UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

Trabajo de Fin de Grado

Especificación de una Ontología de Medidas para Internet

Autor

Daniel Michaud Vallinoto

Tutor

Jorge E. López de Vergara Méndez

Resumen

Hacer medidas de la red puede llegar ser desesperante. Existen muchos tipos distintos de medidas, programas, sistemas, bases de datos… Esto hace que la tarea de comparar los resultados de todas esas pruebas sea demasiado molesta y engorrosa. Para evitar esto, este proyecto define una ontología para uniformizar la forma en que se representan los datos que se obtienen de los distintos programas de medidas, y una plataforma para poder hacer consultas semánticas sobre todos ellos a la vez. Los resultados obtenidos se han aplicado en el proyecto Openlab del 7º programa marco de la UE.

Una ontología es una forma de representar el conocimiento como una serie de conceptos, dentro de un campo concreto (en nuestro caso las medidas de la red), utilizando un vocabulario compartido para especificar los tipos, propiedades y relaciones entre estos conceptos.
A efectos prácticos, la plataforma implementada integra un conjunto de bases de datos con una capa semántica por encima dando significado a todos los datos. Para hacer consultas en esta ontología, se utiliza el lenguaje SPARQL (***S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage)*.

Gracias a esta ontología y a ciertas herramientas, que permiten la correspondencia de cualquier base de datos relacional con la ontología, podemos tener una visión global de todas las medidas que hayamos tomado, de una manera transparente. Así, aunque cada medida puede estar almacenada en una base de datos distinta, se puede consultar con un vocabulario común y a través de una sola plataforma, llamada SPQR.

Palabras clave: Ontología, D2R, Medidas de internet, SPQR, SPARQL, estandarización, OML, Packet Tracking, Zabbix, TopHat, ETOMIC, iLab-t, nmVO

Abstract

Performing network metrics can be exhausting. There are several types of metrics, programs, systems, databases… That make the labor of contrasting all the results very hard. In order to avoid that, this project defines an ontology to uniform the way all this data from the different programs is represented and an unique platform to semantically query it all at once, applied in the Openlab project of the 7th UE framework.

An ontology formally represents knowledge as a hierarchy of concepts within a domain, using a shared vocabulary to denote the types, properties and interrelationships of those concepts.

In practical effects, the platform implemented integrate a set of databases with a semantic layer over them, giving a meaning at all the data. To query this data, we use SPARQL (***S**PARQL **P**rotocol **a**nd **R**DF **Q**uery **L**anguage)*.

With this ontology and some other tools, which allow the "mapping" between any relational database and our ontology, we are able to have a transparent, global view of all the data. Thus, although each measurement is in a different database, we can query it with a common vocabulary from one single platform, called SPQR.

Keywords: Ontology, D2R, internet measurements, SPQR, SPARQL, standardization, OML, Packet Tracking, Zabbix, TopHat, ETOMIC, iLab-t, nmVO

Agradecimientos

Quisiera agradecer a todos mis amigos, que me han apoyado desde el principio en esta carrera, incluso aquellos que me decían que la ingeniería no valía para nada, que hiciera un módulo.
A mi familia, por soportarme durante estos años, abstraído de las tareas del hogar y concentrado en mis estudios.
A mis compañeros que he conocido a lo largo de la carrera, junto a quienes he asistido y soportado las clases durante estos 4 años, y con quienes se hacía todo más llevadero.
A mis compañeros del laboratorio, a Rafael, a Alfredo, a Rubén… con quienes he pasado gran cantidad de horas y me han enseñado todas esas cosas que no se enseñan en la carrera.
A Mario que me ayudó cuando entre por primera vez en el laboratorio y fue mi guía en los proyectos que iba cogiendo.
Y por último, y no por ello menos importante, a Jorge, por tutelarme durante mi estancia en el laboratorio y en el TFG, por ayudarme en todos los problemas que me surgían y por permitirme esa flexibilidad de horario que ha hecho mucho más fácil compenetrar los estudios con el trabajo en el laboratorio.
A todos, gracias.

Índice

Índice de figuras

Índice de tablas

Glosario

SPARQL: SPARQL Protocol and RDF Query Languaje
RDF: Resource Description Framework
SPQR: SPARQL pre-Processor and Query Rewriter
SMR: Semantic Metadata Registry
TFG: Trabajo de Fin de Grado
SQL: Structured Query Language
XML: eXtensible Markup Language
ETSI: European Telecommunications Standards Institute
MOI: Measurement Ontology for IP traffic
ANME: Advanced Network Monitoring Equipment
ETOMIC: European Traffic Observatory Measurement InfraestruCture
UPMC: Université Pierre et Marie Curie
OMF: cOntrol and Management Framework
OML: OMF Measurement Library
MOMENT: Monitoring and Measurement in the Next generation Technologies
CSV: Comma-Separated Values
JDBC: Java DataBase Connectiviy

1 Introducción

1.1 Motivación

Cada vez las redes están más presentes en nuestras vidas y nuestro entorno. Cada vez hay más dispositivos conectados entre sí, ordenadores, móviles, tabletas, electrodomésticos… Conectados de una parte del mundo a otra y que necesitan hacer uso de protocolos de red para comunicarse entre ellos. Para asegurarse de que toda esta red de comunicaciones funciona, hay que realizar mediciones y comprobar que todos los valores relativos a la calidad de servicio son correctos mediante distintos programas.

Estos programas nos permiten medir el ancho de banda entre dos puntos, la latencia, la cantidad de saltos que realiza un paquete entre dos puntos, entre otras cosas, guardando todos los datos obtenidos de muy diversas formas, como bases de datos, ficheros de texto, csv, etc.

Entre otras cosas, esto es lo que hace el proyecto OpenLab[2][3], que proporciona testbeds para poder experimentar nuevos protocolos y aplicaciones en entornos controlados, y parte de dichos experimentos también requieren de medidas.
El problema surge a partir de toda esta variedad de testbeds y medidas. Cuando un usuario quiere hacer dichas medidas, y se encuentra con que, para contrastar resultados, tiene que consultar distintas bases de datos, ficheros, etc.

En este contexto surge este trabajo fin de grado, focalizado en definir una ontología[1] donde representar todos estos datos de distintos orígenes en una sola plataforma y poder consultarlos todos a la vez, mediante un solo lenguaje, el SPARQL.

Esta ontología está compuesta de distintas clases, subclases y propiedades que permiten representar toda esta información de una manera transparente y unificada.

Para que esto funcione, hacen falta varias herramientas. Por un lado, necesitamos un programa que haga de puente entre la ontología y las bases de datos que almacenan las medidas de red. Un programa que, a partir de ciertas reglas de correspondencia, sea capaz de traducir a términos de la ontología, dando un significado a los datos de las bases de datos relacionales. Este programa es el servidor D2R[4].
Por otro lado, también es necesaria una base de datos intermedia temporal, donde se guardan los datos consultados antes de ser visualizados por la aplicación final. Esta puede ser una base de datos cualquiera.

Por último, es oportuno disponer de una plataforma a partir de la cual se acceda a todos estos datos. Una plataforma que se pueda configurar para añadir nuevas bases de datos asociadas y poder consultarlas todas de una manera transparente y global. Esto es el sistema SPQR[5], que se encarga de hacer toda la gestión de los servidores D2R y de hacer las consultas mediante SPARQL. Incluso tiene varias consultas preestablecidas, para aquellos que no sepan SPARQL.

1.2 Objetivos

El objetivo de este trabajo de fin de grado poder trabajar con repositorios de red heterogéneos y distribuidos, y para ello se define la ontología con conceptos que abarcan todo tipo de medidas de red, ya sean ancho de banda, latencia, datos sobre la localización de los nodos, etc. Y se establece la plataforma SPQR para poder acceder a todas ellas de una manera global y transparente.

En concreto, este objetivo se ha plasmado en las siguientes tareas:

- Por un lado, hemos incrementado el número de herramientas integradas con el sistema mediante archivos de correspondencia y el servidor d2r, mencionado anteriormente.

- Por otro lado, para poder integrar estas herramientas, hemos ampliado la cantidad de conceptos y elementos de la ontología.

- Finalmente, una de las bases de datos ya integradas anteriormente (la base de datos TopHat) ha optimizado las consultas sobre una de las tablas, ya que antes la consulta tardaba minutos en completarse, ahora tarda sólo algunos segundos mediante el uso de un procedimiento almacenado. Para este caso, hemos tenido que modificar el código del servidor d2r para permitir "mapear" el procedimiento almacenado donde antes estaba la tabla.

1.3 Estructura del documento

El resto del documento se organiza según se explica a continuación:

Primero se realizará una exposición del Estado del arte, donde se mostrará qué trabajo hay ya realizado y así poder relacionarlo con el trabajo desarrollado en este TFG.

Después se explicarán las distintas líneas de desarrollo que ha seguido este trabajo:

- Nuevas herramientas integradas: Se mostrará cada una de las nuevas herramientas integradas en el sistema, así como los valores nuevos que estas aportan a la ontología.

- Ampliación de la ontología: Se hará un resumen de los nuevos valores que han tenido que añadirse a la ontología para poder implementar las nuevas herramientas.

- Ampliación de la funcionalidad del servidor d2r: El sistema tophat, ya integrado anteriormente, ha cambiado una de sus tablas por un procedimiento almacenado, por lo que hay que adaptar el servidor d2r para poder hacer correspondencias con dichos procedimientos.

Finalmente, se extraerán conclusiones de todo el trabajo realizado, lo aprendido con el desarrollo del mismo, lo que ha aportado al sistema, y cuáles pueden ser las líneas de trabajo futuro.

2 Estado del arte

2.1 Introducción

En esta sección se describirá en qué consisten las ontologías y el SPARQL, se definirá la estructura de la ontología de medidas de red, y se hará una introducción a las distintas herramientas de las que se compone el proyecto. Así se podrá entender mejor cómo funciona el sistema. También se hará una evaluación de las aportaciones que se hacen al proyecto Openlab.

2.2 Ontologías y SPARQL

Para resolver el problema de la heterogeneidad de representaciones de información, es necesario proveer una visión unificada de cada infraestructura de medida, haciendo posible que cualquier investigador pueda obtener medidas de red de cualquier repositorio o herramienta y combinarlas, independientemente de cómo han sido medidas o guardadas.

Para alcanzar este fin, hay distintos métodos posibles. El primero es usar SQL. En tal caso, es necesario definir un esquema común para todas las bases de datos diferentes. El problema es cuando una consulta tiene que hacerse sobre las distintas implementaciones de SQL (SQLServer, MySQL, PostgreSQL, SQLite…) puede haber problemas de compatibilidades. Otra solución es usar XML, pero su limitación es que se queda en el aspecto sintáctico de los datos. Finalmente, decidimos que la mejor decisión era usar una ontología.

Una ontología formalmente representa conocimiento en forma de un conjunto de conceptos, su taxonomía y sus relaciones. Las ontologías tienen varias aplicaciones, las cuales incluyen una manera de especificar el vocabulario para dialogar entre agentes inteligentes, así como el significado del vocabulario. Además, la web semántica usa ontologías para especificar el significado de las páginas web, incluso han sido muy útiles con la emergencia de los servicios web, para definir claramente el significado de las interfaces de estos servicios.

Para poder consultar la ontología, se hace uso del SPARQL[6]. El SPARQL (*SPARQL Protocol and RDF Query Language*) es un lenguaje estandarizado para la consulta de grafos RDF, que son las definiciones semánticas que contiene la ontología. Este lenguaje realiza las consultas preguntando por tres parámetros: el sujeto, el predicado y el objeto. Por ejemplo:

```
PREFIX dc: http://purl.org/dc/elements/1.1/>
SELECT ?title
WHERE { http://ejemplo.org/libros dc:title ?title }
```

Esta consulta pregunta por todos los libros (sujeto), que contengan el atributo o significado dc:title (predicado) y que los devuelva en ?title (objeto).

2.3 Ontología de medidas de red

El proyecto parte de la ontología desarrollada por la ETSI en el grupo de trabajo MOI. La estructura básica de la ontología es la siguiente.

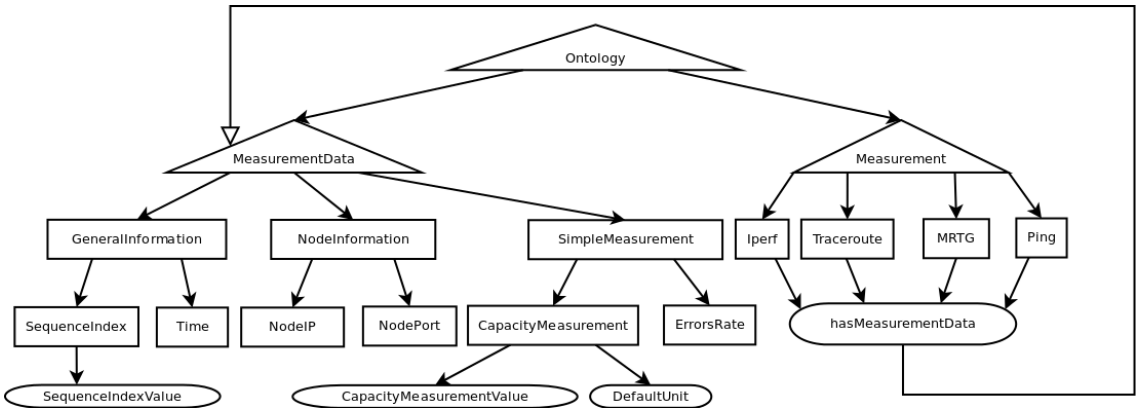

Figura 1 Estructura de la ontología

La ontología está dividida en dos partes, medidas (Mesurement), que indican el tipo de medida realizada (iperf, traceroute, etc.) y los datos en sí (MeasurementData). Estas dos partes se relacionan mediante la propiedad "hasMeasurementData", que asocia cada dato con el tipo de medida al que pertenece. Otra imagen que explica de una manera bastante clara la estructura de la ontología es la siguiente.

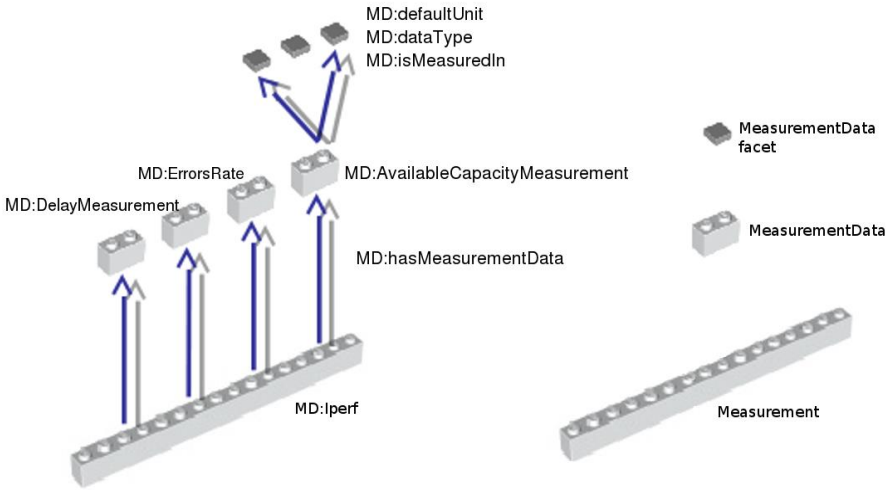

Figura 2 Estructura de la ontología 2

Aquí se puede ver que las medidas (Iperf en este caso) contienen datos de medición (MeasurementData, AvailableCapacityMeasurement, en este caso) y estos datos tienen propiedades que los definen (dataType, unidad,etc.).

2.4 Descripción de las herramientas utilizadas

El funcionamiento del sistema se puede observar en la siguiente figura.

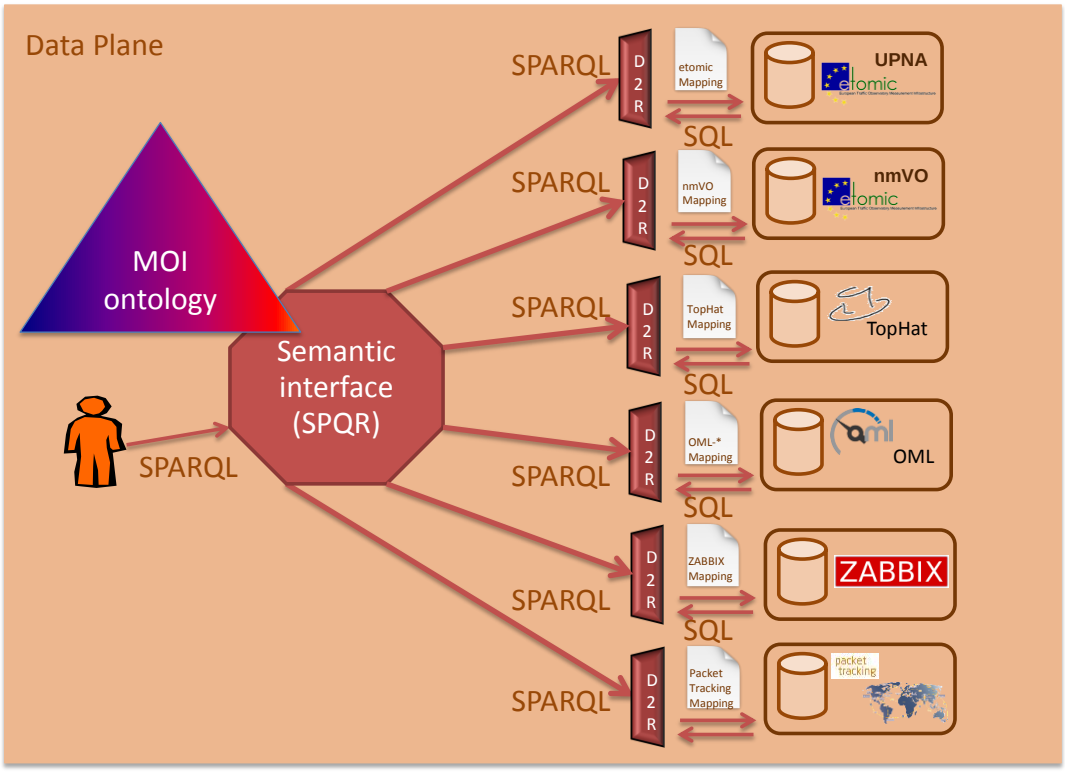

Figura 3 Funcionamiento del sistema

La interfaz SPQR, a partir de los conceptos de la ontología, consulta (mediante SPARQL) en cada servidor d2r, los cuales consultan (mediante SQL) cada base de datos.

Para que todo el sistema funcione, necesitamos hacer uso de varias herramientas:

- SPQR: El sistema SPQR es el que permite hacer las consultas sobre todas las bases de datos. Se despliega en un servidor glassfish y hace uso de SMR (descrito abajo). Este sistema, tiene varias consultas predefinidas (para aquellos no acostumbrados al uso de SPARQL, pero también permite escribir consultas a mano.

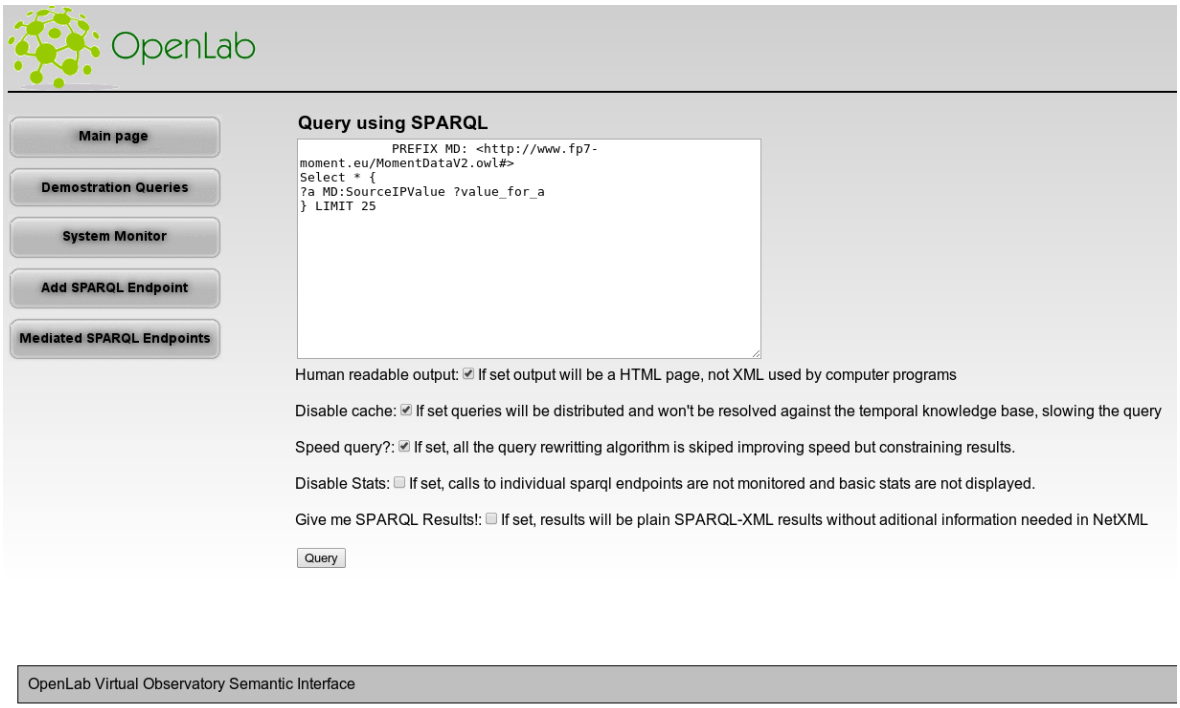

Figura 4 SPQR

- SMR: Esta es una base de datos intermedia donde se almacenan los datos consultados. En ella se guarda una especie de caché, la cual consulta el SQPR antes de consultar los sistemas acoplados.
- d2r-server: Este servidor permite, a través de un fichero con las reglas de correspondencias entre la base de datos relacional y los conceptos de la ontología , hacer consultas sobre dicha base de datos relacional como si de una ontológica se tratara. Estas reglas de correspondencia funcionan de la siguiente manera: Se realiza la correspondencia de una tabla con una clase de medida, se definen las propiedades y relaciones entre conceptos y finalmente se relaciona la tabla con el valor semántico de la ontología. A partir de ese fichero, el servidor d2r se inicia, en la máquina, ofreciendo una interfaz a partir de la cual realizar consultas en SPARQL.

Una vez tenemos desplegados el SMR y el SPQR, sólo tenemos que iniciar instancias de d2r-server para ir añadiendo datos a la ontología. El sistema hará consultas sobre tantas bases de datos como haya vinculadas.

2.5 Bases de datos integradas

- Antes de empezar a ampliar el catálogo de herramientas integradas en el proyecto, este contaba con las siguientes: Etomic[7]: La Infraestructura del Observatorio de Medición de Tráfico Europeo es una infraestructura de medición de alta precisión distribuida a través de Europa, que es capaz de hacer medidas entre las cajas ANME especialmente diseñadas por OneLab que están globalmente sincronizadas a una alta

resolución temporal.

- nmVO[8]: El observatorio virtual de mediciones de red es un repositorio de datos y una plataforma de compartición de datos bajo la infraestructura de ETOMIC.
- redirisMRTG[9]: RedIRIS es la red académica y de investigación española y proporciona servicios avanzados de comunicaciones a la comunidad científica y universitaria nacional.
- TopHat[10]: El proyecto TopHat, iniciado por UPMC durante el proyecto OneLab, tiene como objetivo desarrollar plataformas y herramientas para medir activamente la topología de internet, y proporcionar en vivo e históricamente medidas a la comunidad.

2.6 Conclusiones y trabajo a realizar

El trabajo que se realiza en este trabajo fin de grado sobre el sistema descrito en las secciones anteriores consiste en la ampliación de herramientas integradas, con nuevas bases de datos de medidas de red, y además aumentar la funcionalidad para poder aceptar nuevas formas de datos que antes no admitía, así como obtener los datos desde ficheros csv (no solo de bases de datos estándar) o modificar el código para permitir hacer reglas de correspondencia con procedimientos almacenados.
Los capítulos siguientes presentan en detalle el trabajo realizado.

3 Nuevas bases de datos integradas

3.1 Introducción

Gran parte del trabajo de este proyecto ha consistido en aumentar la funcionalidad de la ontología y el abanico de bases de datos y sistemas integrados con ella, por lo que se integraron varias bases de datos y sistemas de almacenamiento diversos para cumplir este fin.

Han sido integrados tanto nuevas bases de datos típicas (PostgreSQL, MySql), como otras menos comunes (h2), como sistemas de ficheros en csv.

- En este capítulo presentaremos todas las herramientas nuevas que han sido integradas y qué valores han aportado a la ontología, así como si estos ya existían en ella o han tenido que ser creados: iLab-t:
  - wifilinktestchan1
  - zigbeelinktest1
- OML
  - iperf
  - trace
- PacketTracking
  - csv
  - h2
- Zabbix

Y al final se expondrán las conclusiones del trabajo realizado.

3.2 Nuevas bases de datos integradas

3.2.1 iLab-t

iLab-t[11] proporciona un testbed para realizar pruebas sobre redes inalámbricas, por lo que nos viene muy bien añadirlo en la ontología, ya que aumenta en gran cantidad la diversidad de datos que la ontología puede representar.
Este servidor trae dos bases de datos nuevas que añadir al proyecto: wifilinktestchan1 y zigbeelinktext1.
La primera almacena datos de bajo nivel en el campo de la red.

Esta base de datos contiene sólo una tabla, phytest_phytestmp, cuyos datos relevantes son los siguientes:

Columna de la tabla	Valor en la ontología	¿Valor nuevo?
Txpower	MD:PowerMeasurement	Sí
cannel	MD:Channel	Sí
Signal	MD:Signal	Sí
seq_no	MD:SequenceIndex	No
Hostname	MD:NodeName	No
Timestamp	MGC:Timestamp	No
sender_mac	MD:SourceMacAddress	No
Rate	MD:Rate	Sí
Wlansender	MD:SenderInterface	Sí
Wlanreceiver	MD:ReceiverInterface	Sí

Tabla 1 iLab-t wifilinktestchan1 phytest_phytestmp

La otra base de datos, zigbeelinktest1, contiene datos sobre ondas de radio a bajo nivel, con varias tablas para cada frecuencia de onda. No obstante, no nos pidieron implementar todas las columnas de cada tabla, sino sólo algunas de sf_sfdata_am80:

Columna de la tabla	Valor en la ontología	¿Valor nuevo?
ReportMsg_NA_r1_txPower	MD:PowerManagement	Sí
ReportMsg_NA_txPower	MD:PowerManagement	Sí
ReportMsg_NA_r1_minLqi	MD:LQIMeasurement	Sí
ReportMsg_NA_r1_avgLqi	MD:LQIMeasurement	Sí
ReportMsg_NA_r1_maxLqi	MD:LQIMeasurement	Sí
ReportMsg_NA_r1_minRssi	MD:RssiMeasurement	Sí
ReportMsg_NA_r1_avgRssi	MD:RssiMeasurement	Sí
ReportMsg_NA_r1_maxRssi	MD:RssiMeasurement	Sí
ReportMsg_NA_minEstimatedNoiseFloor	MD:NoiseFloorMeasurement	Sí
ReportMsg_NA_avgEstimatedNoiseFloor	MD:NoiseFloorMeasurement	Sí
ReportMsg_NA_maxEstimatedNoiseFloor	MD:NoiseFloorMeasurement	Sí

oml_seq	MD:SequenceIndex	No
oml_ts_client	MGC:Timestamp	No
oml_ts_server	MGC:Timestamp	No
ReportMsg_NA_txErrorCounter / ReportMsg_NA_txCounter	MD:PacketLossMeasurement	No
ReportMsg_NA_r1_numberOfPacketsLost / ReportMsg_NA_r1_numberOfPacketsReceived	MD:PacketLossMeasurement	No

Tabla 2 iLab-t zigbeelinktest1 sf_sfdata_am80

3.2.2 OML

OML[12] es una herramienta de instrumentación que permite a los desarrolladores de aplicaciones definir puntos de medición configurables dentro de nuevas o pre-existentes aplicaciones. Los experimentadores que ejecutan las aplicaciones pueden dirigir los flujos de mediciones desde estos puntos de medición a puntos de recolección remotos, para almacenar en bases de datos de medición.

En este caso, las aplicaciones pre-existente sobre las que se han realizado las correspondencias son Iperf y Trace.

Iperf:

Iperf es una aplicación que permite hacer mediciones del ancho de banda, latencia, jitter, % de paquetes transferidos, etc. punto a punto, ejecutando un programa en el origen y otro en el destino, y aparte la sonda OML que recibe los datos para guardarlos en la base de datos. Estos datos son exportados a una base de datos sqlite3, que es la que mapeamos para la ontología. Se compone de lo siguiente:

Tabla iperf_transfer:

olumna de la tabla	**Valor en la ontología**	**¿Valor nuevo?**
oml_seq	MD:SequenceIndex	No
size * 8 / (end_interval-begin_interval)	MD:AvailableCapacityMeasurement	No

Tabla 3 OML Iperf iperf_transfer

Tabla iperf_losses:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
lost_datagrams/total_datagrams	MD:PacketLossMeasurement	No

Tabla 4 OML Iperf iperf_losses

Tabla iperf_connection:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
local_address	MD:SourceIP	No
local_port	MD:SourcePort	No
remote_address	MD:DestinationIP	No
remore_port	MD:DestinationPort	No

Tabla 5 OML Iperf iperf_connection

Tabla iperf_application:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
starttime_s	MGC:Timestamp	No
starttime_us	MGC:Timestamp	No

Tabla 6 OML Iperf iperf_application

Tabla iperf_jitter:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
Jitter	MD:DelayVariationMeasurement	No

Tabla 7 OML Iperf iperf_jitter

Tabla iperf_settings:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No

bind_address	MD:BindIP	Sí
Multicast	MD:isListeningMulticast	Sí
multicast_ttl	MD:FinalTtlMeasurement	No
transport_protocol	MGC:TransportProtocol	No

Tabla 8 OML Iperf iperf_settings

Tabla iperf_packets:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
packet_id	MD:PacketIdentifier	No
packet_size	MD:PacketSize	No
packet_time_s	MGC:Timestamp	No
packet_time_us	MGC:Timestamp	No
packet_sent_time_s	MGC:Timestamp	No
packet_sent_time_us	MGC:Timestamp	No

Tabla 9 OML Iperf iperf_packets

Trace: Esta herramienta exporta datos sobre el estado de las conexiones tcp y udp. Al igual que Iperf, exporta los datos a una base de datos sqlite3.

Tabla oml2_trace_ip:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
oml_seq	MD:SequenceIndex	No
Pktid	MD:PacketIdentifier	No
ip_id	MD:IpIdentifier	No
ip_ttl	MD:FinalTtlMeasurement	No
ip_proto	MD:TransportProtocol	No
ip_src	MD:SourceIP	No
ip_dst	MD:DestinationIP	No
ip_sizeofpacket	MD:PacketSize	No

Tabla 10 OML Trace oml2_trace_ip

Tabla oml2_trace_tcp:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
packet_id	MD:PacketIdentifier	No
tcp_source	MD:SourcePort	No
tcp_dest	MD:DestinationPort	No
tcp_seq	MD:SequenceIndex	No
tcp_packet_size	MD:PacketSize	No
tco_ts	MGC:Timestamp	No

Tabla 11 OML Trace oml2_trace_tcp

Tabla oml2_trace_udp:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
packet_id	MD:PacketIdentifier	No
udp_source	MD:SourcePort	No
udp_dest	MD:DestinationPort	No
udp_ts	MGC:Timestamp	No

Tabla 12 OML Trace oml2_trace_udp

3.2.3 Packet Tracking

Packet Tracking[13] es una herramienta que sirve para seguir el trayecto de los paquetes a lo largo de un conjunto de nodos de una red, instalando sondas en cada nodo, y luego nos da una representación gráfica de dicho trayecto. Los datos exportados se guardan de dos maneras diferentes. Por un lado, en un fichero csv, el cual añadimos a la ontología mediante un "csv jdbc" adaptado para funcionar con estos datos, y por otro lado una base de datos h2.

Este jdbc hubo que adaptarlo porque no funcionaba bien con los datos que exportaba la herramienta. El Packet Tracking exportaba los csv con filas comentadas (precedidas por '#'), que en el jdbc no sabía cómo tratar y daba errores. Hubo que cambiar su analizador sintáctico para que omitiera esas líneas. Una vez hecho esto, el servidor cargaba perfectamente los datos.

Csv:

Tabla PtMin:

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
ipTTL	MD:FinalTtlMeasurement	No
observationTimeMicroseconds	MGC:Timestamp	No

Tabla 13 Packet Tracking csv PtMin

Tabla PtInterfaceStats_sent

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
interfaceName	MD:Interface	No
interfaceDescription	MD:NodeIpValue	No
observationTimeMilliseconds	MGC:Timestamp	No
exportTime	MGC:Timestamp	No

Tabla 14 Packet Tracking csv PtInterfaceStats_sent

Tabla PtInterfaceStats_received

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
interfaceName	MD:Interface	No
interfaceDescription	MD:NodeIpValue	No
observationTimeMilliseconds	MGC:Timestamp	No
exportTime	MGC:Timestamp	No

Tabla 15 Packet Tracking csv PtInterfaceStats_received

Tabla PacketTrackRecord

olumna de la tabla	Valor en la ontología	¿Valor nuevo?
pktId	MD:PacketIdentifier	No
Size	MD:PacketSize	No
Protocol	MD:Protocol	No
ts1	MGC:Timestamp	No

ttl1	MD:FirstTtlMeasurement	No
ts2	MGC:Timestamp	No
ttl2	MD:FinalTtlMeasurement	No

Tabla 16 Packet Tracking csv PacketTrackRecord

h2:

Tabla INTERFACE_STATS

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
INTERFACE_NAME	MD:Interface	No
PCAP_STAT_RECV	MD:PacketsCount	No
INTERFACE_DESCRIPTION	MD:NodeIP	No
TIMESTAMP	MGC:Timestamp	No

Tabla 17 Packet Tracking h2 INTERFACE_STATS

Tabla NODE_VIEW_PROPERTIES:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
LABEL	MD:NodeName	No

Tabla 18 Packet Tracking h2 NODE_VIEW_PROPERTIES

Tabla PROBE:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
LABEL	MD:NodeIP	No

Tabla 19 Packet Tracking h2 PROBE

Tabla PROBE_STATS:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
TIMESTAMP	MGC:Timestamp	No

Tabla 20 Packet Tracking h2 PROBE_STATS

Tabla RAW_TRACK_DATA:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
TS	MGC:Timestamp	No
TTL	MD:FirstTtlMeasurement	No

DELAY	MD:OneWayDelayMeasurement	No
PACKET_ID	MGC:PacketIdentifier	No

Tabla 21 Packet Tracking h2 RAW_TRACK_DATA

Tabla TRACK_DATA:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
START_TS	MGC:Timestamp	No
START_TS	MGC:Timestamp	No
DELAY	MD:OneWayDelayMeasurement	No

Tabla 22 Packet Tracking h2 TRACK_DATA

Tabla WAY_POINT:

lumna de la tabla	Valor en la ontología	¿Valor nuevo?
LONGITUDE	MD:LongitudeCoordinate	No
LATITUDE	MD:LatitudeCoordinate	No

Tabla 23 Packet Tracking h2 WAY POINT

3.2.4 Zabbix

Zabbix[14] es un sistema de monitorización del funcionamiento de sistemas. Se instala un programa cliente en cada máquina a medir y otro servidor en la máquina desde la que se monitoriza todo. Estas sondas pueden configurarse para medir desde uso de cpu o memoria hasta cantidad de paquetes enviados/recibidos.
Exporta todos los datos en una base de datos PostgreSQL, almacenando los valores de los números siempre en la misma columna de la misma tabla, conociendo el significado de dichos valores mediante el contenido de otra, por lo que no se realiza un mapeo directo, si no que con condiciones en función a la columna que tiene la descripción ("key_").
Este hecho ha generado muchos problemas, ya que no se podía hacer las correspondencias tradicionales de tabla.columna -> significado, ya que dependía del valor de esa columna. La solución implantada fue hacer asociaciones para todos los posibles valores que podía tener esa columna y que se asignaran en función a la cadena que estaba contenida en ese campo.

Tabla items:

key_	Valores a aprovechar	¿Valor nuevo?
net.if.out	MD:Interface	No
net.if.in	MD:Interface	No

net.if.collisions	MD:Interface	No
net.tcp.listen	MD:NodePort	No
net.tcp.port	MD:NodePort	No
	MD:NodeIP	No
net.tcp.service[.perf]	MD:NodePort	No
	MD:NodeIP	No
	MD:ApplicationProtocol	No
net.dns[.record]	MD:NodeIP	No

Tabla 24 Zabbix items

3.3 Conclusiones

Haber incluido todas estas herramientas en el sistema ha hecho que crezca la ontología y ha aumentado la cantidad de datos que puede albergar, sobre todo gracias a la implementación de las bases de datos de iLab-t que han proporcionado muchos datos de bajo nivel de la capa wifi.

Esto es muy importante, ya que ayuda directamente a la finalidad del proyecto, de albergar todas las herramientas posibles para estandarizar las herramientas de medidas de red.

Además ha sido muy instructivo y ha aportado diversidad al proyecto, ya que ha habido que trabajar con distintas herramientas, para adaptar el jdbc, trabajar con expresiones de cadenas… Nunca se sabe con qué se va a tener que lidiar a la hora de tener que incluir una nueva herramienta en el sistema.

Para poder incluir todas estas herramientas en el proyecto, ha hecho falta hacer cambios en la ontología, que durante este capítulo sólo se ha nombrado por encima. Por ello, en el siguiente capítulo se hará énfasis en estos cambios producidos en la ontología.

4 Ampliación de la ontología

4.1 Introducción

Debido al aumento de herramientas integradas en la ontología, ha sido necesario ampliar la información contenida en esta. Además, también ha cambiado el nombre del proyecto, por lo que también ha habido que cambiar los prefijos generales de la ontología.

A lo largo del capítulo se hará una exposición de los cambios ordenados por: cambios generales, cambios por cada herramienta y finalmente cambios menores.

4.2 Cambios realizados

4.2.1 Cambios generales

- Cambio en "Metadata":
  Estos cambios fueron necesarios para hacer el cambio de nombre del proyecto. Cambiamos los nombres de los prefijos para que en vez de representar el antiguo proyecto MOMENT, representen el nuevo ETSI MOI[15]. Antes los prefijos eran:
  - MD: http://www.fp7-moment.eu/MomentDataV2.owl#
  - MGC: http://www.fp7-moment.eu/MomentGeneralConcepts.owl#
  - MM: http://www.fp7-moment.eu/MomentMetadata.owl#
  - MO: http://www.fp7-moment.eu/Moment.owl#
  - MU: http://www.fp7-moment.eu/MomentUnits.owl#
  - Tools: http://www.fp7-moment.eu/tools.owl#

  Ahora están:
  - MD: http://www.etsi.org/isg/moi/data.owl#
  - MGC: http://www.etsi.org/isg/moi/GeneralConcepts.owl#
  - MM: http://www.etsi.org/isg/moi/Metadata.owl#
  - MO: http://www.etsi.org/isg/moi.owl#
  - MU: http://www.etsi.org/isg/moi/Units.owl#
  - Tools: http://www.etsi.org/isg/moi/tools.owl#

4.2.2 Cambios con la integración de iLab-t

- Cambios en "OWLClasses":

  - Creados dos hijos en MD:Measurement/MD:NetworkObject/MD:interface: MD:ReceiverInterface y MD:SenderInterface

- Añadido hijo para almacenar datos de WLAN en MD:MeasurementData/MD:SimpleMeasurement: MD:WLANMeasurement con hijos: MD:Channel, MD:LQIMeasurement, MD:NoiseFloorMeasurement, MD:PowerMeasurement, MD:RSSIMeasurement. Esta ampliación es muy importante, ya que originalmente las medidas que se representaban eran para IP, pero con estas incorporaciones la ontología llega ahora a niveles inferiores.

- Cambios en "Properties":
  - Datatypes:
    - Creados los siguientes tipos de dato: MD:ChannelValue, MD:LQIMeasurementValue, MD:NoiseFloorMeasurementValue, MD:PowerMeasurementValue, MD:RSSIMeasurementValue, MD:TCPEndReasonValue, MD:TCPSeqIndexValue, MD:VLANTagValue, MD:MeasurementDataValue/MD:NodeInformationValue/MD:BindIPValue, MD:MeasurementDataValue/MD:NodeInformationValue/MD:BindIPValue/MD:isListening Multicast

4.2.3 Cambios con la integración de OML

Iperf

- Cambios en "OWLClasses":
  - Añadido hijo para ip de Bind en MD:MeasurementData/MD:NodeInformation/MD:AddressMeasurement/MD:NodeIP: MD:BindIP

4.2.4 Otros cambios

- Cambios en "OWLClasses":
  - Añadido hijo para tráfico IPFIX en MD:Measurement/MD:TrafficMeasurement: MD:IPFIXMeasurement
  - Añadido hijo para almacenar datos de MAC en MD:MeasurementData/MD:SimpleMeasurement: MD:MacMeasurement con un hijo: MD:VLANTag
  - Añadido hijo para almacenar el número de octetos en MD:MeasurementData/MD:SimpleMeasurement: MD:OctetsCount

- Añadido hijo para almacenar datos de TCP en MD:MeasurementData/MD:SimpleMeasurement: MD:TCPMeasurement con hijos: MD:TCPEndReason, MD:TCPFlags, MD:TCPSeqIndex

- Creada clase para identificar los tipos de flag de TCP: MU:TCPFlagTypes

- Cambios en "Properties":

  - Objetos:

    - Creada propiedad MD:hasMetric, inversa con MD:hasMetricAttributes

    - Creada propiedad MD:hasTCPFlags

- Cambios en "Individuals":

  - Creados individuales en MU:TCPFlagTypes: MD:AckFlag, MD:FinFlag, MD:PshFlag, MD:RstFlag, MD:SynFlag, MD:UrgFlag.

- Otros cambios menores

  - MD:MeasurementData/MD:NodeInformation/MD:NodeName/MD:SourceName: Cambiada descripción. "Name or direction of the source node" -> "Name or address of the source node".

  - MGC:TimeStamp: Añadida propiedad de DefaultUnit y isMeasuredIn, para poder indicar que se miden segundos o microsegundos.

4.3 Conclusiones

En este capítulo hemos visto más detenidamente las ampliaciones de la ontología que han sido necesarias para incluir las nuevas herramientas enumeradas en el capítulo anterior. Así la ontología es cada vez más completa y más datos pueden ser consultados a través de ella.

Sobre todo ha sido importante incluir las bases de datos de iLab-t, ya que esto ha permitido acceder a una nueva capa de datos, la capa física inalámbrica.
En el siguiente capítulo veremos la adaptación que ha sido necesaria hacer para poder implementar las correspondencias con procedimientos almacenados, en vez de tablas y columnas.

5 Ampliación del d2r para los procedimientos almacenados

5.1 Introducción

El servidor d2r permite, a partir de las reglas de correspondencia entre bases de datos relacionales y la ontología, hacer consultas sobre estas bases de datos relacionales mediante SPARQL, como si de una ontología se tratara.

Dichas correspondencias suelen basarse en relaciones tabla.columna -> concepto de la ontología, y así se han realizado correspondencias para las herramientas de oml, zabbix, etc.

El problema fue que, para la base de datos TopHat, las consultas sobre los datos del "traceroute" eran demasiado lentas, debido al gran tamaño de esta tabla, así que contactamos con ellos para intentar buscar una solución, y desarrollaron un procedimiento almacenado que optimiza bastante el proceso de búsqueda:

```
get_traceroutes(ARRAY['agent_id', 'destination_id', 'first', 'last', 'hops', 'hop_count', 'tool_id'], 'agent_id = 11824 AND destination_id = 1416 AND hop_count > 20','2012-09-09 14:30:09','2012-09-13 14:30:09') AS foo(agent_id integer, destination_id integer, first timestamp without time zone, last timestamp without time zone, hops ip_hop_t[], hop_count integer, tool_id integer)
```

No obstante, esto nos presentaba un obstáculo. El servidor d2r sólo permite hacer correspondencias de tablas y columnas, no procedimientos almacenados, por lo que es necesario cambiar el código para adaptarlo a nuestras necesidades.

A lo largo del capítulo se irán exponiendo los problemas que han supuesto esta adaptación y cómo se han arreglado para que, finalmente, el servidor admita estos procedimientos almacenados. Primero cómo ha sido arreglado el comprobador de columnas, luego cómo modificar el módulo de las consultas para que gestionara bien los procedimientos almacenados y, por último, cómo listar bien los datos que devuelve el procedimiento almacenado

5.2 Desarrollo de la ampliación

5.2.1 Adaptación del comprobador de columnas

Lo primero que había que hacer era sustituir todas las llamadas en el mapeo que antes eran "traceroute.id" a "procedimiento(...) as foo(...).id", para que el servidor cogiera el procedimiento almacenado.

Esto, aun sabiendo que no iba a funcionar, la primera prueba realizada fue cambiando todas las entradas del fichero de correspondencias de tabla.columna a la nomenclatura del procedimiento.

```
dmichaud@mammitu ~/Documentos/OpenLab/d2rq-0.8.1 $ ./d2r-server ../Mappings/toph
at_mod.n3
11:39:31 WARN  WebAppContext        :: Failed startup of context o.e.j.w.WebAppC
ontext{,file:/home/dmichaud/Documentos/OpenLab/d2rq-0.8.1/webapp/},webapp
de.fuberlin.wiwiss.d2rq.D2RQException: Column @@/*empieza*/get_traceroutes(ARRAY
['agent_id', 'destination_id', 'first', 'last', 'hops', 'hop_count', 'tool_id'],
 'agent_id = 11824 AND destination_id = 1416 AND hop_count > 20','2012-09-09 14:
30:09','2012-09-13 14:30:09') AS foo(agent_id integer, destination_id integer, f
irst timestamp without time zone, last timestamp without time zone, hops ip_hop_
t[], hop_count integer, tool_id integer)/*termina*/.last@@ not found in database
 (E86)
```

Figura 5 Error de reconocimiento de columna

El primero era que el servidor no encontraba esa tabla, cosa que era normal, ya que no existía una tabla con dicho nombre.

El problema residía en que no conseguía validar la columna en dos métodos del módulo que analiza el esquema de la base de datos. Estos métodos se encargan de comprobar los tipos de columna puestos en las reglas de correspondencia, y al no "existir" dichas columnas, no era capaz de encontrarlos.

La solución fue añadir excepciones a estos métodos para el caso en que se estuviera comprobando un procedimiento almacenado, buscando la cadena "foo" (ya que este es el alias que se le da a la salida del procedimiento almacenado, y no hay más columnas ni tablas en esa base de datos con ese nombre). Una vez añadidas estas excepciones, el servidor arrancaba perfectamente

```
dmichaud@mammitu ~/Documentos/OpenLab/d2rq-0.8 (copia).1 $ ./d2r-server ../Mappi
ngs/tophat_mod.n3
12:35:25 INFO  JettyLauncher        :: [[[ Server started at http://localhost:20
21/ ]]]
```

Figura 6 Servidor iniciado correctamente

Se puede observar cómo aparece la tabla "traceroute" en la lista de directorios, en la parte superior de la página.

Figura 7 Vision general D2R

No obstante, si la seleccionamos, salta una excepción porque no encuentra la tabla (cosa que es normal, no tratamos todavía el procedimiento almacenado, así que hace la consulta directamente con toda la llamada a la función):

Figura 8 Error reconociendo procedimiento almacenado

5.2.2 Modificación del módulo de consultas

La razón de este error fue que utilizaba la llamada al procedimiento almacenado al completo sin ningún tipo de control, cosa que despistaba al servidor.

Este error reside en la función que se encarga de hacer las consultas a las bases de datos. Este método recibe la tabla y las columnas a consultar y devuelve el resultado. En un caso de tabla normal, la consulta que haría sería algo del estilo a:

```
SELECT table.id, table.first FROM table WHERE table.id = '123';
```

No obstante, al utilizar los procedimientos almacenados, la consulta que se generaba era así:

```
  SELECT proc(...) as foo(...).id, proc(...) as foo(...).first FROM proc(...) as foo(...) WHERE
proc(...) as foo(...).id = '123';
```

La solución fue tratar la cadena para que la consulta final saliera así

```
  SELECT foo.id, foo.first FROM proc(...) as foo(...) WHERE foo.id = '123';
```

Para esto, hizo falta incluir unos tokens para indicar el inicio y el final del procedimiento almacenado (`/*empieza*/` y `/*termina*/`) . Gracias a estos tokens, se puede reconocer bien el procedimiento almacenado y así poderlo aislar. Una vez aislado, se sustituye, junto a los tokens, por "foo" en todas las ocurrencias, menos en la que está entre el FROM y el WHERE, dejando así la consulta bien formada.

Una vez arreglados estos dos errores, el servidor ya es capaz de listar todos los recursos que devuelve el procedimiento almacenado:

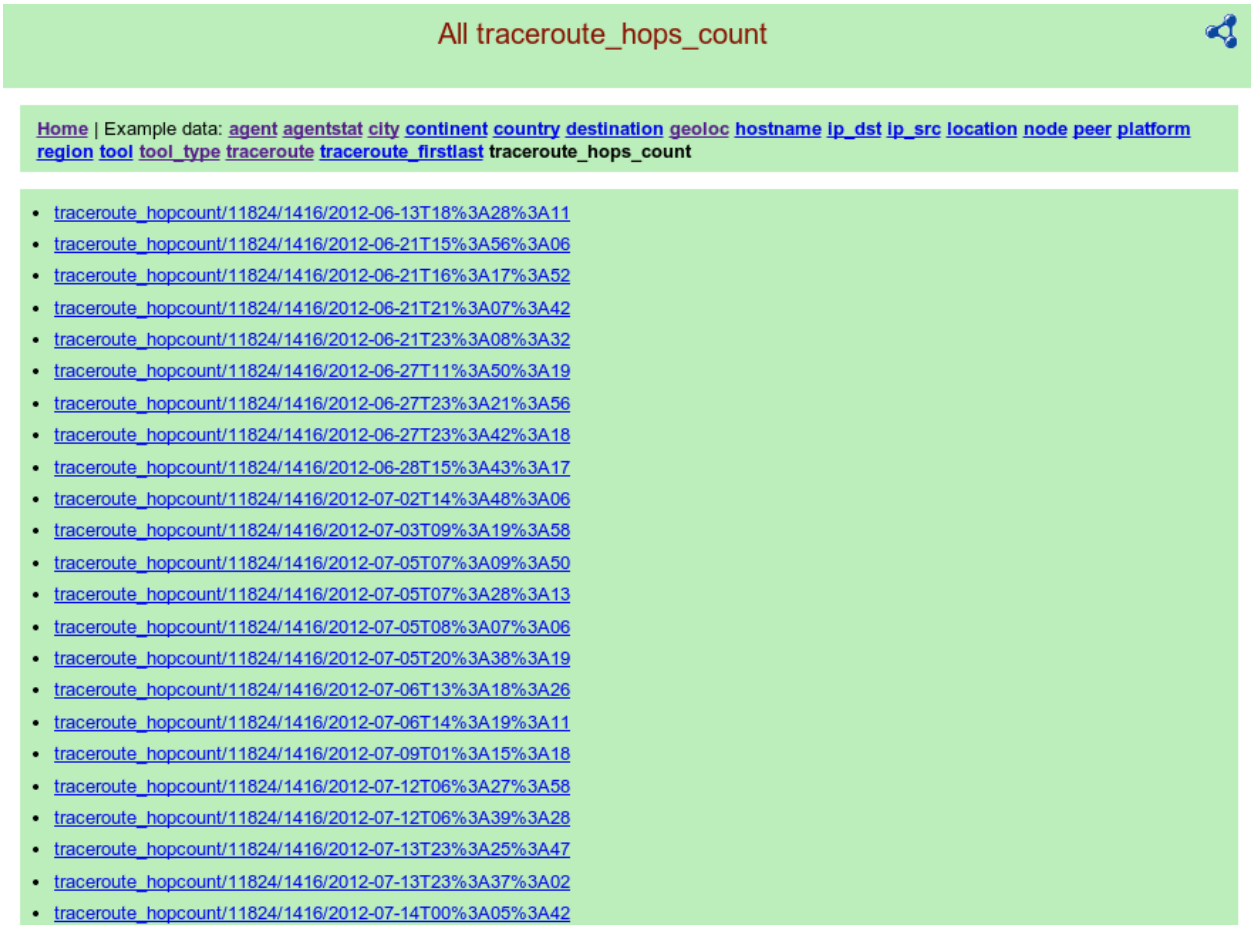

Figura 9 Recursos de traceroute mostrados correctamente

Aunque aún no estaba todo acabado. Al seleccionar una entrada, el servidor todavía nos daba un error:

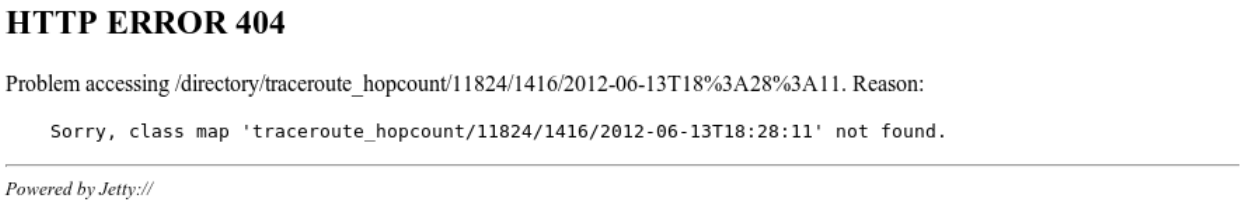

Figura 10 Error accediendo recurso traceroute

5.2.3 Arreglo de la gestión de recursos y directorios

Veamos la causa de esto.

La manera que tiene de organizar los datos el servidor es fijando cada tabla de la base de datos como un directorio, y cada elemento perteneciente a ella como recurso. Con los cambios anteriores, el servidor ya era capaz de listar los elementos dentro de "traceroute" (los que obtenía al consultar con el procedimiento almacenado), pero los categorizaba de directorios, en vez de recursos, y daba error de clase no encontrada porque buscaba un directorio con ese nombre, que no existía porque eso es un recurso.
Para arreglar esto, tras mucho buscar, encontré que en el módulo donde se analizaba el fichero de correspondencia, hay un método que comprueba si la ruta es absoluta (caso del directorio) o hace falta añadirle el prefijo del recurso. Esta comprobación la hacía buscando el carácter ':', y nuestro procedimiento almacenado, en su llamada, como uno de los argumentos es la fecha y hora de inicio de medición, contenía este carácter, por lo que la comprobación daba un falso positivo.
La solución fue ampliar la comprobación para cuando tuviera "foo" (como antes), y una vez añadido esto, el servidor ya clasificaba bien los recursos de traceroute como tal.

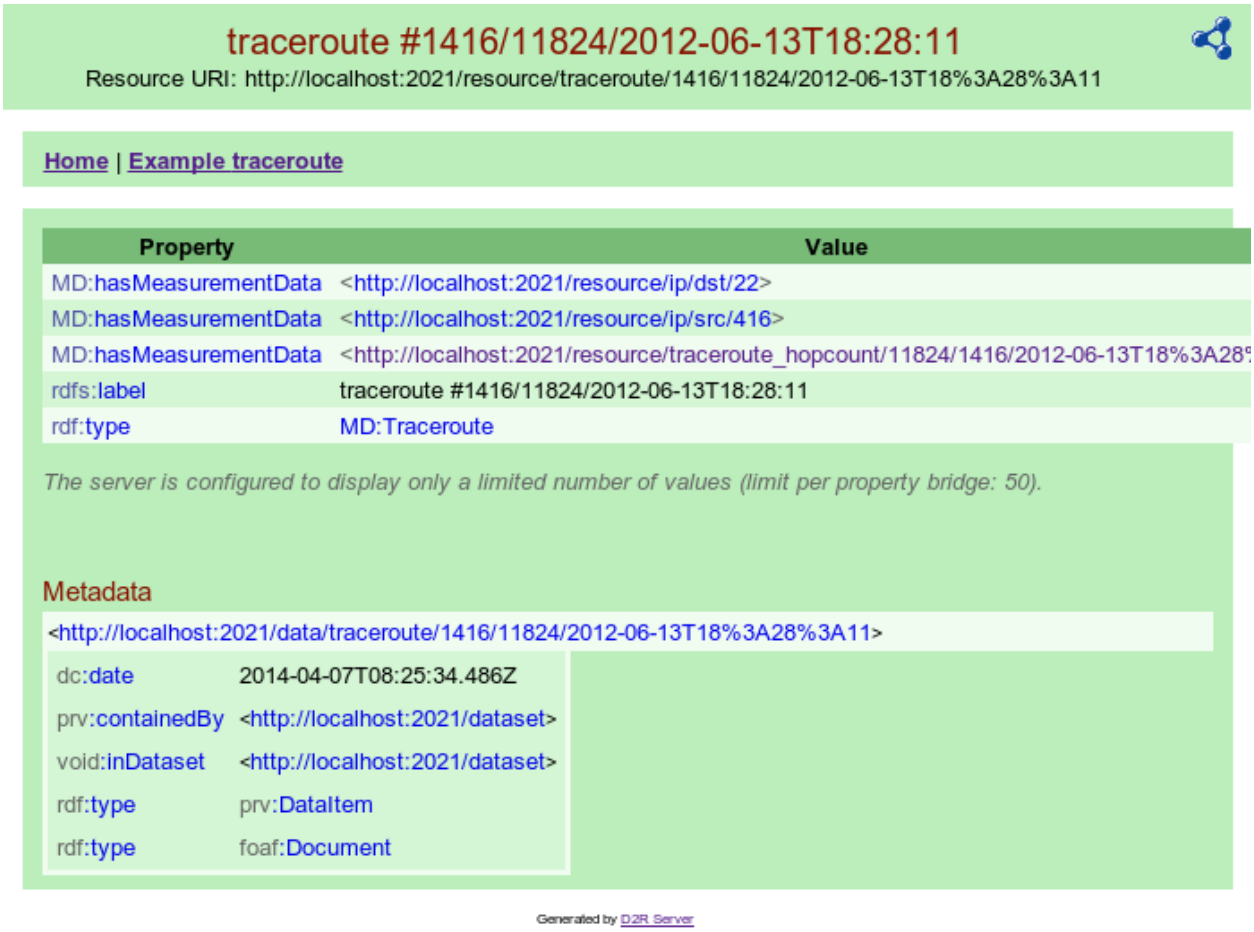

Figura 11 Recurso de traceroute mostrado correctamente

5.3 Conclusiones

Gracias a esta adaptación, nos es posible volver a realizar consultas sobre esta base de datos, aprovechando al máximo las optimizaciones del servidor TopHat.
La manera de implementarlo quizá no es la mejor(con los tokens de inicio y final del procedimiento almacenado o el tener que escribirlo entero a cada vez), pero como sólo va a ser usada en nuestro caso para estas consultas, así ahorramos en tiempo de desarrollo.
Además, el exhaustivo trabajo de "buceo" por el código ha sido ampliamente didáctico, ya que han sido necesarias muchas técnicas distintas para seguir el flujo del programa y poder adaptarlo a nuestra necesidad.

6 Conclusiones generales y trabajo futuro

Finalmente, tras este trabajo, la ontología ha crecido a casi el doble. El número de sistemas y bases de datos integradas ha pasado de 4 a 8 y se ha aumentado la posibilidad de integrar más medidas, desde sólo bases de datos estándar, a poder usar ficheros csv, o hacer correspondencias con procedimientos almacenados.

Este trabajo ha sido didácticamente muy completo y constructivo, ya que ha habido que escribir ficheros con las reglas de correspondencia, que usan su propio lenguaje, y haciendo uso de condiciones SQL para el caso de Zabbix.
Ha habido que aprender a usar las nuevas herramientas cuyos datos se han integrado, que no siempre eran fáciles de instalar y probar, por ejemplo en el caso del Packet Tracking, era necesario instalar sondas en distintos equipos, ir arrancándolas por separado y enviar paquetes de una a otra. Se probó la plataforma Etomic, en el sistema Openlab y al final hubo que instalarlo sobre una red virtualizada en una máquina.
También ha sido necesario hacer mucha investigación en código ajeno, en el caso de la ampliación del servidor d2r para aceptar procedimientos almacenados, haciendo lo imposible por seguir el flujo del programa. Finalmente, esta ampliación está en proceso de ser contribuida al proyecto d2r y hacerla libre al resto de la comunidad de usuarios.

En un futuro, se podría seguir aumentando el contenido de la ontología, aumentando la cantidad de conceptos que esta alberga. Añadir conceptos de nuevas capas, capas físicas u ópticas y así poder abarcar datos de muchas más herramientas, para que el día en que alguien quiera hacer uso de esta herramienta para realizar medidas de distintos programas, encuentre todo lo que pueda necesitar.

7 Referencias

[1] Wikipedia, ***Ontology***, http://en.wikipedia.org/wiki/Ontology_(information_science)
[2] ***OpenLab***, http://www.ict-openlab.eu/
[3] OneLab, ***OpenLab***, http://www.onelab.eu/index.php/projects/openlab.html
[4] ***D2R Server | The D2RQ Platform***, http://d2rq.org/d2r-server
[5] Alfredo Salvador, Jorge E. López de Vergara, Alvaro Katsu, Javier Aracil, ***SPQR: SPARQL pre-Processor and Query Rewriter for a semantic access to multiple heterogeneous network measurements***, Proc. 1st IFIP/IEEE International Workshop on Knowledge Management for Future Services and Networks (KMFSAN 2010), Osaka, Japan, 23 april 2010.
[6] Wikipedia, ***SPARQL***, http://es.wikipedia.org/wiki/SPARQL

7.1 Herramientas:

[7] The ETOMIC Project, ***ETOMIC***, http://www.etomic.org/
[8] ***Network Measurement Virtual Observatory***, http://nm.vo.elte.hu/
[9] Gobierno de España, ***RedIris***, http://www.rediris.es/[10] OneLab, ***TopHat***, https://www.top-hat.info/
[11] Crew project, ***w-iLab.t (iMinds)***, http://www.crew-project.eu/wilabt/
[12] ***OML***, http://mytestbed.net/projects/oml/
[13] Fraunhofer FOCUS, ***Multi-Hop Packet Tracking***, http://www.fokus.fraunhofer.de/en/ngni/projects/archive/archive_2013/Packet_Tracking/index.html
[1] ***Zabbix :: An Enterprise-Class Open Source Distributed Monitoring Solution***, http://www.zabbix.com/es/
[15] European Telecommunication Standards Institute, ***Measurement Ontology for IP traffic (MOI); IP traffic measurement ontologies architecture***, ETSI Group Specification GS MOI 003, V1.1.1, (2013-05), May 2013.