de clases que definen la representación de la entidades y de la estructuras en el grafo. Además de establecer la representación ofrecen operaciones necesarias en el proceso de visualización como es el cálculo de su posición. A su vez se define una jerarquía de formas, dando la opción en la

representación de que una entidad de tipo punto pueda dibujarse según la forma asociada dinámicamente a su grafo.

La clase `TemporalGraph` sería la encargada de generar el grafico de una estructura temporal, haciendo uso las clases `RelationGraph`, `EntityGraph` y sus subclases en las que delega el generar la representación gráfica de una relación o entidad respectivamente, que posteriormente será añadido al grafico de la estructura. Para generar esta representación gráfica se apoya en la jerarquía de formas, ya que al dibujar una entidad de tipo punto podremos asociarle dinámicamente una u otra forma dependiendo de si por ejemplo se trata un punto preciso o impreciso, variando por tanto su representación, como se puede ver en la sección 3.11 de este documento. Ello permite extender en versiones futuras las formas de representar los distintos elementos del gráfico.

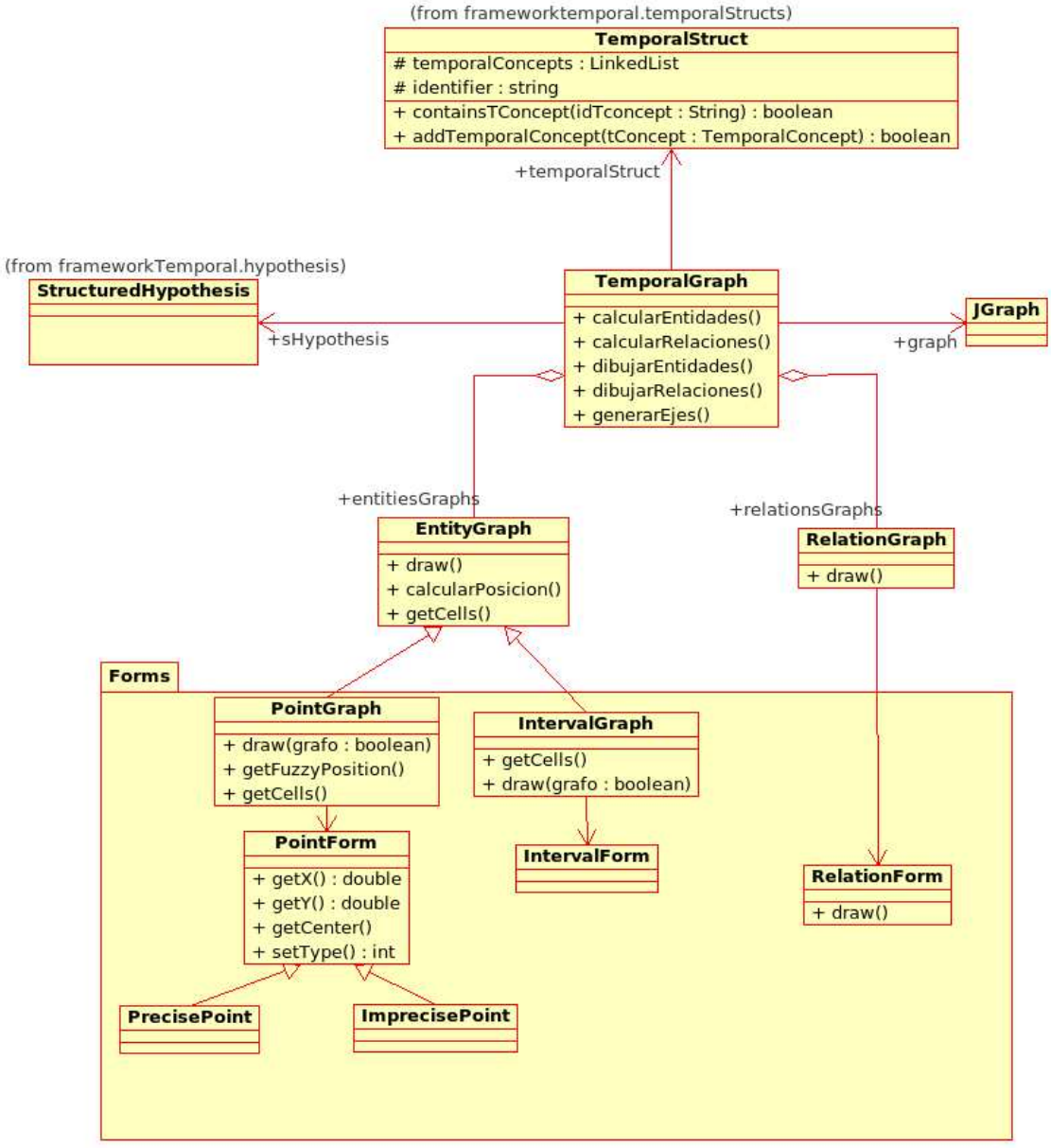


Ilustración 10: Modelo Conceptual. Grafos y formas

Mecanismos de abstracción

Cabe destacar, como parte del modelo conceptual del módulo de abstracción, la representación de los mecanismos de abstracción implementados. Al definir la interfaz `IMechanism` que implementan todos los mecanismos de abstracción, hace que en general, si se desea definir un nuevo mecanismo baste con que implemente dicha interfaz.

Los mecanismos de abstracción poseen el método `apply` que ejecutará el proceso de abstracción, siendo necesario hacer uso de la ontología del dominio, la cual es pasada como parámetro.

Haciendo uso del patrón de diseño *composite* se ofrece la posibilidad de componer varios mecanismos, donde la clase `TAComposite` ejecutará varios procesos de abstracción, donde la salida de uno es la entrada del siguiente de forma transparente al usuario.

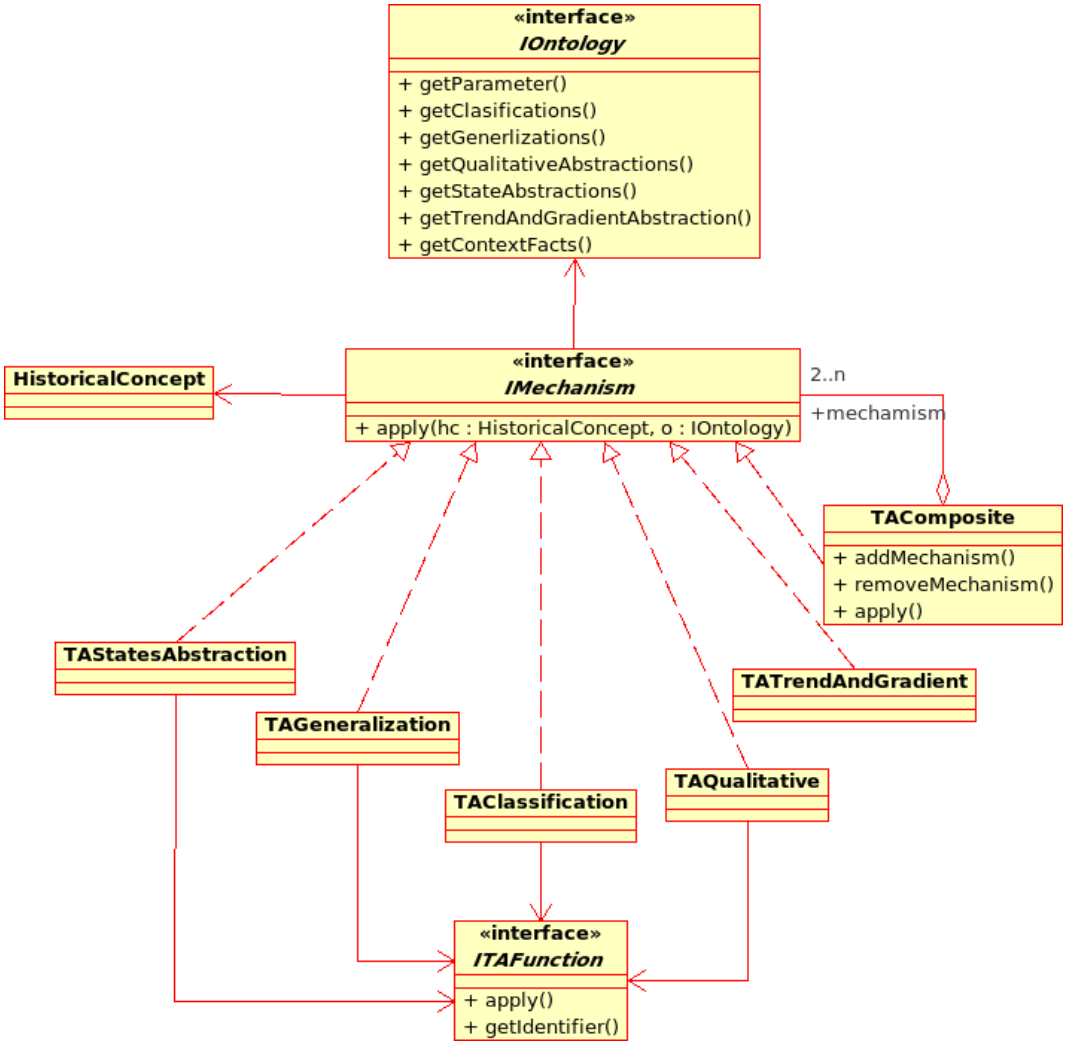


Ilustración 11: Modelo Conceptual. Mecanismos de abstracción (módulo abstracción)

Un ejemplo de composición de mecanismos sería el de la Ilustración 12, donde la entrada será el concepto histórico que se desea abstraer y la salida una colección de conceptos históricos, siendo la salida de una caja la entrada de la siguiente.

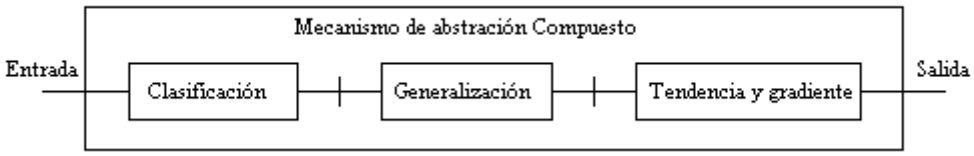


Ilustración 12: Ejemplo de composición de mecanismos de abstracción

3.3. Estructura del modelo temporal extendido

El modelo temporal extendido está compuesto por cuatro paquetes como se muestra en la Ilustración 13.

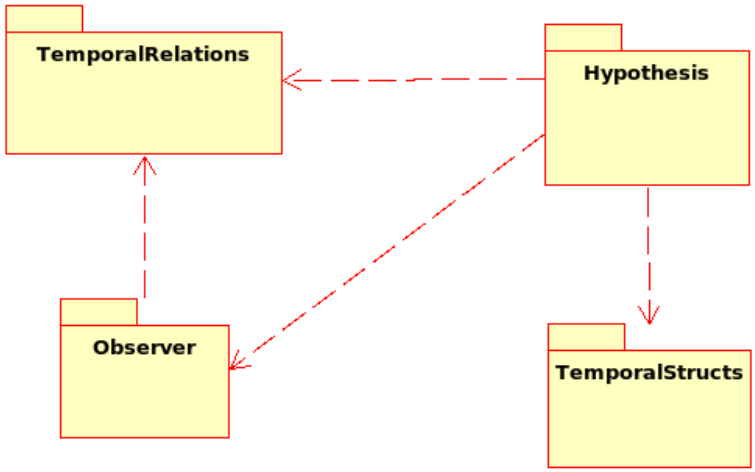


Ilustración 13: Estructura del modelo temporal extendido

El paquete `Observer` contiene las clases necesarias para la comunicación con el sistema de adquisición de datos para la entrada de datos temporales. Puesto que este proyecto se ha desarrollado de forma que el sistema de adquisición de datos pueda ser un fichero XML o una base de datos, se han implementado dos mecanismos de comunicación, uno en el que la comunicación se realiza mediante un analizador de ficheros XML y otro en el que se comunica mediante un controlador JDBC con la base de datos.

El paquete `Hypothesis` contiene las clases relacionadas con la hipótesis de abstracción. Contendrá una clase `ShypothesisBuilder`, ya que se usa el patrón builder para la construcción de la hipótesis de abstracción.

El paquete `TemporalStructs` contiene las clases que representan las distintas estructuras temporales que se pueden instanciar las cuales contendrán la información temporal.

El paquete `TemporalRelations` contiene las clases necesarias para la representación de las relaciones temporales establecidas entre conceptos temporales en los datos temporales de entrada. Da soporte a los 5 tipos de relaciones: MPP, QPP, QPI, QIP y QII.

3.4. Estructura del módulo de visualización

El módulo de visualización está compuesto por tres paquetes principales como se muestra en la Ilustración 14.

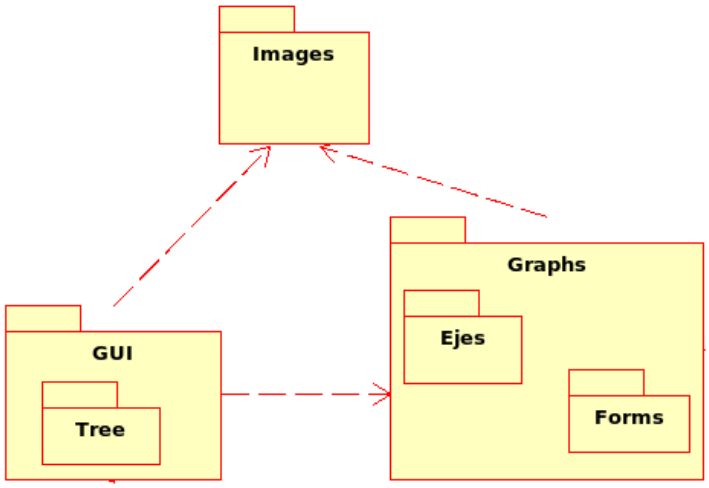


Ilustración 14: Estructura de paquetes del módulo de visualización

El paquete `GUI` contiene las clases que conforman la interfaz gráfica de usuario que permitirá al usuario la interacción y visualización de contenidos. Además contiene otro paquete donde se proporciona un árbol que da una visión jerárquica y resumida de los elementos visualizados.

El paquete `Graphs` contiene las clases que generan los grafos para las entidades del dominio, así como de los ejes de las gráficas. Las clases que generan la representación de las distintas entidades en el gráfico ofrecen funcionalidad para el cálculo de su posición y la adición de información sobre la entidad dibujada.

Además tenemos el paquete `imágenes` que contiene los archivos de imagen necesarios para la interfaz.

3.5. Estructura del módulo de adquisición

El módulo de adquisición está compuesto por paquetes como se muestra en la Ilustración 15.

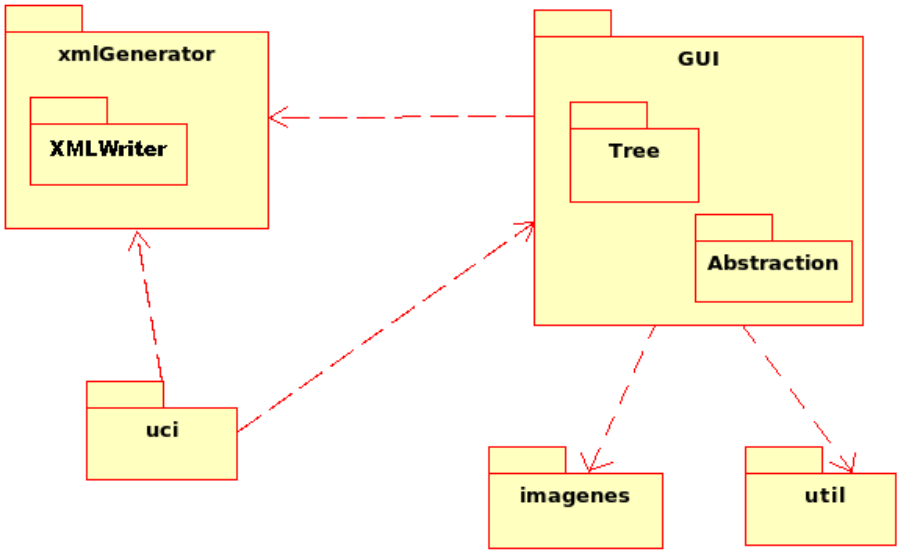


Ilustración 15: Estructura de paquetes del módulo de adquisición

El paquete `GUI` contiene las clases que conforman la interfaz gráfica de usuario, que permita el proceso de adquisición de datos. Además, contiene otro paquete donde se proporciona un navegador que da una visión jerárquica y resumida de la ontología, contenedora del conocimiento adquirido a través del asistente.

El paquete `util` contiene clases utilizadas por otros paquetes, como es el implementar una lista ordenada de elementos.

El paquete `XMLGenerador` contiene la interfaz de comunicación con la ontología. El proyecto se ha desarrollado de forma que la comunicación ha sido implementada de forma local, basada en el acceso a un fichero XML. Además contiene la clase encargada del volcado a XML de la hipótesis de abstracción obtenida.

Además tenemos el paquete `imágenes` que contiene los archivos de imagen necesarios para la interfaz.

3.6. Diseño de la Base de datos

La base de datos diseñada contendrá la información temporal objeto y/o fruto del proceso de abstracción. En la Ilustración 16 se muestra el esquema de la base de datos:

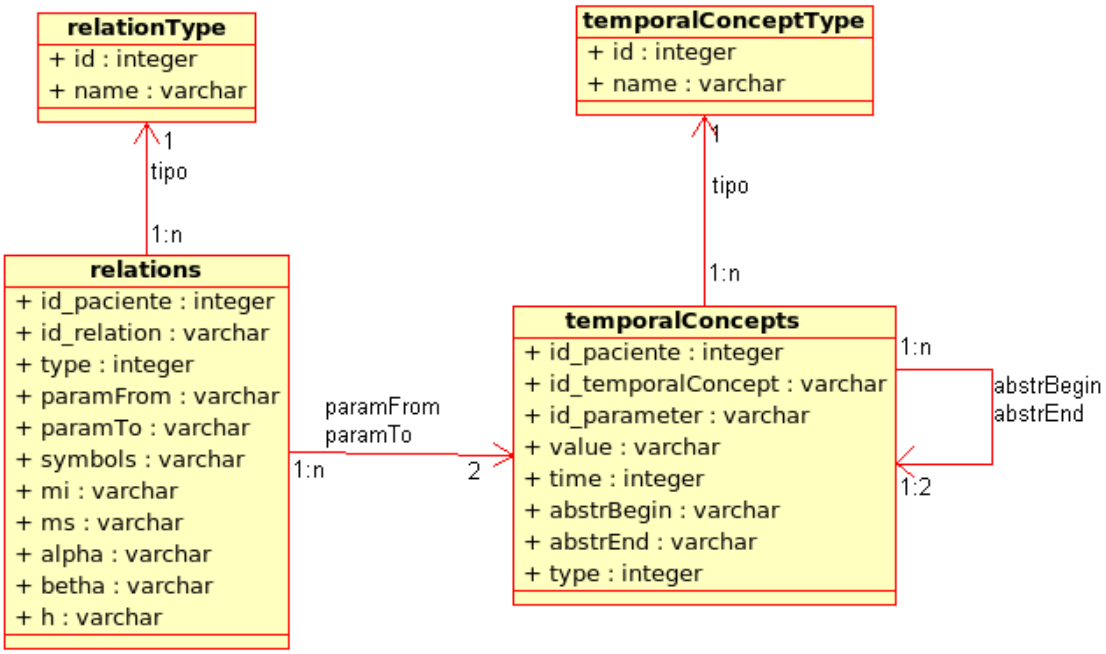


Ilustración 16: Esquema relacional de la base de datos

La tabla `relationType` contiene una columna que representa un identificador numérico para cada tipo de relación entre conceptos temporales, teniendo cada uno asociado un nombre descriptivo.

La tabla `relations` almacena relaciones entre conceptos temporales para un paciente determinado. El campo `type` hace relación a la tabla `relationType`. Si el tipo es `mpp`, el campo `symbols` estará a null y la duración vendrá indicada por los

campos `mi`, `ms`, `alpha`, `betha` y `h`. En caso contrario, los campos que indican la duración estarían a null y el campo `symbols` contendría un array de cadenas de caracteres. Los campos `paramFrom` y `paramTo` hacen referencia a un concepto temporal de la tabla `temporalConcepts`.

La tabla `temporalConceptType` contiene una primera columna con un identificador numérico por cada tipo de concepto temporal y una segunda columna que contiene un nombre descriptivo para cada concepto temporal.

La tabla `temporalConcepts` almacena las historias de datos temporales asociadas a la adquisición de datos de distintos pacientes. El campo tipo que es una clave ajena a la tabla `temporalConceptType` indica el tipo del concepto que contiene la fila de la tabla. Para `Observations`, los campos `abstrBegin` y `abstrEnd` tendrán valor nulo y el campo time contendrá el instante de tiempo en el que se obtuvo el dato. Para `Occurrences`, los campos time y `abstrEnd` tendrán valor nulo y el campo `abstrBegin` contendrá el identificador del concepto temporal del que ha sido abtraido. Para `States`, el campo time tiene valor nulo y los campos `abstrBegin` y `abstrEnd` contiene los conceptos de los que ha sido abstraído el estado.

3.7. Persistencia de la base de conocimiento

En el módulo de adquisición de conocimiento se incorpora un sistema de comunicación con la base de conocimiento, la cual se almacenará de forma estática y local en archivos XML. Por tanto la interfaz de comunicación se implementará como un intérprete de ficheros XML, los cuales deben ser validados contra un esquema para evitar conflictos.

La interfaz de comunicación hereda gran parte de su funcionalidad de la interfaz de comunicación definida para el módulo de abstracción. Ha sido necesario ampliarla ya que estaba definida para la lectura de la base de conocimiento, sin embargo en el contexto de este proyecto también es necesario hacer persistente la ontología, por lo que ha habido que implementar funcionalidad de escritura en la base de conocimiento.

La base de conocimiento constará de una colección de parámetros, una colección de hechos de contexto y las abstracciones definidas entre los parámetros definidos anteriormente.

En la Ilustración 17 se muestra como se integra el sistema de comunicación del módulo de adquisición con el módulo de adquisición de conocimiento implementado en este proyecto, donde el paquete `XMLGenerator` contendrá la implementación del sistema de comunicación con la base de conocimiento en formato XML, y en concreto el paquete `XMLWriter` contendrá las clases necesarias para hacer persistente la ontología.

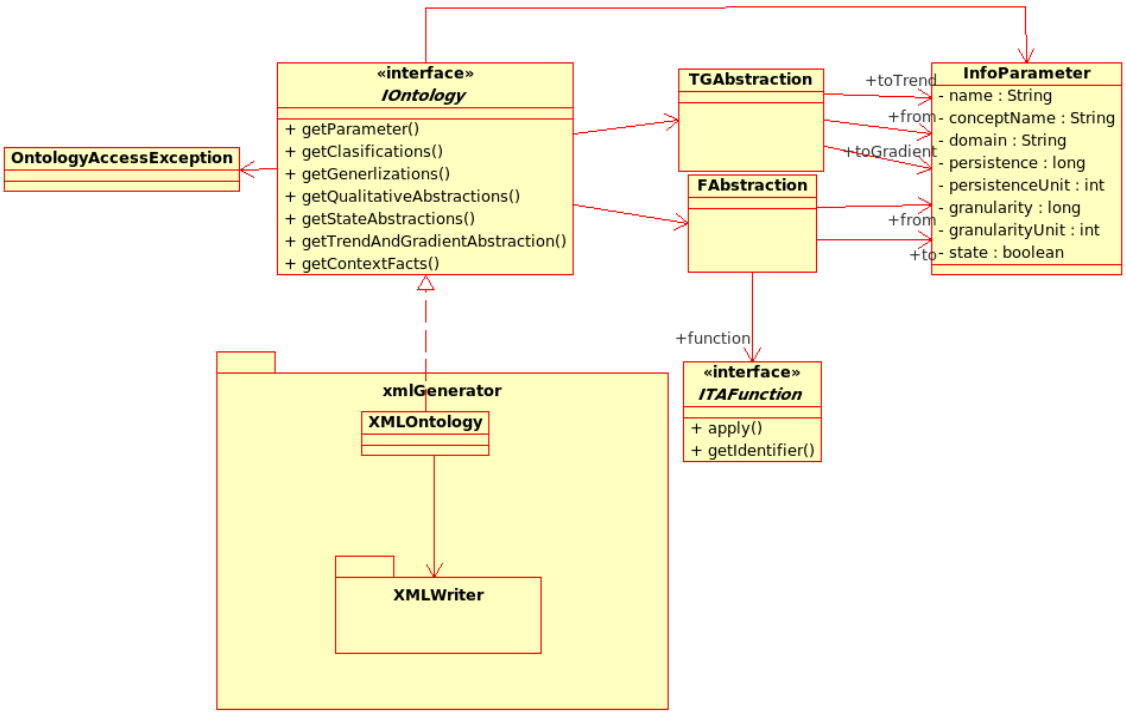


Ilustración 17: Interfaz de comunicación con la base de conocimiento

3.8. Diseño del sistema de adquisición de datos de datos temporales

Puesto que queda fuera del alcance de este proyecto la integración del prototipo con el sistema de adquisición de datos real, se ha implementado una aproximación en local de forma que sirva como un primer paso para la posterior comunicación con el sistema real. Se han implementado dos vías para la comunicación con el sistema de adquisición de datos, una basada en XML y otra en base de datos, mediante una conexión JDBC.

Para el sistema de adquisición de datos local basado en XML los datos temporales (conceptos y relaciones temporales) se almacenan en un fichero XML de forma estática, ofreciendo la interfaz de comunicación métodos para el acceso de forma secuencial a los conceptos temporales y relaciones temporales, que han sido obtenidos del fichero XML.

El sistema de comunicación que se comunica con el sistema de adquisición de datos representado por el fichero XML contendrá colecciones de observaciones, ocurrencias y estados, y otra colección para las relaciones temporales, que según la interfaz de comunicación serán accedidas secuencialmente. Una observación estará representada por una tupla compuesta por los atributos concepto, característica, valor y marca de tiempo, una ocurrencia estará representada por la tupla compuesta por los atributos concepto, característica, valor y concepto del que ha sido abstraída, un estado lo componen los elementos concepto, característica, valor, concepto del que ha sido abstraído el inicio y concepto del que ha sido abstraído el fin. Por otra parte las relaciones temporales, donde habrá que hacer dos grupos, las relaciones MPP que se representarán por la tupla compuesta por los atributos concepto origen de la relación, concepto destino de la relación y duración, y el segundo grupo lo formarán el resto de

relaciones QPP, QPI, QIP y QII, que estarán representadas por la tupla compuesta por los atributos concepto origen de la relación, concepto destino de la relación y los símbolos que expresan la relación entre los conceptos temporales.

Para evitar errores en la comunicación con el sistema de adquisición de datos que representa el fichero XML, habrá que validarlo contra el esquema XML que define la estructura que debe tener, el cual ha sido creado usando XMLSchema.

Por otra parte, para realizar la comunicación con el sistema de adquisición de datos local basado en una base de datos, almacenará los datos temporales según el diseño de la base de datos, donde habrá campos para representar cada uno de los elementos de las tuplas que devolverá el sistema de adquisición de datos al que representa. En este caso, el sistema de comunicación también almacenará colecciones para observaciones, ocurrencias y estados, y otra colección para las relaciones temporales, que más tarde serán accedidas en la implementación de los métodos de la interfaz de comunicación.

La interfaz de comunicación define principalmente tres tipos de métodos, los cuales se implementarán particularmente para cada uno de los tipos de conceptos históricos y para relaciones. El método `read<Observations/Occurrences/States/Relations>` devuelve el número indicado de objetos del tipo al que haga referencia el método. Los otros dos métodos permiten acceder a las colecciones creadas a partir de los datos obtenidos del sistema de adquisición de datos como si de un iterador de Java se tratase. El método `next<Observations/Occurrences/States/Relations>` devuelve el siguiente elemento y `hasMore<Observations/Occurrences/States/Relations>` devuelve un booleano para indicar si quedan más elementos por recorrer. Las clases encargadas de la comunicación con los distintos sistemas de adquisición de datos implementarán esta interfaz de comunicación, de forma que se oculte a los usuarios del sistema de comunicación la naturaleza del sistema de adquisición de datos.

En el diseño se intenta desacoplar los conceptos históricos y relaciones definidas en el modelo de abstracción de las devueltas por el sistema de adquisición de datos. De esta forma un concepto del sistema de adquisición de datos puede ser usado en varios procesos de abstracción, mientras que una instancia de un concepto histórico sólo puede pertenecer únicamente a una hipótesis, ya que tiene asociada una entidad temporal la cual sólo puede pertenecer a una red de restricciones borrosas. Las clases utilizadas son `ObserverObservation`, `ObserverOccurrence`, `ObserverState` y `ObserverRelation`.

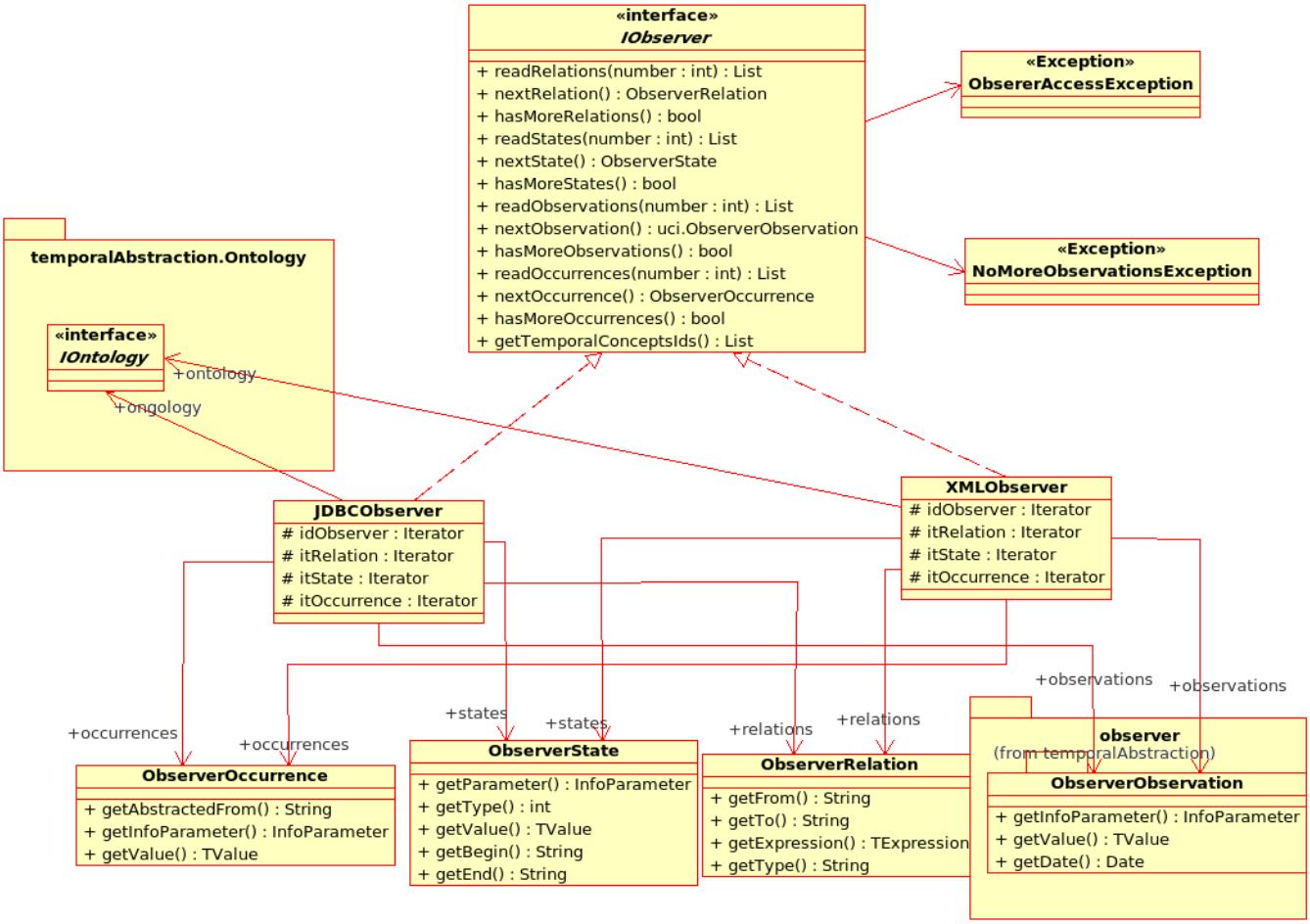


Ilustración 18: Estructura del sistema de comunicación

3.9. Persistencia del modelo de abstracción

El modelo de abstracción ya definido hace persistente el modelo resultante serializando la instancia de `TAHypothesis`. En este proyecto se ha extendido esta funcionalidad de forma que el modelo resultante del proceso de abstracción pueda hacerse persistente en forma de fichero XML, siguiendo la estructura del sistema de adquisición de datos basado en XML.

En otro momento podría volver a realizarse el proceso de abstracción sobre este fichero XML, resultado de una anterior abstracción, o una posterior visualización de estos datos temporales, ya que tanto el sistema de comunicación como el sistema de adquisición de datos se han ampliado para que además de tratar observaciones operen con ocurrencias y estados, que son conceptos obtenidos por la abstracción de otros.

3.10. Integración de las herramientas

Este proyecto está estructurado en distintos módulos, siendo algunos objeto de desarrollo en este proyecto y otros ya desarrollados. Para la construcción de un prototipo final, será objeto de estudio la forma de integrar dichos módulos, ya que han sido desarrollados de forma totalmente independiente. Por ejemplo, se podría usar el módulo de adquisición en cualquier otra herramienta que necesitase dicha funcionalidad

para realizar otro procesamiento posterior.

El módulo de abstracción temporal ya desarrollado ofrece una API en la que previamente a la abstracción se realiza la comunicación con el sistema de adquisición de datos para acceder a la información temporal, la cual resulta insuficiente ya que la interfaz de comunicación sólo permite la lectura de observaciones. Se ha decidido por tanto, ampliarla de forma que los datos temporales también puedan contener otros tipos de conceptos que quizá sean fruto de una anterior abstracción, que hayan sido almacenados y que posteriormente se puedan volver a abstraer.

Por lo tanto, no será suficiente el acceder a la API proporcionada por el módulo de abstracción temporal, sino que también habrá que heredar de parte de la API, en concreto de la parte referida a la comunicación con el sistema de adquisición de datos para extender su funcionalidad, dando cabida a la interacción con otros tipos de conceptos y relaciones temporales.

La integración con el módulo de abstracción ha sido costosa, puesto que han sido necesarios cambios en partes de la implementación del módulo. En concreto, ya que se plantea en este proyecto el añadir la posibilidad de definir relaciones entre conceptos temporales, habrá que ofrecer la posibilidad de que los datos temporales de entrada puedan tener indefinida su marca temporal, para posteriormente ser definida mediante una relación respecto a otro concepto. Es el caso de observaciones sin un instante de tiempo, ocurrencias sin origen de abstracción, estados donde su inicio o fin no se sabe de que concepto han sido abstraídos.

Para el resto de módulos, puesto que se ha diseñado e implementado durante el desarrollo del proyecto, es lógico diseñar las distintas API pensando también en la integración con el resto. Por lo tanto, lo más eficiente e intuitivo es usar directamente las API proporcionadas por cada uno de los módulos.

3.11. Ejemplo de adquisición de conocimiento y visualización

Siguiendo los casos de uso definidos, en este apartado se hace un seguimiento detallado del uso de la aplicación, mostrando para ello la interfaz gráfica.

En este ejemplo se adquiere conocimiento para realizar una abstracción de estado sobre un parámetro de temperatura y visualizarla posteriormente. Este ejemplo muestra la utilidad de la aplicación tanto en el dominio médico como en el dominio de la cronobiología, ya que se realiza sobre un parámetro de temperatura cuyos datos no están vinculados a ningún dominio concreto.

Se ejemplifica como el usuario sigue el asistente en el proceso de adquisición guiado por el caso de uso “adquirir conocimiento”. El proporcionar al usuario el modo asistente facilita la adquisición de conocimiento, ya que es un proceso con una dificultad añadida, pero que se ve disminuida al guiar al usuario por los distintos pasos que ha de seguir:

1. Crea una nueva ontología.
2. Introduce tres parámetros nuevos (Fiebre:Temperatura, Fiebre:Cualitativa, Fiebre:Presencia).
3. Introduce el hecho de contexto “Fumador”.
4. Se crea un abstracción de estado, que define una función de mapeo del

parámetro Fiebre:Temperatura a Fiebre:Cualitativa.

Tras la adquisición se realiza el proceso de abstracción sobre la ontología definida, indicando la fuente de datos y seleccionando una abstracción de tipo estado.

Para la visualización se usarán los datos que se exportaron de la abstracción, seleccionando por tanto la interfaz de comunicación con el sistema en formato XML. Para este ejemplo se visualizarán los datos como un único grafo, de forma que se muestren las relaciones entre conceptos temporales como resultado de la abstracción de unos conceptos temporales de otros.

Proceso de adquisición de conocimiento

Tras iniciar la aplicación se comienza creando la ontología del dominio. Para ello se tienen dos opciones: mediante el botón de la barra de herramientas o en el menú *File/Nueva ontología*.

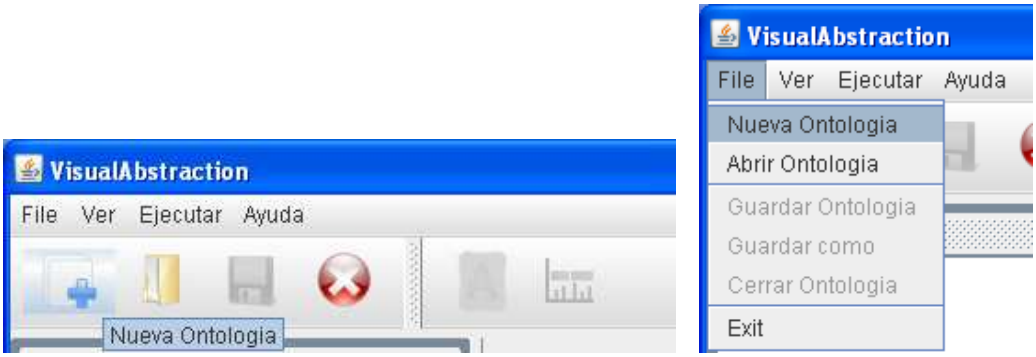


Ilustración 19: Nueva ontología

Una vez creada la ontología, se definen los parámetros, contextos y abstracciones que componen la ontología seleccionando en cada caso la pestaña correspondiente.

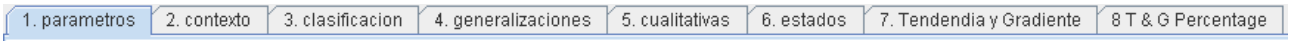


Ilustración 20: Pestañas de elementos de la ontología

Para definir los parámetros se selecciona la primera pestaña *parámetros*. Se crea un nuevo parámetro seleccionando el botón *Nuevo parámetro* del panel de parámetros.

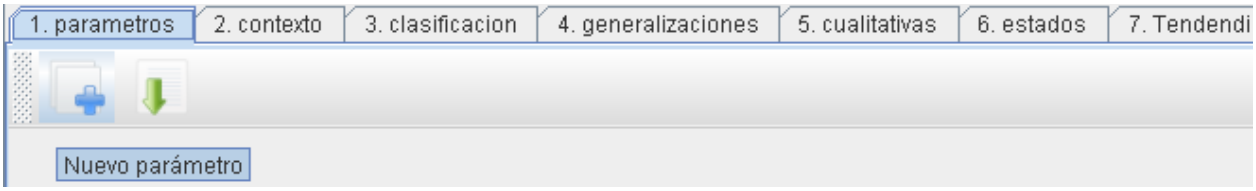


Ilustración 21: Nuevo parámetro

Se abre la ventana con los datos del parámetro, pulsando el botón *aceptar* una vez introducidos los datos necesarios que han sido definidos en el modelo de abstracción. En el caso de temperatura se selecciona un dominio `VDouble`, un valor de granularidad y persistencia de 30, cuya unidad será en segundos.

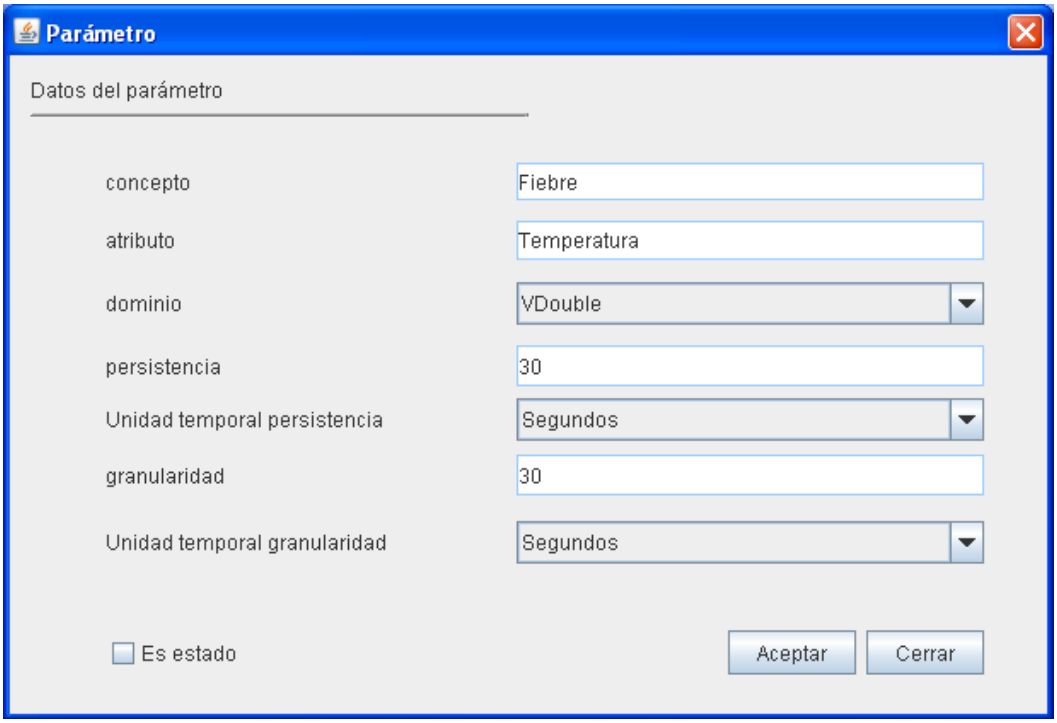


Ilustración 22: Ventana de información del parámetro

Para añadir el parámetro creado a la ontología se selecciona en la lista de parámetros definidos y se pulsa el botón *añadir* (>>) para añadirlo a la lista de parámetros de la ontología.

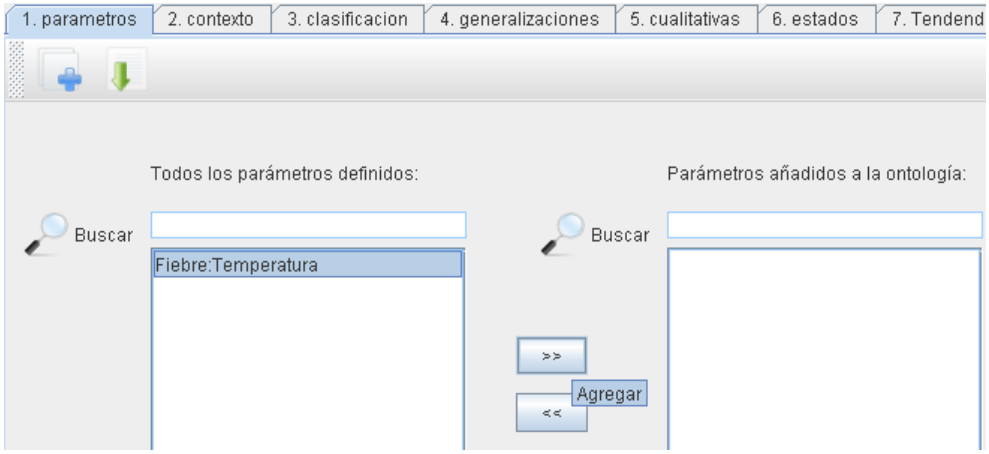


Ilustración 23: Añadir parámetro a la ontología

Se realizará el mismo proceso con cada parámetro que se quiera crear en la ontología. Para este ejemplo crearemos también los parámetros *Fiebre:Cualitativa* y *Fiebre:Presencia*. Una vez creados los parámetros e insertados en la ontología, se pueden comprobar los parámetros de la ontología tanto en la lista de *parámetros añadidos a la ontología* o en el árbol resumen de la ontología de la parte izquierda de la ventana.

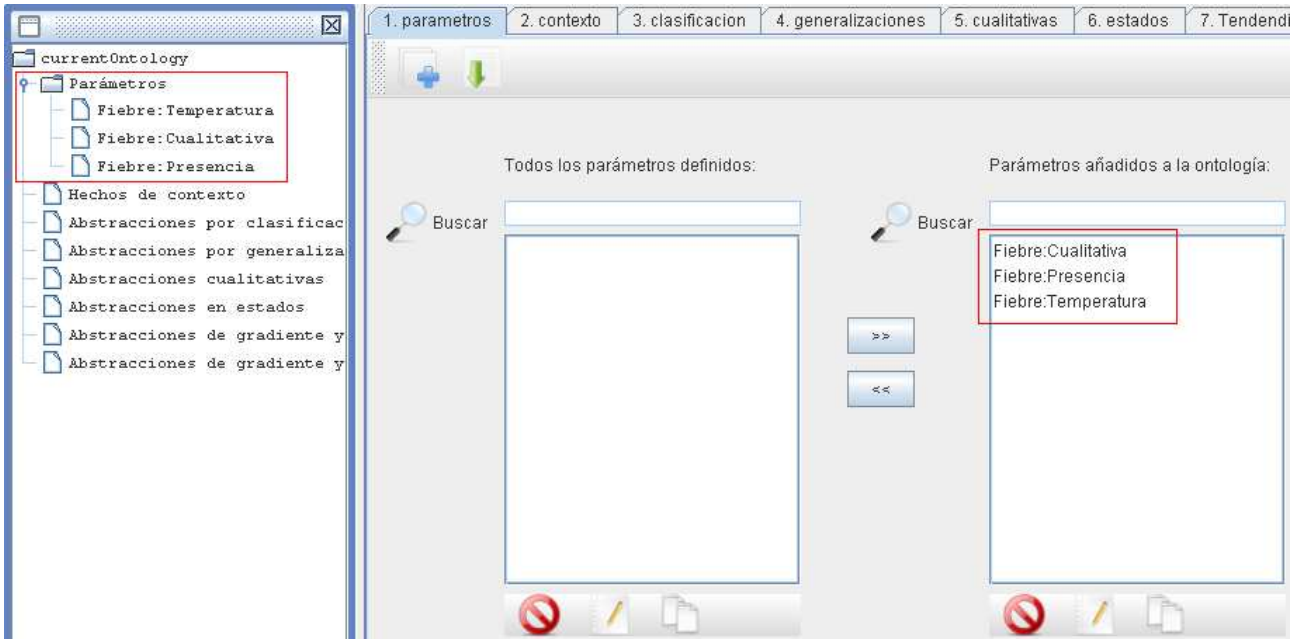


Ilustración 24: Vista de parámetros de la ontología

Para definir los contextos se selecciona la segunda pestaña *contexto*. Se crea un nuevo contexto seleccionando el botón *Nuevo contexto* del panel de contextos.

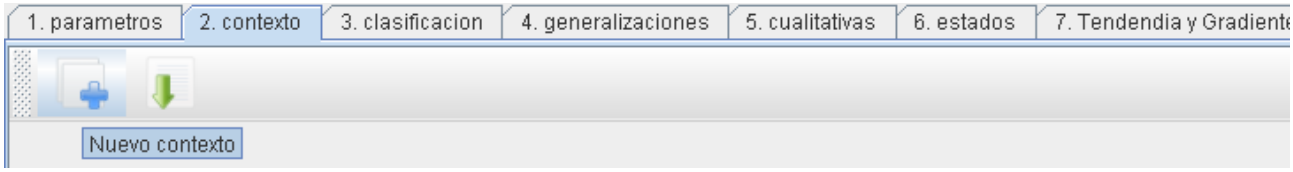


Ilustración 25: Nuevo contexto

Se abre la ventana con los datos del contexto, pulsando el botón *aceptar* una vez introducidos los datos. Al seleccionar en esta ventana el tipo o dominio del hecho de contexto se restringirá para el atributo *Valor* los valores que puede tomar. En este ejemplo se ha definido el contexto "fumador".

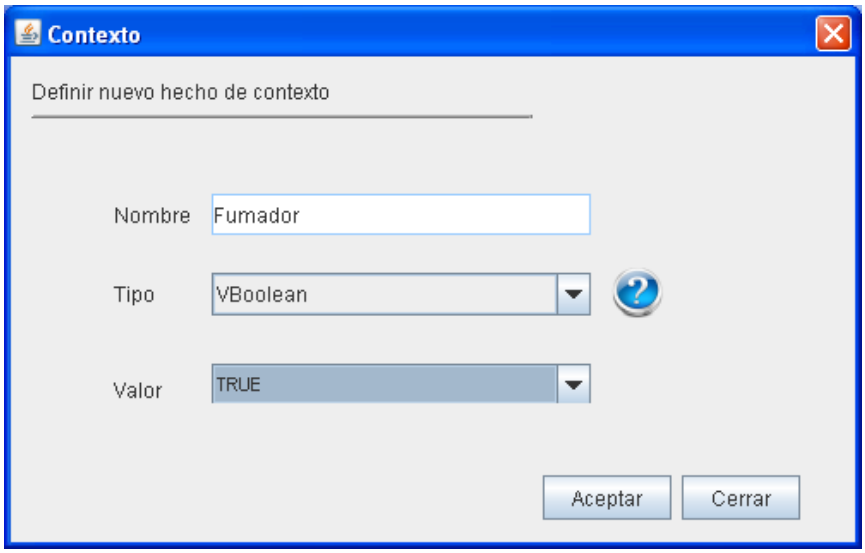


Ilustración 26: Ventana de información del hecho de contexto

Para añadir el contexto creado a la ontología se selecciona en la lista de

contextos definidos y se pulsa el botón *añadir* (>>) para añadirlo a la lista de parámetros de la ontología.

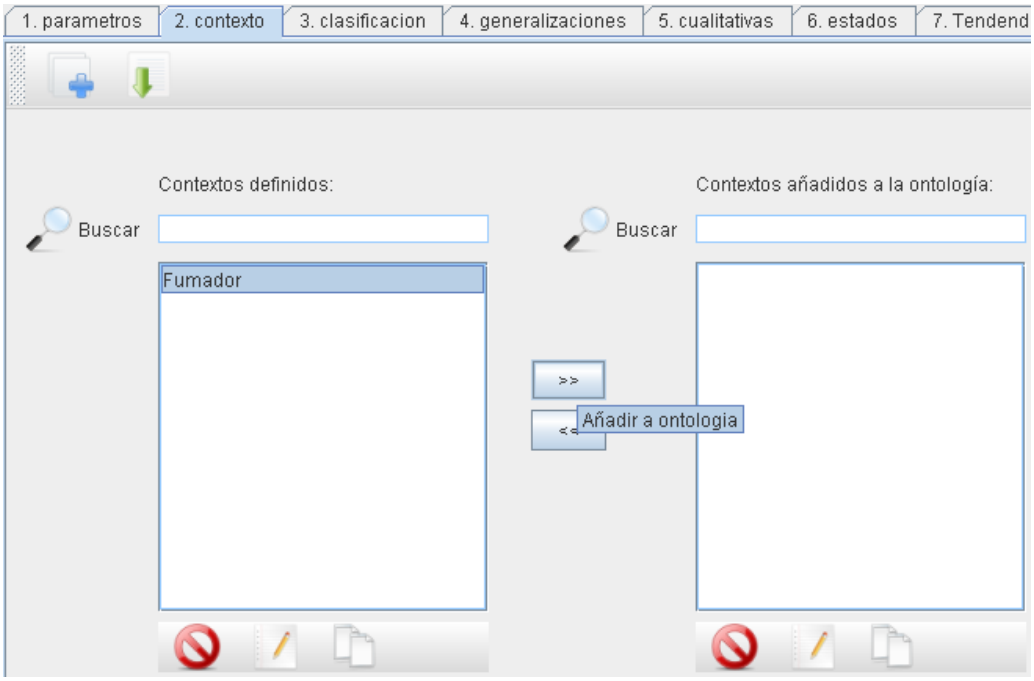


Ilustración 27: Añadir contexto a la ontología

Se realizará el mismo proceso con cada hecho de contexto que se quiera crear en la ontología. Una vez creados los contextos e insertados en la ontología, se puede comprobar su inclusión en la ontología tanto en la lista de *contextos añadidos a la ontología* o en el árbol resumen de la ontología de la parte izquierda de la ventana.

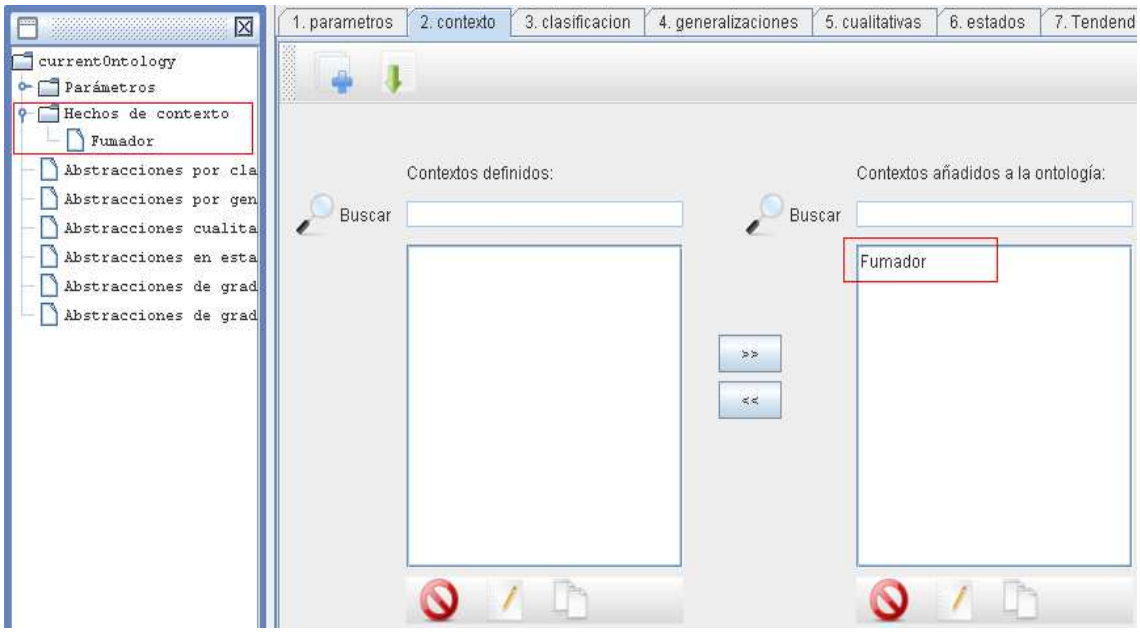


Ilustración 28: Vista de los contextos de la ontología

Para definir las abstracciones se selecciona la pestaña correspondiente al tipo de abstracción que se desea definir. En este ejemplo se creará una abstracción de estado, por lo que se seleccionará la pestaña *estados*. Se crea una nueva abstracción seleccionando el botón *Nueva abstracción* del panel de abstracciones de estado.

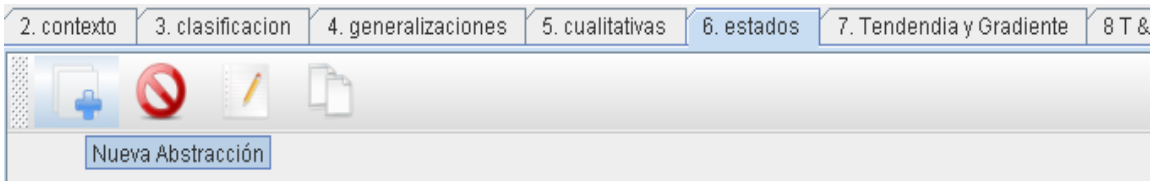


Ilustración 29: Nueva abstracción de estado

En la ventana que se abre se definen los datos de la abstracción. En un primer paso se seleccionan los parámetros origen y destino de la abstracción.

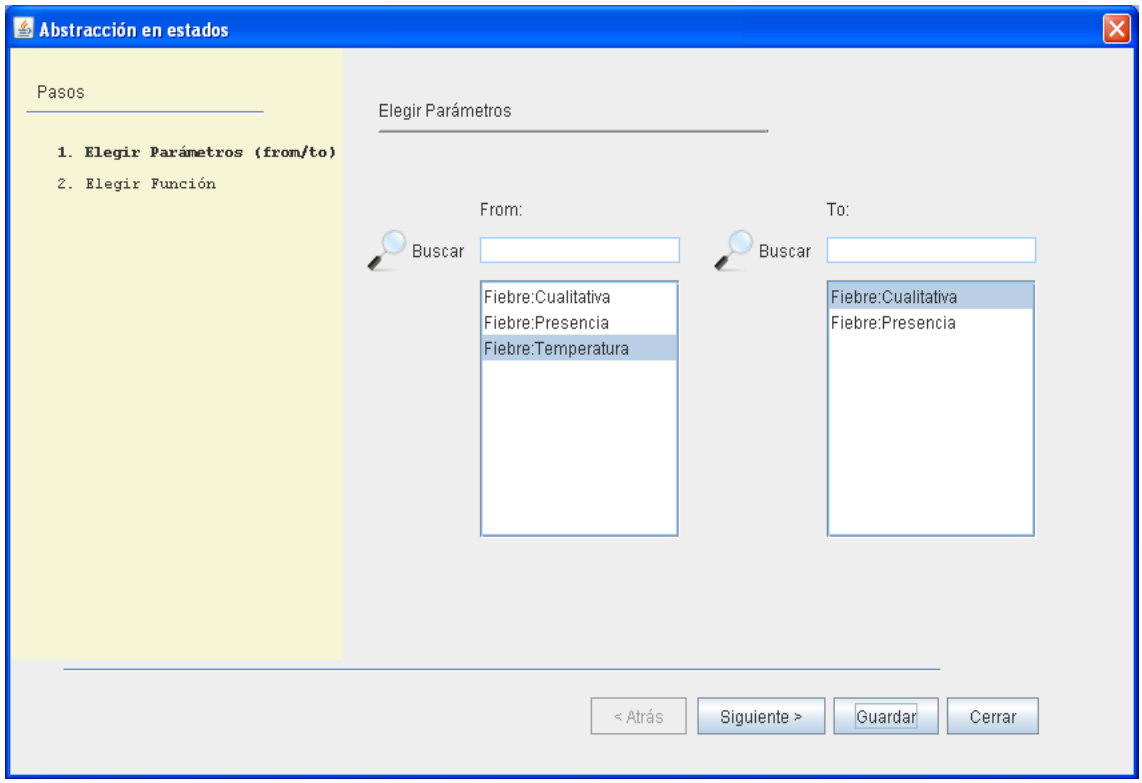


Ilustración 30: Selección de los parámetros de la abstracción

En el siguiente paso se elige la función de abstracción. En este ejemplo se seleccionará la función de mapeo, para la que hay que insertar las reglas pulsando el botón *insertar*.

La función de mapeo consiste en mapear un valor de entrada en un valor de salida en función de un conjunto de reglas. Para gestionar estas reglas de mapeo se usa el panel de la función de mapeo, donde se muestra en una tabla las reglas definidas, bajo la que se muestran tres botones para insertar, borrar y editar una regla existente.

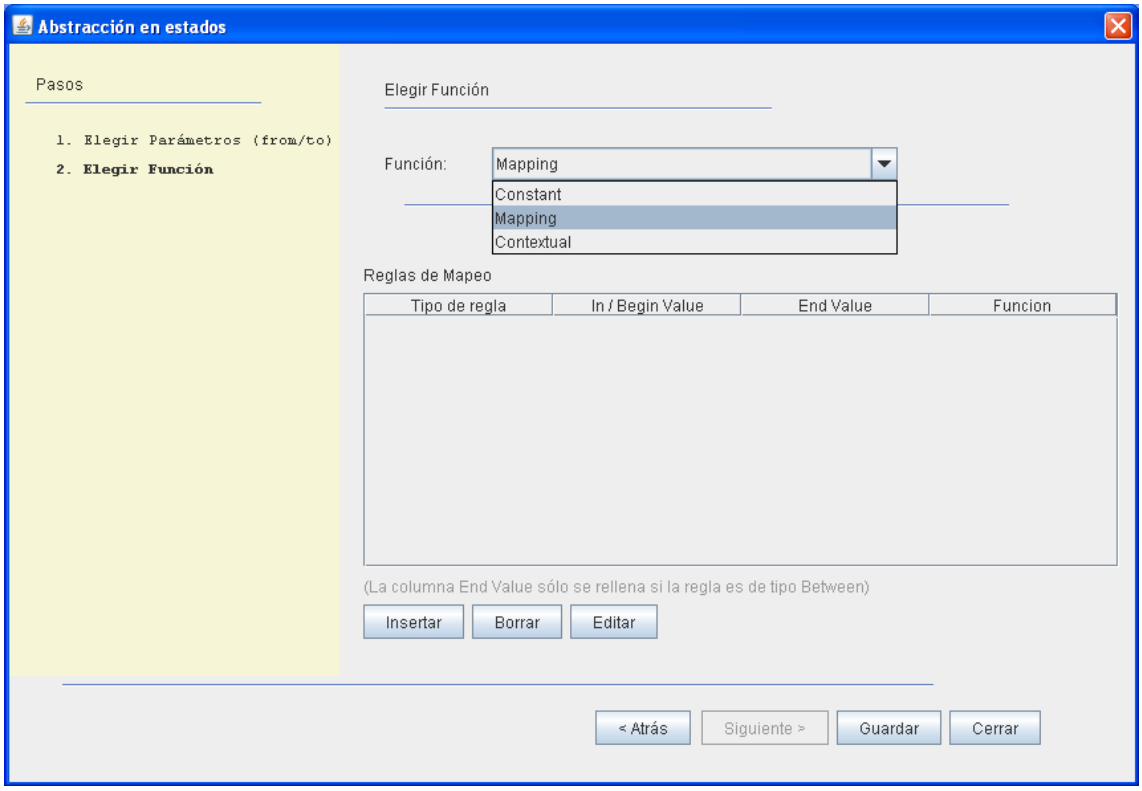


Ilustración 31: Seleccionar función de abstracción

En la ventana para añadir una regla se selecciona el tipo de regla, el valor para el parámetro origen y la función que se aplicará para calcular el valor del parámetro destino de la abstracción. En este ejemplo se elige la aplicación de una función constante, pero también se pueden usar otras funciones como identidad o expresión, siempre y cuando el dominio de los parámetros permita su uso. Se pulsa el botón *aceptar* para añadirla a la abstracción.

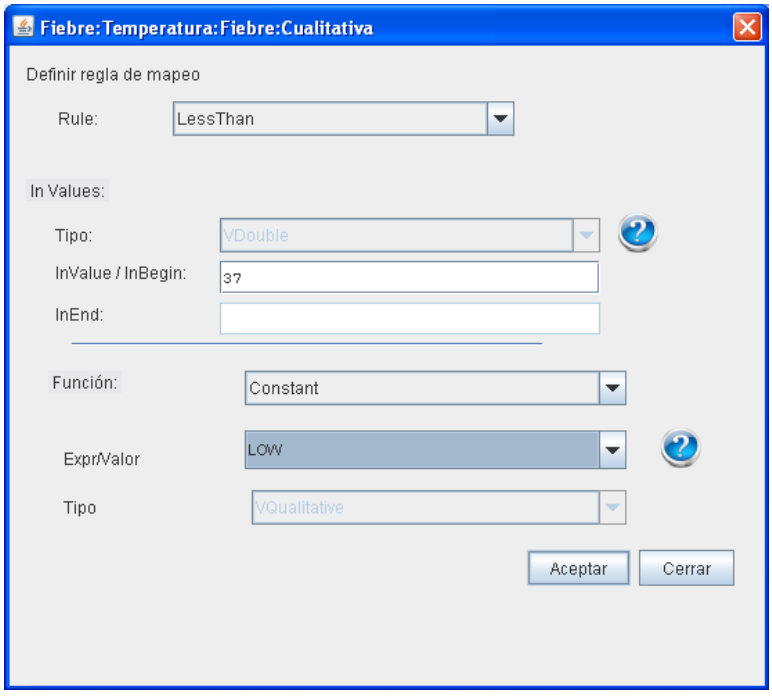


Ilustración 32: Añadir regla de mapeo

Se añaden de mismo modo tantas reglas como se deseen, las cuales aparecen en la tabla de reglas de mapeo. Se finaliza la definición de la nueva abstracción pulsando el botón *guardar*.

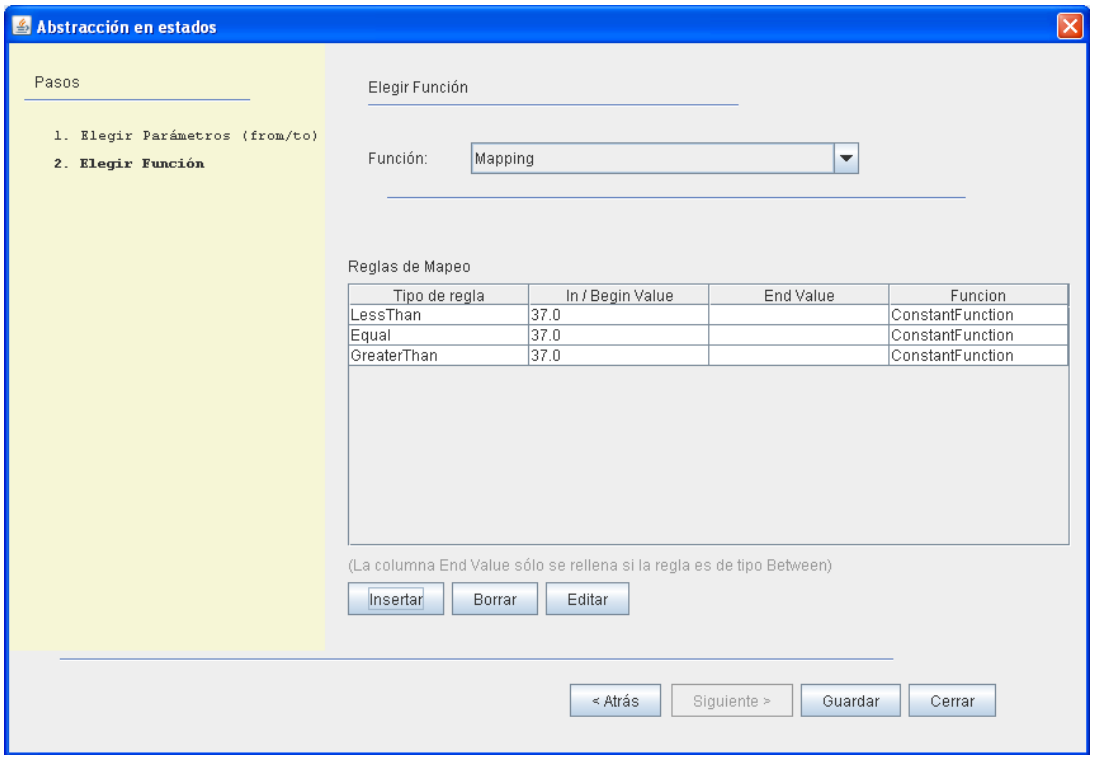


Ilustración 33: Tabla de reglas de mapeo

Se comprueba que se ha añadido la abstracción, tanto en la lista de abstracciones de estado como en el árbol de navegación de la ontología.

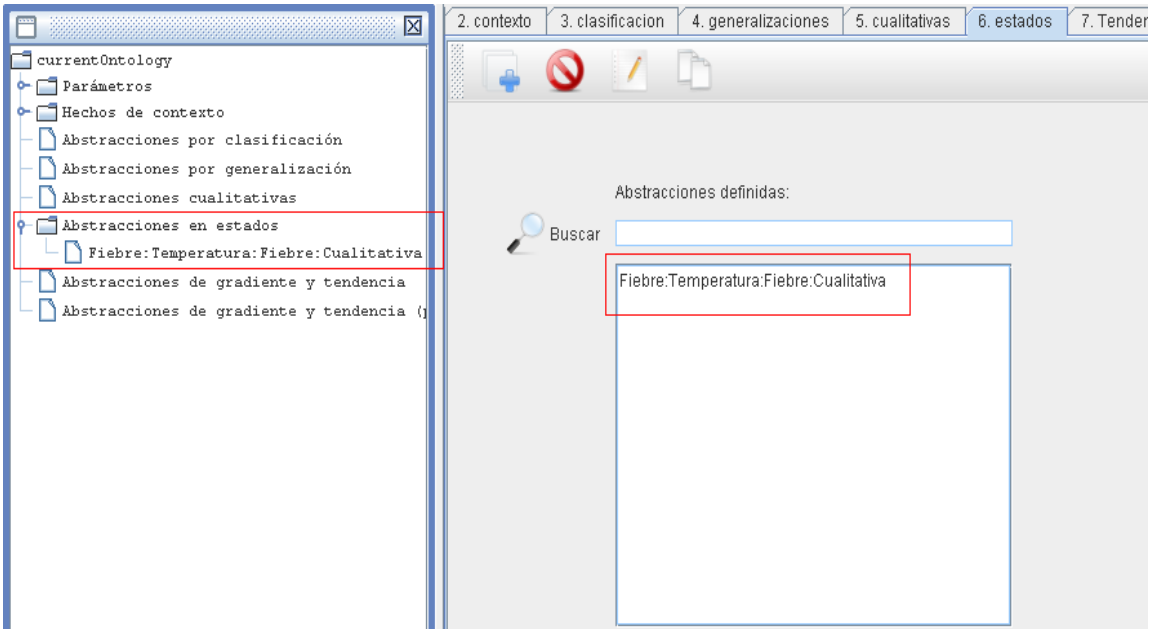


Ilustración 34: Vista de las abstracciones de estado

Para finalizar se guarda la ontología, seleccionando el botón *guardar* de la barra de herramientas o en el menú F*ile/guardar Ontología.*

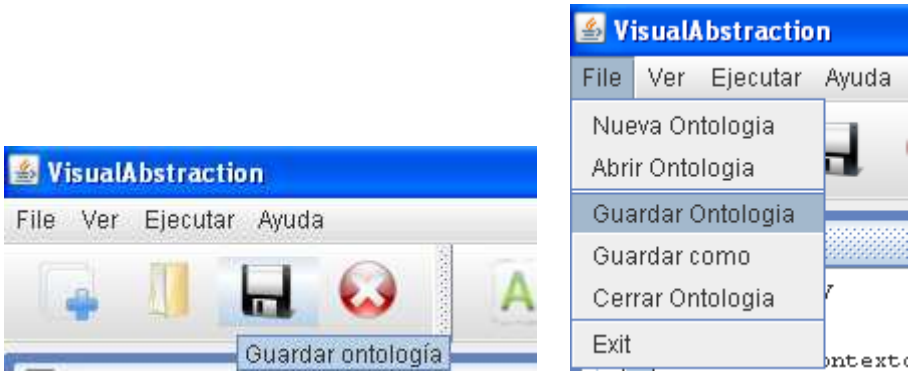


Ilustración 35: Guardar ontología

Abstracción

Para iniciar el proceso de abstracción hay que seleccionar en el menú *Ejecutar/Ejecutar abstracción* o en la barra de herramientas seleccionar el botón *abstracción*.

Ilustración 36: Ejecutar abstracción

En la ventana que se muestra a continuación, se elige la fuente de los datos temporales generados por el sistema de adquisición de datos, se da nombre a la abstracción.

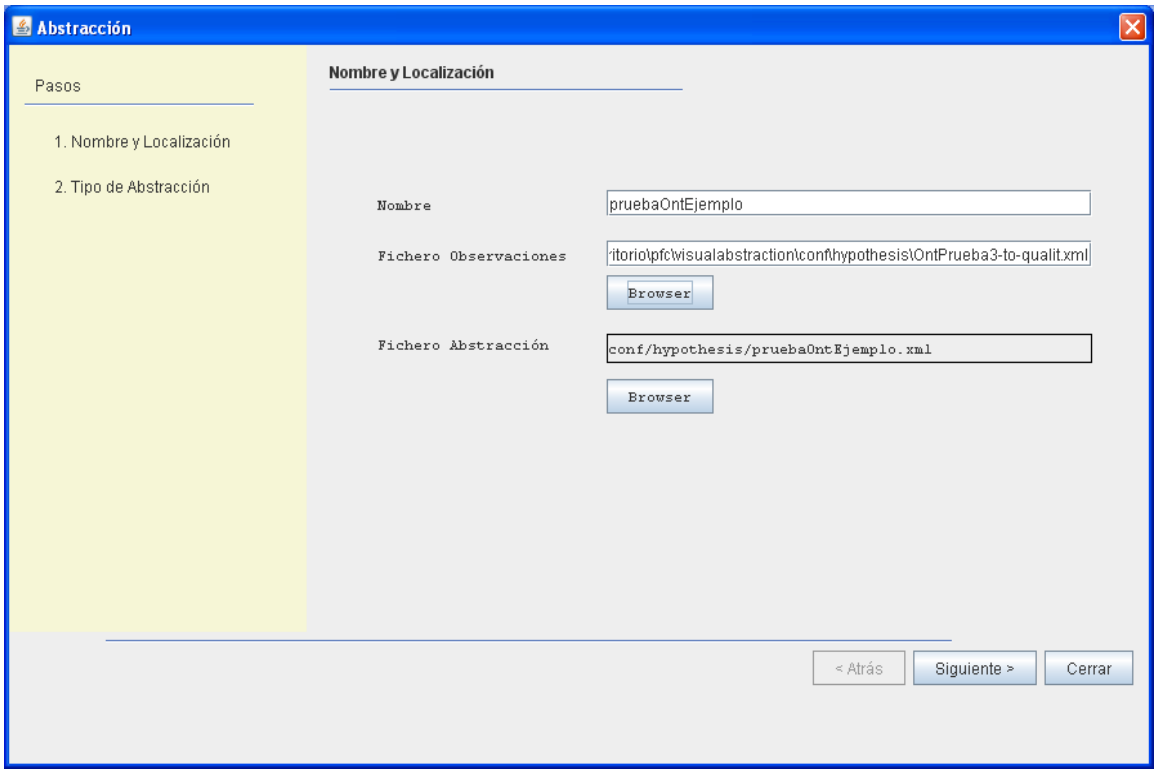


Ilustración 37: Nombre y localización de información temporal

En el siguiente paso se indica el tipo de abstracción, que para este ejemplo se usa la abstracción de estado, que es la que se ha definido. Se selecciona a continuación el botón *Ejecutar*.

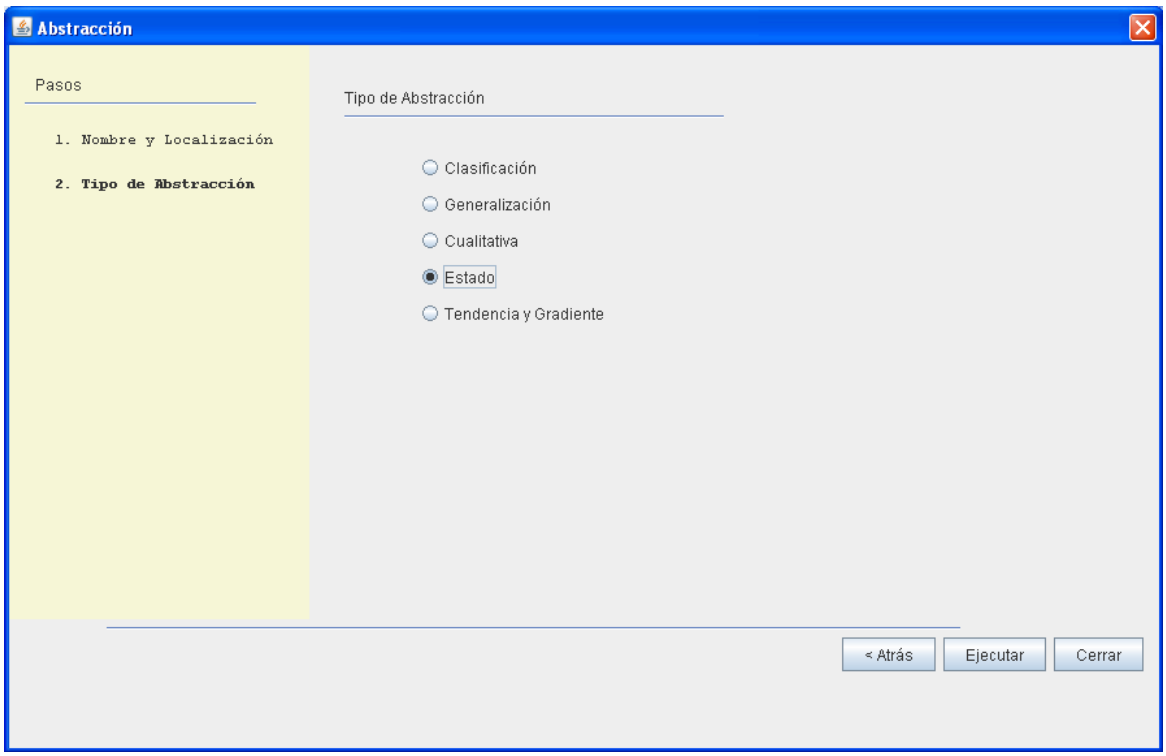


Ilustración 38: Tipo de abstracción

A continuación se muestra un cuadro de diálogo indicando si ha funcionado correctamente el proceso de abstracción y si se desea hacer una posterior visualización.

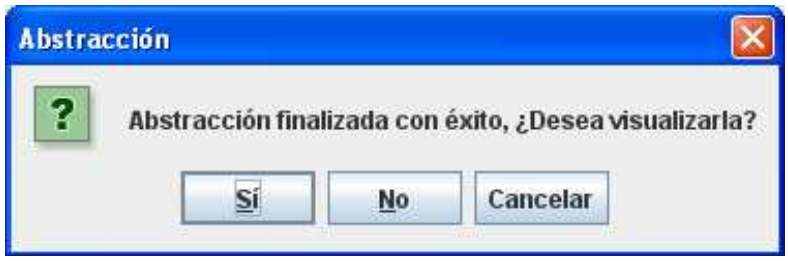


Ilustración 39: Confirmación de abstracción

Visualización

Para iniciar el proceso de visualización hay que seleccionar en el menú *Ejecutar/Ejecutar visualización* o en la barra de herramientas seleccionar el botón de *visualización*.

Ilustración 40: Ejecutar visualización

En la ventana que se abre, se selecciona la fuente de datos, que podrá ser un

XML o la base de datos y la ruta. Puesto que la abstracción generada se hizo persistente en formato XML se seleccionará esa fuente.

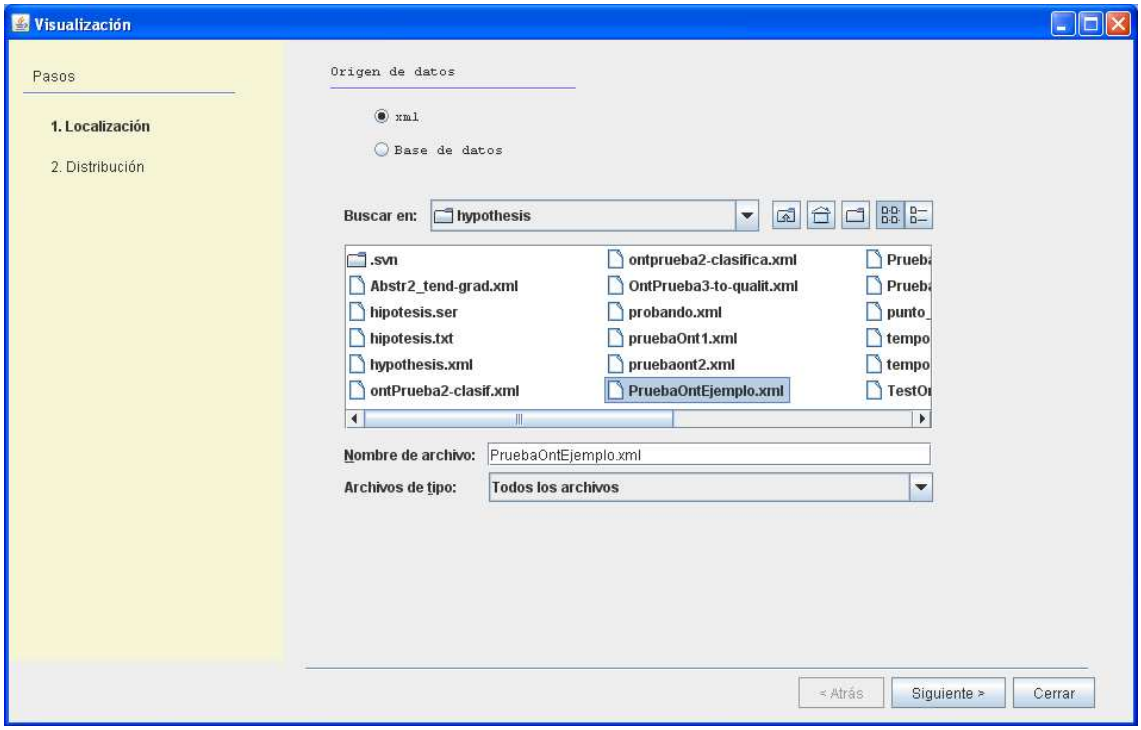


Ilustración 41: Origen de datos para la visualización

Para este ejemplo se elige visualizar todos los conceptos en una sola gráfica, de forma que se muestren las relaciones de abstracción.

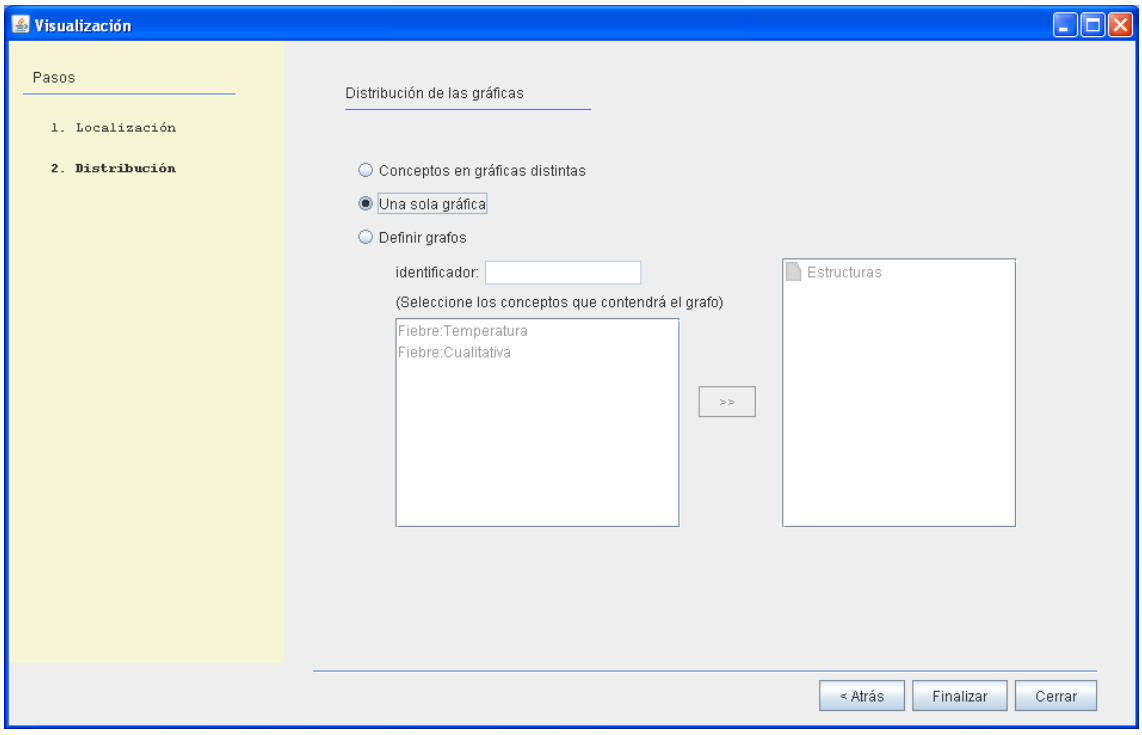


Ilustración 42: Seleccionar estructuras de visualización

Al finalizar la selección de los parámetros de visualización, se muestra en el panel principal el grafo.

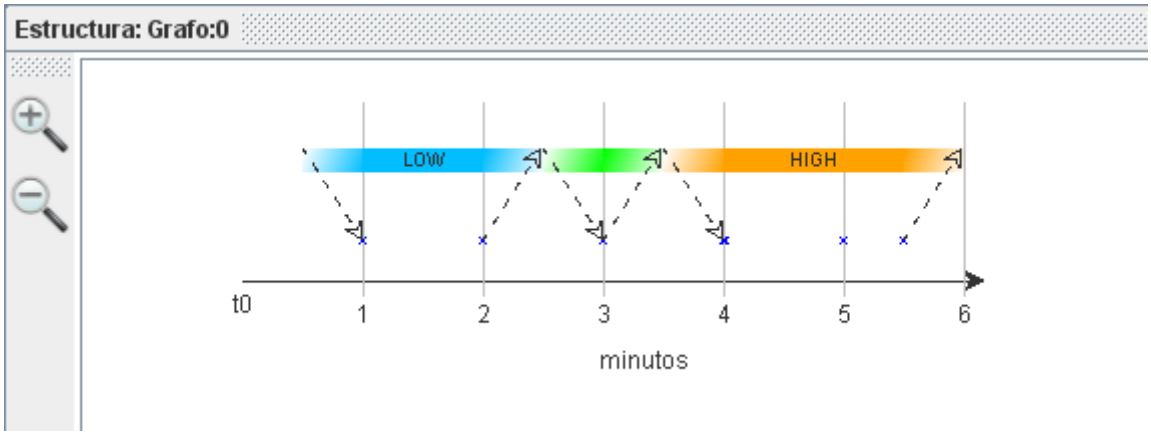


Ilustración 43: Visualización de los datos temporales

En la parte izquierda de la ventana principal se puede ver en forma jerárquica las estructuras y conceptos visualizados.

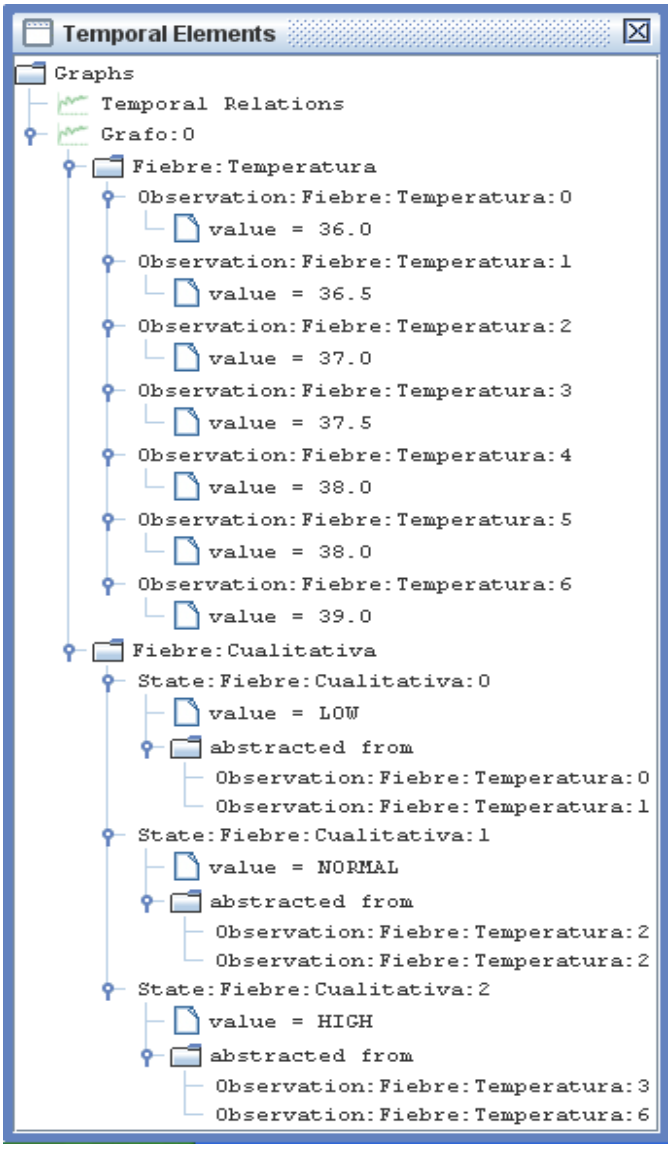


Ilustración 44: Árbol de información de la visualización

Por otra parte, podemos ver un ejemplo en el que se visualizan varios conceptos temporales en grafos separados, para lo que habrá que definirlos en la ventana de visualización.

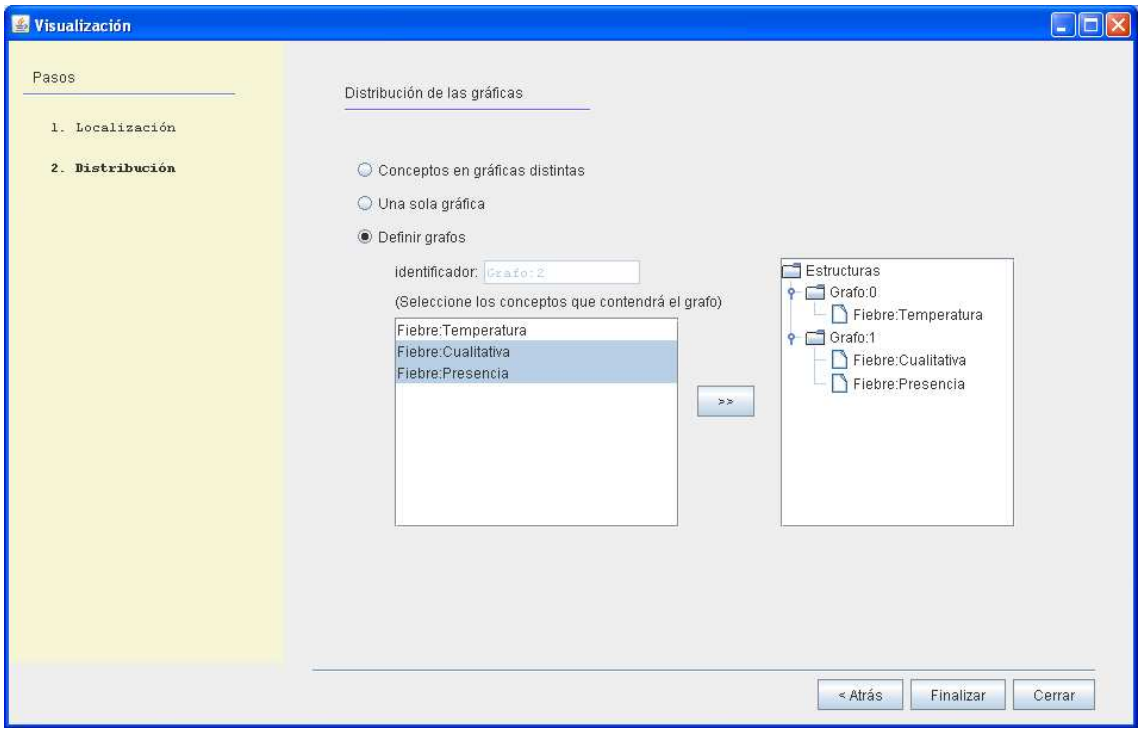


Ilustración 45: Definir varios grafos

La visualización que obtendremos contendrá dos estructuras dibujadas en dos gráficas distintas.

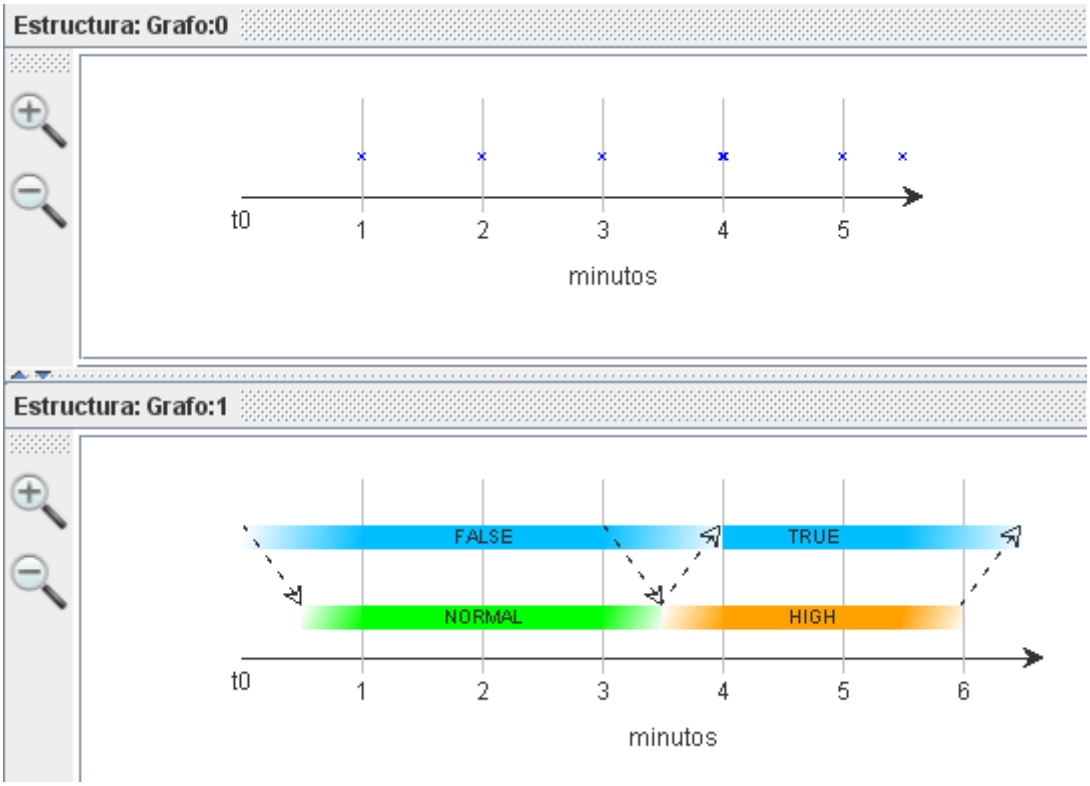
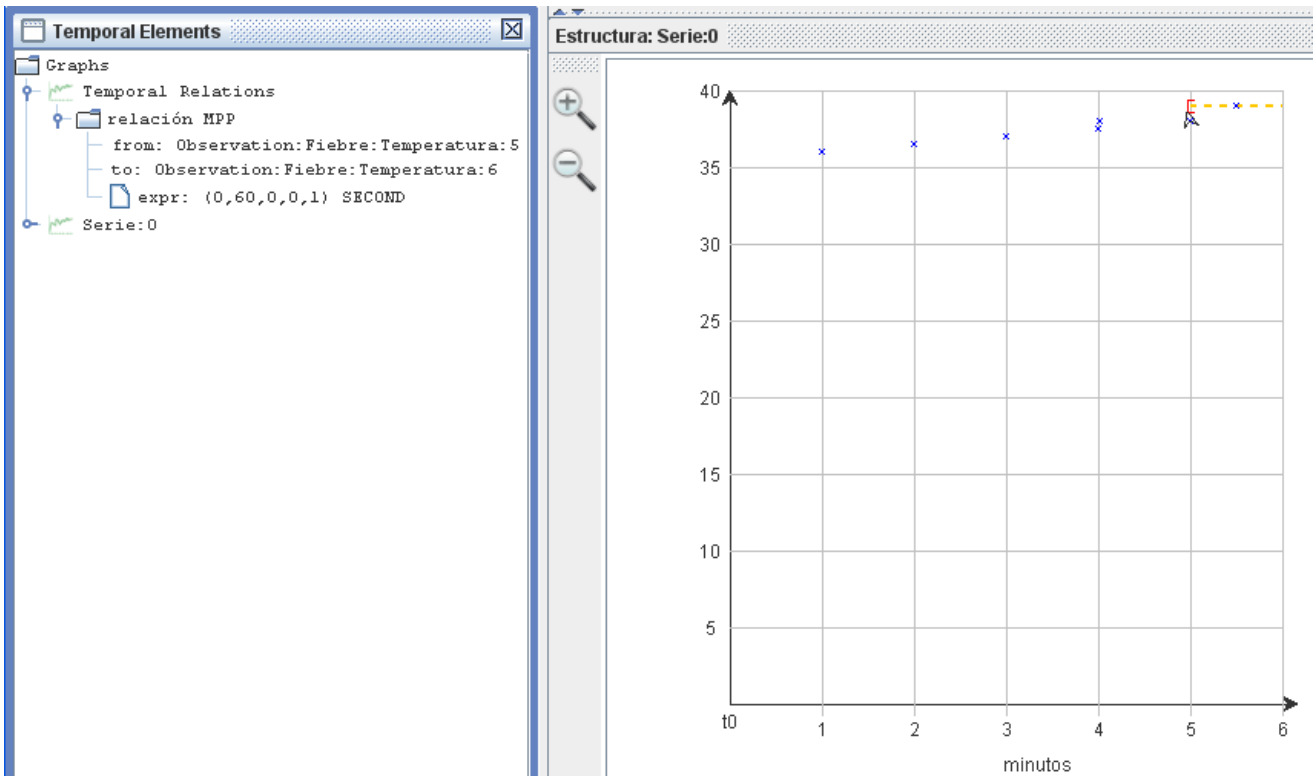


Ilustración 46: Visualización con dos grafos

Como último ejemplo se muestra cómo sería una gráfica en la que aparece un punto impreciso, el cual no posee marca temporal, por lo que se establece una relación MPP entre la observación 5 y la 6.



4. Conclusiones y vías futuras

En este proyecto se ha perseguido el objetivo de desarrollar un entorno de adquisición de conocimiento y visualización de abstracciones temporales. Para ello se han desarrollado tres módulos independientes, que conjuntamente con el módulo de abstracción, ya definido, conforma un sistema de ayuda. En este proyecto hemos orientado el trabajo al dominio clínico o biológico, pero ha sido desarrollado independientemente del dominio, de forma que se pueda aplicar a cualquier dominio donde el tiempo y la abstracción de datos temporales sean elementos representativos.

En un proceso de abstracción es esencial la adquisición de conocimiento, así como la representación y visualización de las abstracciones temporales. Estos aspectos constituyen el objetivo de éste proyecto.

Tras una primera fase de revisión bibliográfica y análisis del módulo de abstracción, junto con el estudio del modelo de abstracción en él definido, se inició una segunda etapa de desarrollo del prototipo, añadiéndole nuevas funcionalidades con el desarrollo de cada uno de los módulos que han sido implementados en este proyecto de forma independiente.

El módulo de adquisición de conocimiento se ha diseñado con el cometido de extraer conocimiento del usuario para hacerlo persistente en la base de conocimiento estática en forma de documento XML o para ser utilizado directamente por el módulo de abstracción, siendo esta base de conocimiento contenedora de la ontología del dominio. Durante la adquisición de conocimiento se guía al usuario mediante un asistente que facilite el proceso, haciéndole transparente la complejidad subyacente. En el módulo de adquisición, una parte fundamenta ha sido el desarrollo de una interfaz de comunicación con la base de conocimiento. Así, este módulo permite realizar adquisición sobre ontologías del dominio ya existentes, o simplemente para la recuperación de una ontología del dominio sobre la que se va a realizar el proceso de abstracción.

El módulo de representación temporal extendido se ha diseñado para representar la información temporal mediante estructuras de más alto nivel que la representación existente. En su desarrollo se ha incluido el diseño e implementación de la interfaz de comunicación, la cual se encarga de la interacción con el sistema de adquisición de datos, para después con los datos temporales construir la hipótesis. Se han propuesto dos implementaciones de la interfaz de comunicación, la primera interactúa con un sistema de adquisición de datos local implementado mediante archivos XML y en la segunda el sistema adquisición de datos se implementa mediante una base de datos.

El módulo de visualización se ha diseñado para que el usuario pueda definir como desea estructurar los datos a visualizar. Se han incluido datos puntuales, intervalos y relaciones temporales entre estos, cuyo dominio puede ser de tipo cuantitativo o cualitativo. Para ello se han implementado distintos modos de visualización para cada una de las representaciones definidas en el módulo temporal extendido.

Para versiones futuras del módulo de adquisición se podría tener en cuenta la integración con el servidor de Ontologías. Como se ha comentado, el módulo de adquisición se ha implementado de forma que interacciona con una base de conocimiento donde las ontologías están representadas por ficheros XML, quedando

abierto a que en una versión futura se integre con el servidor de ontologías.

Se podría mejorar la aplicación mediante la integración con el sistema de adquisición de datos, puesto que en el proyecto se ha implementado la comunicación con un sistema de adquisición de datos estático en formato XML y una base de datos local. Así pues, queda para sucesivas versiones la integración con el sistema de adquisición de datos real.

Otra opción de mejora sería extender la representación creando otros métodos de visualización que permitan mostrar de manera más intuitiva las relaciones borrosas.

5. Bibliografía

[Campos, 2007] M. Campos. Abstracción temporal basada en redes de restricciones. 2007

[Cheung, 1990] Y. Cheung y G. Stephanopoulos. Representation of process trends-Part I. a forma representation framework. *Computers and Chemical Engineering,* 14:495-510, 1990.

[Drakopoulos, 1998] J. A. Drakopoulos y B. Hayes-Roth. tFPR: A fuzzy and structural pattern recognition system of multi-variate time-dependent pattern clases base don sigmoidal functions. *Fuzzy Sets and Systems,* 99:57-72, 1998.

[Haimowitz, 1995] I.J. Haimowitz, P.P. Le y I.S. Kohane. Clinical monitoring using regressions-based trend templates. *Artificial Intelligence in Medicine, 7:473-496, 1995.*

[Lowe, 1999] A. Lowe, M.J. Harrison y R.W. Jones. Diagnostic monitoring in anaesthesia using fuzzy trend templates for matching temporal patterns. *Artificial Intelligence in Medicione,* 16:183-199, 1999.

[Larman, 2005] Craig Larman. Applying UML and Patterns. Prentice Hall, NJ, 2005.

[Otero, 2005] A. Otero: Un modelo de redes de restricciones borrosas para la abstracción y el reconocimiento sobre señales. Universidad de Santiago de Compostela, Octubre 2005.

[Shahar, 2004] Yuval Shahar, Dina Goren-Bar, Maya Galperin, David Boaz, and Gil Tahan. KNAVE-II: A Distributed Architecture for Interactive Visualization and Intelligent Exploration of Time-Oriented Clinical Data. 2004

[Steimann, 1996] F. Steimann. The interpretation of time-varying data with DIAMON-1. *Artificial Intelligence in Medicine*, 8:333-357, 1996.