UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Departamento de Telecomunicaciones y Electrónica

TRABAJO DE DIPLOMA

Título: Diseño del modelo de streams de una estación agrometeorológica para agricultura de precisión en la empresa agropecuaria Cubasoy.

Autora: Angelica Maria Pérez Alfaro.

Tutores: Dr. C. Eduardo Izaguirre Castellanos.

Ms. C. Redney Rodríguez Rodríguez.

Consultante: Dr. C. Amed Abel Leiva Mederos.

Santa Clara

UCLV
Universidad Central
"Marta Abreu" de Las Villas

FIE
Facultad de
Ingeniería Eléctrica

Telecommunications and Electronics Department

TRABAJO DE DIPLOMA

Title: Design of the streams model of an agrometeorological station for precision agriculture in the agricultural company Cubasoy.

Author: Angelica Maria Pérez Alfaro.

Thesis Director: Dr. C. Eduardo Izaguirre Castellanos.

Ms. C. Redney Rodríguez Rodríguez.

Consultant: Dr. C. Amed Abel Leiva Mederos.

Santa Clara

PENSAMIENTO

"La educación es lo que queda una vez olvidamos todo lo que aprendimos en la escuela".

Albert Einstein.

DEDICATORIA

A mi familia, en especial a mis padres por su apoyo incondicional.

AGRADECIMIENTOS

A mis padres, por guiarme a través de los años por el camino correcto y haberme enseñado que todo sacrificio tiene resultado satisfactorio. A mi madre, que le debo la vida y todos mis logros, más que un agradecimiento, el compromiso de que allá en donde esté, siempre la haré sentir orgullosa.

A mis amigos Daniela, Amanda, Ana Laura, Isvel, Marco y Onier por tantas vivencias, noches largas y experiencias.

A Carlos por entenderme, quererme, apoyarme y por estar ahí en los momentos buenos y malos.

Al claustro de profesores por contribuir a lo largo de estos cinco años en mi formación profesional.

A mis tutores Redney, Eduardo y Amed.

RESUMEN

Con la proliferación de sensores y tecnologías IoT, los datos en *stream* se han almacenado y analizado incrementalmente, pero raramente combinados, debido a la heterogeneidad de sus fuentes y tecnologías. La semántica se ha utilizado en gran medida para compartir datos de sensores, pero no mucho para anotar los datos de sensores. Los modelos semánticos para la anotación de *streams* son escasos, debido a que generalmente la semántica tarda en procesarse y no es ideal para entornos IoT, donde los datos son frecuentemente actualizados. En este trabajo se presenta un modelo ligero para anotar semánticamente *streams* de datos. El mismo toma ventaja del conocimiento común compartido, pero mantiene las inferencias y consultas simples. Además, se propone una arquitectura del sistema para demostrar la adopción del modelo semántico, la cual está basada en arquitecturas comunes en el campo de IoT como servicios web y *middlewares*. La ontología es aplicable a cualquier área de la agricultura, pieza fundamental de la economía en Cuba, lo cual provee un alcance muy amplio de aplicación en nuestro país. La misma facilita la visualización de datos y la toma de decisiones tanto a máquinas como a seres humanos.

TABLA DE CONTENIDO

INTRODUCCIÓN

El crecimiento de la población mundial está generando gran preocupación por la demanda de alimentos, mientras el calentamiento global continúa afectando las diferentes áreas del sector agrícola. La agricultura tradicional se ve afectada en rendimiento y calidad por diferentes factores ambientales externos. Para enfrentar este problema se vuelve necesaria la innovación tecnológica, permitiendo de esta manera, optimizar recursos invertidos en cultivos para volverlos más productivos y rentables (Nuñez-Agurto et al., 2020). La Agricultura de Precisión (AP) es un sistema de gestión agrícola que se basa en las tecnologías de la información y las comunicaciones. Emplea sistemas sensoriales y técnicas de monitoreo avanzadas para la obtención de datos del suelo, los cultivos, etc. de manera integral, correcta y oportuna para ayudar a directivos, investigadores, especialistas y obreros a la toma de decisiones (Otero Barrera, 2019).

La implementación de dispositivos IoT en la agricultura ayudará a optimizar los recursos invertidos en cultivos, haciéndolos más rentables y productivos(Nuñez-Agurto et al., 2020). La monitorización de diferentes parámetros de interés en un cultivo ha mostrado ser una herramienta útil para mejorar la producción agrícola. Esta monitorización puede ser alcanzada en agricultura de precisión por múltiples tecnologías; sin embargo, el uso de redes de sensores inalámbricos (WSN) resulta en despliegues de menor costo y menor consumo, por tanto se ha convertido en la opción dominante (Garcia-Sanchez, 2011). La utilización de las redes de sensores inalámbricas (WSN) en la medición de los parámetros agrometeorológicos ayudan a resolver problemas críticos sobre el continuo cambio de las condiciones ambientales en la agricultura de precisión y, actualmente, muchas de estas redes están siendo implementadas sobre este campo con el objetivo de obtener una mayor

productividad de los cultivos. Con las especificaciones de las Redes de Sensores Semánticos (SSN) se logra la implementación de ontologías para las estaciones agrometeorológicas, lo que conlleva a una mejora tanto de la comunicación, como de la observación y el muestreo de los sensores, teniendo el sistema la posibilidad de actuar en consecuencia a la variación recibida, lo cual es conocido como SOSA (Sensor, Observación, Muestreo (*Sampling*) y Actuador), una ontología ligera que permite el monitoreo remoto de los cultivos a través de la estación (Valdés Valle Alberto, 2020).

En este escenario los datos son recolectados por varios dispositivos y representados de diferentes formas. La heterogeneidad de los datos y los problemas de operación entre diferentes fuentes y plataformas es un desafío común en los servicios y aplicaciones de análisis de datos en IoT. Para resolver este problema se opta por aplicar semántica ya que ofrece modelos para anotar datos heterogéneos de fuentes heterogéneas (Elsaleh et al., 2020a). Por un lado, el modelo de datos propuesto debe ser ligero para reducir el tráfico y el tiempo de procesamiento. Por otro lado, debe representar explícitamente el significado y las relaciones de términos en los vocabularios (Kolozali et al., 2014). El modelo SSN recientemente ha publicado su enfoque ligero SOSA, pero este modelo está centrado a los dispositivos, no presta mucha atención a los streams de datos de IoT. Para la anotación de *streams*, ha surgido un modelo ligero para proveer anotaciones detalladas en los datos en *stream* conocido como SAO (Ontología de Anotación de *Stream*) ampliamente utilizado en las aplicaciones de ciudades inteligentes; otro modelo ligero para la anotación de estos datos es IoT-Stream que extiende SOSA y por tanto es totalmente compatible con la misma (Elsaleh et al., 2020a).

Una ontología, en Web Semántica, es una especificación formal de un dominio del conocimiento que, en su expresión más simple, se identifica con una taxonomía. Una taxonomía consiste en una jerarquía de conceptos y sus relaciones del tipo clase-subclase. Una ontología formaliza la relación de clase, añade otras relaciones y especifica propiedades para individuos y clases. Esto se implementa mediante el uso de un lenguaje fuertemente basado en lógica simbólica y susceptible, por tanto, de ser eventualmente interpretado por un ordenador. De esto se ocupa el denominado Lenguaje de Ontologías Web (OWL). Con la norma RDF (Marco de Descripción de Recursos) (*Resource Description Framework*) se aportan descripciones (metadatos) de las páginas y sitios web con un formato que sea

compatible con la estructura general de la Web y con diversas categorías de páginas e interoperable entre distintos sistemas informáticos. Las páginas deben tener una codificación en la cual las etiquetas tengan, precisamente, carga semántica, este apartado corresponde al estándar denominado XML (Lenguaje de Etiquetado Extensible o *Extensible Markup Language*) (Codina and Rovira, 2006).

La Web semántica es un conjunto de iniciativas destinadas a promover una futura Web cuyas páginas estén organizadas, estructuradas y codificadas de tal manera que los ordenadores sean capaces de efectuar inferencias y razonar a partir de sus contenidos. Entre los objetivos de la Web semántica se encuentra la posibilidad de que sea posible sostener una interacción entre un usuario y un agente de software mediante el cual el primero pueda ir expresando y perfilando sin ambigüedad varios puntos para que el segundo sea capaz de elaborar una estrategia de búsqueda según su propia iniciativa que involucre el uso de lenguajes documentales, metadatos y ontologías para responder con eficacia y rapidez al usuario.

Situación Problémica:

Los datos de *streaming* son datos generados continuamente en disímiles de orígenes, que normalmente envían los registros de datos simultáneamente, y en tamaños pequeños (del orden de los Kilobytes). El procesamiento de datos de streaming resulta beneficioso en la mayoría de las situaciones en las que se generan datos nuevos y dinámicos de forma continua. Es apto para la mayoría de los sectores y casos de uso de *big data*.

En este contexto, para la transmisión de los datos generados por los sensores de la IoT se emplean tecnologías de *streamData*. Estos de datos deben procesarse de forma secuencial y gradual registro por registro o en ventanas de tiempos graduales, y se utilizan para una amplia variedad de tipos de aplicaciones, entre las que se encuentran las aplicaciones agrícolas. Por ejemplo, los sensores de las maquinarias y vehículos agrícolas envían datos a una aplicación de datos de *streaming*. La aplicación supervisa el rendimiento, y permite la toma de decisiones en función del análisis de los datos en tiempo real.

En nuestro país hay varias empresas que han comenzado a tomar medidas en aras de implementar las tecnologías IoT en el proceso de cultivo. Un ejemplo de esto, es la empresa agropecuaria Cubasoy que radica en la provincia de Ciego de Ávila y se dedica al cultivo de viandas, granos y hortalizas que se encuentra automatizando sus actividades, la misma ya

cuenta con una red de sensores inalámbricos y una estación agrometeorológica que recientemente se le implementó un sensor de datos que permite la toma de decisiones en el sistema de riego de manera no supervisada, mediante el uso de la ontología SSN.

Una de las problemáticas existentes en la estación agrometeorológica es la necesidad de un modelo que permita el monitoreo constante y de manera automática de los datos del clima para la toma de decisiones en lo referente a las cosechas.

Objeto de Investigación: Diseño de un *stream* de datos para la información proveniente de una estación agrometeorológica y su integración en una infraestructura global.

Campo de aplicación: Diseño de *Stream Data* para su aplicación en el contexto de la agricultura de precisión.

Problema Científico:

Las redes de sensores inalámbricos proveen datos que se comportan de forma muy diferente a las formas de tradicionales de bases de datos, estos datos arriban de forma continua, múltiple, con variaciones en el tiempo muy impredecibles y no contienen ninguna información histórica. Las aplicaciones de IoT surgen para colectar, unificar, compartir y publicar datos provenientes de sensores.

De la problemática anterior se desprende la interrogante científica de la investigación:

¿Cómo facilitar la adquisición, el procesamiento y el análisis de los datos en *streaming* provenientes de estaciones agrometeorológicas en Cuba?

Hipótesis de Investigación:

El diseño de un modelo de *streams* de datos provenientes de una estación agrometeorológica utilizando las especificaciones de una ontología para la adquisición y procesamiento de los datos en tiempo real, que facilitará la toma de decisiones en lo referente a los cultivos.

Objetivo general:

- Diseñar un modelo para el *streaming* de datos provenientes de una estación agrometeorológica basada en el modelo SSN con capacidad de integración en una infraestructura web aplicable a IoT.

Objetivos específicos:

1. Analizar los fundamentos teóricos, conceptuales y principales resultados investigativos publicados en relación con el objeto de investigación.
2. Efectuar la caracterización de la Empresa Agropecuaria Cubasoy y del equipamiento tecnológico disponible para llevar a cabo el diseño de la propuesta.
3. Diseñar el modelo de *streams* para la comunicación de la información y datos provenientes de la estación agrometeorológica objeto de estudio.
4. Validar el modelo de streams para la comunicación de la información y datos provenientes de la estación agrometeorológica objeto de estudio, partiendo de que el RDF es un modelo de datos para intercambio.

Tareas Investigativas:

- Realización de búsqueda bibliográfica sobre los fundamentos teóricos conceptuales relacionados con el objeto de investigación.
- Evaluación de las diferentes soluciones relacionadas con el *streaming* de datos en el contexto de aplicaciones de agricultura de precisión.
- Definición de los parámetros y características de los datos a medir por la estación agrometeorológica y de la comunicación en función en función de aplicar la tecnología de streaming de datos en el contexto de agricultura de precisión.
- Realización del modelo de *streams* para la comunicación con la estación agrometeorológica de la empresa agropecuaria Cubasoy.

Resultados esperados:

El diseño de un modelo de *streams* para clasificar la información y datos provenientes de una Estación Agrometeorológica, ubicada en la Empresa Agropecuaria Cubasoy, que ayude al monitoreo y toma de decisiones en tiempo real para optimizar el proceso de producción de los cultivos.

Aplicabilidad:

Los resultados de esta investigación tienen una gran aplicación en el campo de la agricultura de precisión ya que les brindará a sus profesionales la ayuda necesaria para un mejor manejo de los datos en tiempo real.

Viabilidad:

El diseño del modelo de *streams* da solución a problemáticas generadas por la gestión, procesamiento y análisis en tiempo real de gran cantidad de datos que generan los sensores relacionados con los cultivos, evitando la adquisición por nuestro país de software costoso. Además, se optimiza el proceso de producción agropecuaria reduciendo sustancialmente el tiempo de respuesta automática ante cualquier cambio en las variables meteorológicas.

Estructura del Trabajo:

El informe de la investigación se estructurará en introducción, capitulario, conclusiones, referencias bibliográficas y anexos.

En la introducción de quedarán definidos los antecedentes, la importancia, actualidad y la necesidad del tema abordado en el informe.

Capitulario:

CAPÍTULO I: Estará dedicado a describir y caracterizar las ontologías SAO e IoT-Stream con la finalidad de elegir la más adecuada para el diseño y su integración con el sensor semántico.

CAPÍTULO II: Se definirán los requisitos del diseño y se procederá a realizar el diseño del modelo de *streams* de datos en tiempo real para la estación agrometeorológica.

CAPÍTULO III: Se dedicará a la prueba y validación del modelo diseñado evaluando sus resultados ante ejemplos prácticos para verificar la calidad y efectividad del mismo.

Conclusiones

Recomendaciones

Referencias Bibliográficas

Anexos

Antecedentes Internacionales:

Agri-IoT: A semantic framework for Internet of Things-enabled smart farming applications [WWW Document], n.d. URL https://ieeexplore.ieee.org/abstract/document/7845467/ (accessed 5.6.21).

Chaterji, S., DeLay, N., Evans, J., Mosier, N., Engel, B., Buckmaster, D., Chandra, R., 2020. Artificial Intelligence for Digital Agriculture at Scale: Techniques, Policies, and Challenges. ArXiv200109786 Cs.

Communicating agrometeorological information to farming communities, 2000. Agric. For. Meteorol. 103, 185–196. https://doi.org/10.1016/S0168-1923(00)00111-8

Khan, F.A., Abubakar, A., Mahmoud, M., Al-Khasawneh, M.A., Alarood, A.A., 2019. Cotton crop cultivation oriented semantic framework based on IoT smart farming application. Int. J. Eng. Adv. Technol. 8, 480–484.

Lowering Data Dimensionality in Big Data for the Benefit of Precision Agriculture, 2015. Procedia Comput. Sci. 48, 548–554. https://doi.org/10.1016/j.procs.2015.04.134

Navarro, E., Costa, N., Pereira, A., 2020. A Systematic Review of IoT Solutions for Smart Farming. Sensors 20, 4231. https://doi.org/10.3390/s20154231

Rahman, H., Hussain, M.I., 2018. LiO-IoT: A Light-weight Ontology to provide Semantic Interoperability in Internet of Things (SSRN Scholarly Paper No. ID 3361579). Social Science Research Network, Rochester, NY.

Tantalaki, N., Souravlas, S., Roumeliotis, M., 2019. Data-Driven Decision Making in Precision Agriculture: The Rise of Big Data in Agricultural Systems. J. Agric. Food Inf. 20, 344–380. https://doi.org/10.1080/10496505.2019.1638264

Antecedentes Nacionales:

Morales Machado, R., 2016. Sistemas inteligentes de irrigación en la agricultura (Thesis). Universidad Central “Marta Abreu” de Las Villas. Facultad de Ingeniería Eléctrica. Departamento de Automática y Sistemas Computacionales.

Pérez, P., Rubén, J., 2017. Diseño de un sistema automatizado para riego por goteo en la empresa agroindustrial Victoria de Girón (Thesis). Universidad Central “Marta Abreu” de Las Villas. Facultad de Ingeniería Eléctrica. Departamento de Automática y Sistemas Computacionales.

Rodríguez Rodríguez, R., Izaguirre Castellanos, E., Pérez, P., Rubén, J., 2017. Diseño de sistema automatizado para riego por goteo empleando red inalámbrica de sensores de humedad y estación agrometeorológica.

CAPÍTULO 1. Ontologías IoT-Stream y SAO. Su aplicación en la Agricultura de Precisión.

Luego de una profunda revisión bibliográfica sobre el tema abordado, en este capítulo se resumirán los principales aspectos a tener en cuenta en lo referido a Agricultura de Precisión, Web Semántica y procesamiento de datos en *streaming*, su necesidad, ventajas y limitaciones, actualidad y una breve descripción de los elementos más utilizados para el diseño de estos modelos.

1.1. Agricultura de Precisión

La ascendente y constante preocupación por las condiciones medioambientales, así como la necesidad de cosechar productos de manera sostenible y respetuosa con el entorno pone al sector agroalimentario en el punto de mira de la sociedad. Cuando el agricultor debe enfrentar a un sector en el cual existe alta competitividad, con demandas que crecen a diario, precios cada vez más ajustados y exigencia de lograr producir alimentos que se ajusten a las demandas de la sociedad, la aplicación de nuevas tecnologías es una obligación y una necesidad.

Debido a lo antes expuesto hace unos años se produjeron una serie de avances en la automatización vinculada con el área de la agricultura que trajeron consigo la introducción de un nuevo término conocido como Agricultura de Precisión (Callís Flaqué, 2019).

El concepto sobre el que se basa la Agricultura de Precisión es aplicar la cantidad correcta de insumos, en el momento adecuado y en el lugar exacto. Es el uso de la tecnología de la información para adecuar el manejo de suelos y cultivos a la variabilidad presente dentro de un lote. La Agricultura de Precisión (AP) involucra el uso de sistemas de posicionamiento global (GPS) y de otros medios electrónicos para obtener datos del cultivo. Las tecnologías de la Agricultura de Precisión permiten satisfacer una de las exigencias de la agricultura moderna: el manejo óptimo de grandes extensiones. Se presenta como principal ventaja que el análisis de resultados de los ensayos se puede realizar por sectores diferentes dentro de un mismo lote, y de esta manera ajustar el manejo diferencial dentro de los mismos (García and Flego, 2008).

En (Pinto et al., 2007) se define el concepto de Agricultura de Precisión como el conjunto de técnicas orientado a optimizar el uso de insumos agrícolas (semillas, agroquímicos y correctivos) en función de la cuantificación de la variabilidad espacial y temporal de la producción agrícola.

Por otro lado en (Navarro et al., 2020) se define AP como la aplicación de tecnologías suplementarias para las técnicas de producción agrícola para ayudar a minimizar el derroche e impulsar la productividad. Con este propósito, se utilizan recursos tecnológicos que ayudan en varias etapas del proceso de producción, como en la monitorización de las plantaciones, manejo del suelo, irrigación, control de plagas, etc. Estos recursos incluyen, entre otros, temperatura, luminosidad, humedad, presión, concentración química del suelo, videocámaras, sistemas de manejo de información agrícola, sistemas de posicionamiento global (GPS) y redes de comunicación.

La integración de los recursos tecnológicos en el proceso de producción agrícola es un problema relevante. Desde el punto de vista económico, el mercado de la Agricultura de Precisión se espera que tenga un ingreso de 10 billones de USD en 2023, con oportunidades para los proveedores de tecnología, proveedores de equipo y de maquinaria agrícola, productores y otros involucrados en el negocio. En adición, se espera que se optimice la producción de cultivos mejorando la aplicación de nutrientes al suelo, reduciendo la cantidad de pesticidas y el consumo de agua durante la irrigación (Navarro et al., 2020).

Estudios y análisis realizados en el ámbito de la Agricultura de Precisión conllevan a la conclusión de que es un ciclo en el que se destacan cuatro etapas: recolección de datos, procesamiento e interpretación de la información, aplicación de insumos, y evaluación y seguimientos.

La recolección de datos se lleva a cabo mediante el uso de sensores, GPS, satélites, imágenes aéreas, etc. con el objetivo de obtener una detallada caracterización de la zona o cultivo que se desea controlar. Por su parte, en la etapa de procesamiento e interpretación de la información, se recogen los datos y se procesan mediante el uso de Controladores Lógicos Programables, redes neuronales, software estadístico, Sistemas Expertos, etc. con el fin de conocer las condiciones reales en la que se encuentran los suelos y cultivos. Mediante esta información se decide entonces la cantidad de insumos que se necesita aplicar, llegando de

esta forma a la tercera etapa. Finalmente, se analizan mediante los mismos instrumentos que en la primera etapa, si se cumplieron los objetivos buscados, adentrándose en la etapa de evaluación y seguimiento (Callís Flaqué, 2019).

En la agricultura de precisión existen fundamentalmente 3 tipos de análisis, cada uno relacionado con una fase del ciclo de cultivo (Valdés Valle Alberto, 2020).

- Análisis de suelo: Consiste en medir las características del suelo de cultivo, para identificar que cultivos y variedades son adecuados al suelo y que carencias tienen que ser complementadas mediante fertilizantes. Se lleva a cabo en la fase previa a la siembra.
- Análisis de cultivos: Consiste en la monitorización de los cultivos y los factores que afectan y favorecen el óptimo desarrollo de la planta, se monitorizan aspectos tales como crecimiento de la planta, grado de coloración, tiempo atmosférico, detección de plagas. Se aplica en la fase de crecimiento y maduración.
- Análisis de cosechas: Consiste en la monitorización de la cantidad y calidad del producto cosechado en cada zona. Permite medir el resultado de las técnicas aplicadas en fases previas, ayudando a tomar decisiones sobre futuros cultivos y tratamientos. Se aplica en la fase de recolección.

En nuestro país existen algunas limitaciones que dan al traste con el problema de la no explotación de la Agricultura de Precisión (Callís Flaqué, 2019). Entre ellas podemos mencionar:

- Escasa cultura de innovación en el sector agrónomo.
- Grandes problemas de compatibilidad con la maquinaria agrícola.
- Esta tecnología no está al alcance de todos los agricultores, ya que por lo general se encuentra confinada a los de mayor producción.
- Se requiere algunos conocimientos informáticos para implementar los sistemas.
- El costo de los medios tecnológicos (tanto de software como de hardware) necesarios es considerado alto sin antes analizar las ventajas que acarrea.

En cuanto a las ventajas que ofrece la Agricultura de Precisión, primeramente, hay que destacar que la agricultura tradicional trata los campos de cultivo como campos homogéneos,

es decir, sin tener en cuenta la variabilidad espacial y temporal que existe dentro de una misma parcela y, por consiguiente, no se analizan las causas que provocan dicha variabilidad. Esta es una de las causas que trae como consecuencia el derroche insumos, por otra parte, el exceso de productos químicos cuando no son necesarios en toda la parcela, sino que, en un sector específico, lo que perjudica en muchas ocasiones al medio ambiente y a los demás cultivos, lo que provoca pérdidas en los resultados finales de la producción.

Con lo antes expuesto, las ventajas que trae consigo la AP son contundentes, ya que esta permite el empleo de los insumos sólo donde son realmente necesarios, en el momento cuando son requeridos y en la cantidad que se precise. Esto es posible con la integración de las tecnologías de la información y las comunicaciones que permiten el sensado de una gran cantidad de variables en cada sector del campo. Luego la transmisión de esta información para su procesamiento y almacenamiento y posterior a este proceso será usada para el análisis y toma de decisiones. Todas estas ventajas traen consigo un mejor tratamiento de los campos, con menos pérdidas económicas al ahorrar insumos y, por consiguiente, una mayor eficacia de la producción (Otero Barrera, 2019).

El Ministerio de la Agricultura es el organismo de la Administración Pública que implementa un modelo agrícola sustentado en cuatro claves fundamentales: alimentar a todos, proteger el medio ambiente, desarrollar el conocimiento, la ciencia y la tecnología, y mantener la soberanía nacional, de aquí que se pondere su desarrollo con el uso de los avances científicos (Sánchez, 2019). El manejo óptimo de grandes extensiones de tierra y la necesidad de preservación del medio ambiente, hace necesario la evolución de esta actividad hacia tecnologías que logren estos propósitos.

En el escenario agrario cubano aún no hay reconocimiento adecuado a la AP en el siglo XXI dentro de la empresa agraria; no obstante, las bondades que la misma aporta se aplican por el Ministerio de la Agricultura y por el Grupo empresarial dedicado al cultivo y producción del azúcar (AZCUBA) en el desarrollo de cultivos de la caña de azúcar, del arroz, hortalizas, frijoles, y frutales, entre otros.

1.1.1. Riego de precisión

El riego se puede definir como la ciencia de la aplicación artificial de agua a la tierra o el suelo. Se utiliza para ayudar al crecimiento de los cultivos agrícolas, el mantenimiento de

paisajes, y la revegetación de suelos alterados en las zonas secas y durante los períodos de escasez de lluvias. Existen diferentes métodos de riego que se clasifican en riego de superficie o por gravedad, riego por aspersión, riego localizado o microrriego y riego subterráneo (Pozo Pérez, 2017).

El riego por goteo o riego localizado es donde se liberan gotas o un chorro fino, a través de los agujeros de una tubería plástica que se coloca sobre o debajo de la superficie de la tierra. Este permite la utilización óptima de agua y abonos ya que el agua aplicada por este método de riego se infiltra hacia las raíces de las plantas irrigando directamente la zona de influencia de las raíces a través de un sistema de tuberías y emisores (Morales Machado, 2016).

La agricultura en Cuba constituye una esfera económicamente decisiva para la economía. Transformaciones profundas se han iniciado en este sector en las últimas décadas, de manera particular en los sistemas automatizados de riego por goteo, donde se vienen efectuando inversiones por alta eficiencia que los caracteriza. El riego por goteo brinda la posibilidad de efectuar riegos frecuentes reduciendo el peligro de estés hídrico, mantiene la humedad del suelo a niveles óptimos y mejora las condiciones para el desarrollo de plantas. Por consiguiente, su automatización juega un rol fundamental, lo cual adquiere una mayor relevancia si funcionalmente se integra al mismo, información relacionada con la humedad del terreno y las variables climáticas (Rodríguez Rodríguez et al., 2017).

En (Pozo Pérez, 2017) se realiza una propuesta de diseño de la automatización de un sistema de riego por goteo en la Empresa Agroindustrial “Victoria de Girón”, en Jagüey, Matanzas para introducir la automática en la agricultura del país. En el mismo, además del control del sistema de bombeo se evalúa el empleo de sistemas sensoriales con la idea de lograr resultados que tributen a un ahorro de recursos hídricos y energéticos, así como al aumento de la productividad del cultivo de cítricos.

En (Valdés Valle Alberto, 2020) se realiza un paso adelante en la automatización del país, mediante el desarrollo de un sensor de datos para la gestión inteligente del riego en las estaciones agrometeorológicas automatizadas en Cuba utilizando SSN.

El riego localizado brinda la posibilidad de una automatización total, permite la aplicación de abonos en el agua de riego, adaptándose a las necesidades del cultivo, posibilita el control total sobre el suministro hídrico de las plantas, lo que permite provocar estrés o garantizar

una humedad óptima en los momentos del ciclo del cultivo que se desee. Además, permite un gran ahorro de agua respecto a otros tipos de riego, debido a que se elimina la pérdida de agua durante su transporte, al llegar mediante las tuberías hasta la propia planta y se reduce la evaporación directa del suelo al mojarse solo una parte del terreno. Otra de las ventajas es que aumenta la cantidad y la calidad de las cosechas, presenta una mayor uniformidad de riego, menor infestación por malas hierbas debido a la menor superficie del suelo humedecida y facilidad de ejecución de las labores agrícolas, al permanecer seca una buena parte de la superficie del suelo (Pozo Pérez, 2017).

1.2. Estaciones Agrometeorológicas

La Agrometeorología tiene como objetivo analizar y definir los sucesos meteorológicos y aplicar conocimientos del clima a usos prácticos en la agricultura, esta disciplina proporciona al productor agropecuario información relevante para mejorar la producción agropecuaria, tanto en cantidad como en calidad.

Una estación meteorológica es un equipo de monitoreo en la cual se podrá medir y registrar regularmente diversas variables meteorológicas. Estos datos se utilizan para la elaboración de predicciones meteorológicas a partir de modelos numéricos, así como para estudios climáticos regularmente enfocados a la agronomía (Gutiérrez Junco et al., 2015).

Una estación meteorológica automatizada es una estructura o dispositivo dotado con sensores que responden a estímulos electrónicos, que tienen la capacidad de registrar y colectar información meteorológica en forma automática y en tiempo real (Pozo Pérez, 2017). Debe estar dotada de una serie de sensores capaces de medir diversas variables, necesarias para lograr controlar el momento de irrigación, estas deben ser: temperatura del aire, precipitación pluvial, humedad ambiental, radiación solar, velocidad del viento, dirección del viento y humedad de las hojas (Rodríguez Rodríguez et al., 2017). Disponen de sistemas de energía autónoma, bien con paneles solares o con baterías de larga duración.

Una red meteorológica es un conjunto de estaciones meteorológicas automatizadas distribuidas estratégicamente en una región o zona, las cuales registran en forma continua las condiciones del tiempo atmosférico de una región y envían los datos a una base central para ser almacenados, procesados, interpretados y distribuidos de manera oportuna a los usuarios, incluso en tiempo real o cercano al real (Pozo Pérez, 2017).

Con la información que aportan las estaciones agrometeorológicas, es posible calcular parámetros muy importantes para la agricultura como: la probabilidad de ocurrencia de lluvias; ocurrencia de temperaturas extremas en duración y localización; estimación de tasas de fotosíntesis, posible ocurrencia de enfermedades y la evotranspiración (Rodríguez Rodríguez et al., 2017).

1.2.1. Componentes de una estación meteorológica

Para conseguir su función, una estación meteorológica recoge datos y características del clima mediante variados sensores o instrumentos. Según el tipo de estación meteorológica, variará el tipo y la cantidad de instrumentos a utilizar.

Entre los componentes más comunes de una estación agrometeorológica automatizada podemos encontrar:

Tanque Evaporímetro: Mide la evaporación ambiental mediante un preciso sensor ultrasónico, se monta en un pocillo tranquilizador para medir la altura de la capa superficial del agua en el tanque, obteniendo la cantidad de agua evaporada entre diferentes lecturas, pudiendo detectar variaciones de altura de décimas de milímetro. Dispone de un sistema de rellenado automático del tanque, controlado por la Estación Meteorológica Digital, permitiendo realizar las lecturas siempre en el mismo rango de alturas.

Anemómetro: Nos permite conocer la velocidad del viento incidente. El tipo de anemómetro más usado es el llamado “de molinete o rotación”. El viento golpea el molinete y este comienza a girar, un contador registra las vueltas que da en un tiempo determinado y marca la velocidad.

Veleta: Mide la dirección del viento incidente. Se compone de dos partes, una con forma de flecha y que va girando hacia el viento y la otra parte es algo más ancha para que atrape la brisa.

Radiómetro: Nos permite conocer la radiación solar en un rango determinado.

Pluviómetro: Permite la medición de la precipitación de lluvia. Los pluviómetros modernos nos permiten conocer la cantidad de lluvia acumulada, minutos en los que ha estado lloviendo, intensidad de lluvia instantánea, intensidad de lluvia máxima y número de basculaciones.

Termómetro: Es el instrumento más conocido y utilizado, y nos permite saber qué temperatura hay en un lugar determinado.

Higrómetro: Recoge los datos de humedad relativa que existe en el aire.

Barómetro: Mide la presión atmosférica, es decir, el peso que ejerce el aire sobre la corteza terrestre. Es necesario conocer de antemano la altura de la instalación para suministrar el rango correcto de lectura.

Geotermómetro: Permite la medición de la temperatura del suelo, lo cual es de vital importancia para la agricultura.

Estos dispositivos permiten contar con información meteorológica automatizada en tiempo real que ayuda al productor a tomar decisiones oportunas en el manejo de cultivos para librar de mejor manera los riesgos climáticos que representan las olas de calor, vientos fuertes, sequías e inundaciones; también para obtener los máximos beneficios de condiciones climáticas favorables para la obtención de altas cosechas (Pozo Pérez, 2017).

1.3. Internet de las Cosas

IoT (*Internet of Things*) puede ser entendido como una red de dispositivos inteligentes interconectados capaces de comunicarse los unos con los otros, generando datos relevantes del ambiente en el cual se encuentran operando. Por tanto, cualquier dispositivo capaz de establecer conexión con el Internet puede ser considerado una “cosa” en el contexto del IoT, pudiendo ser aplicados en electrodomésticos, electrónica, muebles, maquinaria agrícola o industrial y hasta en personas (Navarro et al., 2020). Los dispositivos más utilizados por IoT son los microcontroladores y sensores que permiten la interconexión y envío de datos, utilizando distintos tipos de protocolos, también posibilitan recibir y ejecutar comandos, mediante aplicaciones específicas instaladas en un computador, o alojadas en la nube, e incluso existen aplicaciones desarrolladas para dispositivos móviles (Nuñez-Agurto et al., 2020). En general, un ambiente IoT consiste componentes de hardware físicos, una plataforma IoT que conecta el hardware al mundo digital y aplicaciones IoT que interactúa con los componentes de hardware físico a través de la plataforma IoT (Silva et al., 2020).

1.3.1. Tecnologías IoT

El concepto de IoT se derivó en primer lugar por el *Automatic Identification (Auto-ID) Labs* en el Instituto de Tecnología de Massachusetts (MIT) en 1999. El Auto-ID Labs propuso los sistemas identificación por radiofrecuencia (RFID) que conectan los dispositivos y transmiten información vía radiofrecuencia al Internet con el objetivo de alcanzar un manejo e identificación inteligente. Para formalizar el concepto de "*Internet of Things*", la Unión Internacional de Telecomunicaciones (ITU) lanzó el reporte de "*ITU Internet reports 2005: the Internet of Things*" en la *World Summit on Information Society* (WSIS) llevada a cabo en Tunis en 2005, donde fueron introducidas las características de IoT, los desafíos técnicos relacionados y las futuras oportunidades del mercado (Wang et al., 2015).

A pesar de que la idea de IoT no es nueva, su adopción ha incrementado en años recientes (Navarro et al., 2020). El desarrollo de las tecnologías de sensores nano-embebidos, redes inalámbricas, computación en la nube y análisis de *Big Data* revolucionaron la integración del IoT permitiendo su aplicación en varios dominios (Khan et al., 2019). Todos estos componentes tecnológicos ayudaron a construir una red de dispositivos capaz de compartir datos e información (Navarro et al., 2020).

A medida que sigue aumentando la población del planeta, se torna cada vez más importante que las personas se conviertan en guardianes de la Tierra y sus recursos. Además, las personas desean vidas saludables, plenas y confortables para sí mismas, sus familias y las personas que quieren. Si se combina la capacidad del IoT para percibir, recolectar, transmitir, analizar y distribuir datos a escala masiva con la manera en que las personas la información, la humanidad tendrá el conocimiento y la sabiduría necesarios no solo para sobrevivir, sino para mejorar y prosperar en los próximos meses, años y siglos (Evans, 2011). Sin embargo, lidiar con grandes cantidades de datos generados por diversos dispositivos y sensores embebidos requiere un diseño cuidadoso y la aplicación de las mejores prácticas (Tran, 2020).

1.4. Big Data

El término "*Big Data*" se refiere a un grupo de tecnologías y métodos orientados al análisis y procesamiento de grandes cantidades de datos (estructurados y desestructurados) que no pueden ser procesados por los métodos tradicionales; estos sirven para obtener cualitativamente nuevo conocimiento. Su utilización está determinada por el crecimiento

exponencial de la cantidad de datos generados de forma electrónica y almacenados en bancos de datos para su uso futuro (Rogushina et al., 2018).

Las herramientas de *Big Data* utilizan tecnologías multinúcleo para ofrecer mayor capacidad de procesamiento a través de altas prestaciones, en base de datos y de análisis en memoria que ofrecen un mayor conocimiento más rápidamente de grandes volúmenes de datos y flujo de datos. Todo ello independientemente de los formatos y las fuentes de los orígenes de datos. Con las herramientas de *Big Data* se pueden procesar información online proveniente de múltiples orígenes como pueden ser las redes sociales o grandes bases de datos no estructuradas. También se pueden tratar los datos de múltiples fuentes y formatos, ya sean texto, datos, imágenes o mezcla de todo ello. Actualmente es posible implementar herramientas de *Big Data* en la forma que mejor se adapte a las necesidades de los usuarios.

Las técnicas de *Big Data* persiguen complementar el manejo de grandes volúmenes de datos con las técnicas de análisis de la información más avanzadas y efectivas para extraer de modo óptimo el conocimiento contenido en los datos (MARQUÉS, 2015).

Desde hace algunos años el mundo científico comenzó a usar grandes bases de datos para analizar patrones biológicos, lo que se conoce como *Data Science*. La ciencia de datos unifica la estadística, la matemática y el análisis de datos con el fin de comprender y analizar fenómenos reales usando métodos, procesos y algoritmos para extraer conocimiento e información. Esta ciencia se encarga de utilizar bases de datos, inventarios o repositorios climáticos online, resultados de otros investigadores u organizaciones y nuevas herramientas tecnológicas como imágenes satelitales o inteligencia artificial para contestar nuevas preguntas, aprovechando la gran cantidad de información recolectada por años. Esta nueva ciencia tiene como ventaja que un mismo grupo de datos se puede analizar de maneras distintas pudiendo ayudar a resolver diferentes preguntas (Berryman, 2020).

1.4.1. Big Data en la Agricultura

La agricultura está enfrentando una revolución con la integración de herramientas y sistemas de decisiones potenciados por *Big Data*. Los datos se están utilizando en el mundo agrícola para aumentar la eficiencia y al mismo tiempo disminuir el impacto sobre el medio ambiente. La capacidad computacional moderna ha permitido aumentar la capacidad de recolectar, intercambiar, procesar y sintetizar datos de una forma tal que está impactando en todo el

ámbito agrícola: maquinaria, optimización de semillas, fertilizantes e insumos, riego y gestión predial. Para poder obtener valor de la *Big Data*, esta debe ser procesada y analizada a tiempo y sus resultados deben estar disponibles para tomar decisiones en las operaciones agrícolas. La efectividad en su uso también está relacionada con tener una combinación acertada de gente, procesos y tecnologías.

La producción agrícola es muy compleja ya que interactúan la biología, el clima y las acciones humanas. Los productores han adoptado tecnologías de precisión en los años recientes. Con la disponibilidad de GPS y de otras tecnologías, los productores pueden hacer seguimiento a los rendimientos, guiar y controlar las máquinas, monitorear las condiciones del campo y gestionar los insumos a niveles muy precisos dentro de los campos, aumentando de esta manera la productividad y la rentabilidad.

Al mismo tiempo, los datos se van acumulando en conjuntos tan grandes y complejos que no pueden ser estudiados sin software. Los datos en sí, no pueden generar ideas ni visiones; por lo tanto, se requieren servicios de consultoría y análisis para ayudar a que los productores saquen provecho de las enormes cantidades de datos que recolectan. Las aplicaciones de software basadas en sistemas de máquinas aprenden a través de sus interacciones con los datos, aparatos y personas. Y a medida que van aprendiendo, generan grandes oportunidades para tomar mejores decisiones dentro de los campos. Las empresas de *Big Data* pueden evaluar muchísimas variedades genéticas, agroinsumos y condiciones de los campos, suelos y climas. Pueden realizar ensayos de campos en tiempo real en millones de hectáreas. Esto entrega a los agricultores información para optimizar las siembras, a nivel de cada planta en suelos y climas específicos (Agtech, 2017).

1.5. Web Semántica

Según el W3C (*World Wide Web Consortium*), la Web Semántica proporciona un marco común que permite que los datos sean compartidos y reutilizados a través de aplicaciones, empresas y fronteras comunitarias. Es un esfuerzo colaborativo liderado por el W3C con la participación de un gran número de investigadores y socios industriales. Está basado en *Resource Description Framework* (RDF) e integra una variedad de aplicaciones utilizando XML para la sintaxis y URI para las denominaciones (Codina and Rovira, 2006).

La Web Semántica tiene diferentes definiciones según la visión utilizada. Desde el punto de vista de la Inteligencia Artificial se puede definir como un conjunto iniciativas destinadas a promover una futura Web cuyas páginas estén organizadas, estructuradas y codificadas de tal manera que los ordenadores sean capaces de efectuar inferencias y razonar partir de sus contenidos. Según la visión del procesamiento robusto se define a la Web Semántica como un conjunto de iniciativas destinadas a convertir la *World Wide Web* en una gran base de datos capaz de soportar un procesamiento sistemático y consistente de la información (Codina and Rovira, 2006).

La comunidad de la Web Semántica juega un rol relevante cuando se requiere interoperabilidad de los datos e integración de los resultados. El análisis semántico es una iniciativa emergente que habla del razonamiento de *Linked Open Data* (LOD), provee una visión de cómo deducir información significativa de los datos de IoT, apuntando a compartir la forma de interpretar los datos en una forma interoperable para producir nuevo conocimiento. El análisis semántico unifica diferentes tecnologías semánticas y herramientas de análisis como el razonamiento basado en lógica, machine learning, LOD, el objetivo principal es convertir datos en conocimientos aplicables (Soldatos, 2016).

1.6. Ontologías. Definiciones

El término ontología ha gozado de varias vidas. Originalmente, el término provenía de la Filosofía clásica. En ese contexto, la ontología era una parte de la metafísica que se ocupaba de estudiar la naturaleza de la existencia. Era una especialidad que gozó del favor de los filósofos casi desde el nacimiento mismo de la Filosofía en la Grecia clásica. La llegada de nuevas corrientes filosóficas a finales del siglo XIX de corte anti metafísico hizo que el término perdiera gran parte de su vigencia fuera de algunas escuelas de pensamiento minoritarias o muy especializadas (Codina and Rovira, 2006).

Sin embargo, mucho después, tanto el colectivo de los informáticos como el los lingüistas rescataron el término para darle significados distintos e incorporarlo al lenguaje de la ciencia.

En particular, los estudios de Inteligencia Artificial recuperaron el término para designar esquemas conceptuales formalizados sobre algún aspecto de la realidad, con la finalidad de facilitar su reutilización en diferentes contextos o la comunicación entre diferentes sistemas,

casi siempre con el telón de fondo de la construcción de sistemas expertos (Codina and Rovira, 2006).

En los últimos años, el proyecto de la Web semántica ha servido para asentar lo que podríamos denominar el uso “moderno” del término ontología. En este nuevo contexto, una de las definiciones más citadas es la debida a Gruber (1993) según la cual una ontología es “la especificación de una conceptualización”.

Con posterioridad, (Guarino, 1998) definió ontología como un artefacto de ingeniería, constituido por un vocabulario específico utilizado para describir una cierta realidad, más un conjunto de suposiciones explícitas con respecto al significado pretendido de las palabras del vocabulario. Este conjunto de suposiciones tiene usualmente la forma de una teoría lógica de primer orden, donde las palabras del vocabulario aparecen como nombres de predicados unarios o binarios, respectivamente llamados conceptos y relaciones. En el caso más simple, una ontología describe una jerarquía de conceptos relacionados por subsunción; en los casos más sofisticados, los axiomas apropiados son agregados en orden de expresar otras relaciones entre conceptos y restringir su interpretación prevista.

En (Smith et al., 2006) la denominaron un artefacto de representación que comprende una taxonomía como parte propia, cuyas representaciones tienen la intención de designar alguna combinación de clases tanto universales como definidas y ciertas relaciones entre ellas. Finalmente para (Stuart, 2016) una ontología es una representación formal del conocimiento con ricas relaciones semánticas entre términos.

En (Rogushina et al., 2018), una ontología se define como una descripción detallada de alguna área problemática, que es utilizada para la definición formal y declarativa de esta conceptualización. Frecuentemente una ontología es llamada al base de conocimientos de un tipo especial, que puede ser dividido, alienado y utilizado independientemente en el marco de trabajo del dominio considerado.

Una ontología es un tipo específico de taxonomía, por lo tanto, la formalización de los datos mediante ontologías constituye un soporte para la organización, recuperación y navegación. Puede agregarse que constituye una tecnología soporte de la creación de conocimiento. Las ontologías se representan mediante grafos constituidos por conceptos y las relaciones existentes entre ellos; pero el grafo que se obtiene como respuesta a una consulta no se

construye solo con las relaciones explícitas, puede construirse, además, mediante relaciones inferidas (Barber et al., 2018).

Actualmente, el uso de ontologías como el medio adecuado para la descripción de diferentes dominios es un hecho generalmente aceptado y un amplio rango de ontologías están disponibles a través de la Web, confirmando la popularidad de este enfoque entre varios grupos de desarrolladores y usuarios de las aplicaciones Web (Rogushina et al., 2018), incluyendo aplicaciones de IoT y *Big Data*.

1.6.1. Ontologías relacionadas al IoT

En los años recientes, las redes de sensores inalámbricos han sido desplegadas en varios dominios (ciencias médicas para el cuidado de pacientes utilizando sensores biométricos, detección de fuegos, meteorología para la predicción del tiempo, imágenes de satélite para la observación de la Tierra y el espacio, agricultura, etc.). Los sensores están distribuidos alrededor del globo, capturando y produciendo continuamente una enorme cantidad de streams de datos (Llanes et al., 2016). Sin embargo, la carencia de una integración y comunicación entre estas redes y la falta de información contextual y conocimiento de fondo, a menudo aísla streams de datos importantes e intensifica los problemas existentes de demasiados datos y conocimiento insuficiente sobre el significado implícito de estos datos y las intenciones del usuario (Le Phuoc and Hauswirth, 2019).

Para lograr semántica en los datos (datos enriquecidos), varias ontologías y descripciones de datos comunes son ampliamente utilizados (Bermudez-Edo et al., 2017). Una ontología provee una descripción comprensiva de los datos que incluye definiciones, categorizaciones, así como las restricciones que se le imponen y las reglas lógicas que debe cumplir (Ganzha et al., 2017).

Los avances recientes en esta área son discutidos en varios trabajos existentes incluyendo la ontología del *Semantic Sensor Network Incubator Group del W3C* (ontología SSN). La investigación sobre los datos de IoT hasta ahora se ha enfocado en la representación del conocimiento, en la publicación y anotación de datos y en los modelos de datos enlazados. Sin embargo, modelar e integrar los datos de observación y medición, transmitir los datos de sensores en tiempo real y proporcionar mecanismos de descubrimiento para habilitar los mecanismos de consulta distribuidos son otros problemas claves para permitir soluciones de

extremo a extremo para la publicación y consumo de datos de sensores provenientes de recursos de IoT (Barnaghi et al., n.d.).

1.6.2. Lenguajes de Ontologías

El Lenguaje de Ontologías Web (OWL) es el lenguaje estándar de la web semántica para expresar y codificar ontologías. Está concebido para ser utilizado cuando la información contenida en los documentos necesita ser procesada por aplicaciones informáticas, en oposición a las situaciones donde el contenido solamente debe ser presentado a seres humanos. OWL puede ser utilizado para representar explícitamente el significado de términos en vocabularios y las relaciones entre estos términos (Codina and Rovira, 2006).

El modelo RDF permite representar información por medio de grafos dirigidos en los cuales los vértices tienen un sentido definido que constituyen triples. La estructura del triple RDF permite, sin limitaciones, que se puedan enunciar afirmaciones sobre cualquier recurso (Barber et al., 2018). OWL utiliza RDF/XML para representar y codificar las ontologías, por lo que es una extensión de RDF que añade elementos para describir características y clases.

1.6.3. Ontología para estaciones agrometeorológicas sobre plataforma IoT

Esta investigación se centra en la construcción para el entorno cubano del proceso de ingestión de datos de un sistema de IoT para su aplicación en una estación agrometeorológica, mediante el diseño de una ontología para gestionar y procesar los streams que guardan la información de los sensores físicos.

El desarrollo de este sistema se centra en la plataforma de IoT Sofia2 (figura 1), un sistema de código abierto, distribuido (Cebrián and Manuel, 2017) y capaz de interactuar con los sistemas que existen el país.

Utiliza ontologías y visión semántica para garantizar independencia de protocolos, permitiendo la representación del mundo físico en el mundo digital. Es agnóstica de las comunicaciones, con implementaciones en múltiples protocolos de comunicación ligeros (REST, OPC, MODBUS, WebSockets, MQTT, WS, JMS, AMQP…). Permite el procesamiento en tiempo real de la información intercambiada, la publicación de datos independientemente del repositorio (tiempo real o histórico) y publicación en portales *Open Data* (Cebrián and Manuel, 2017).

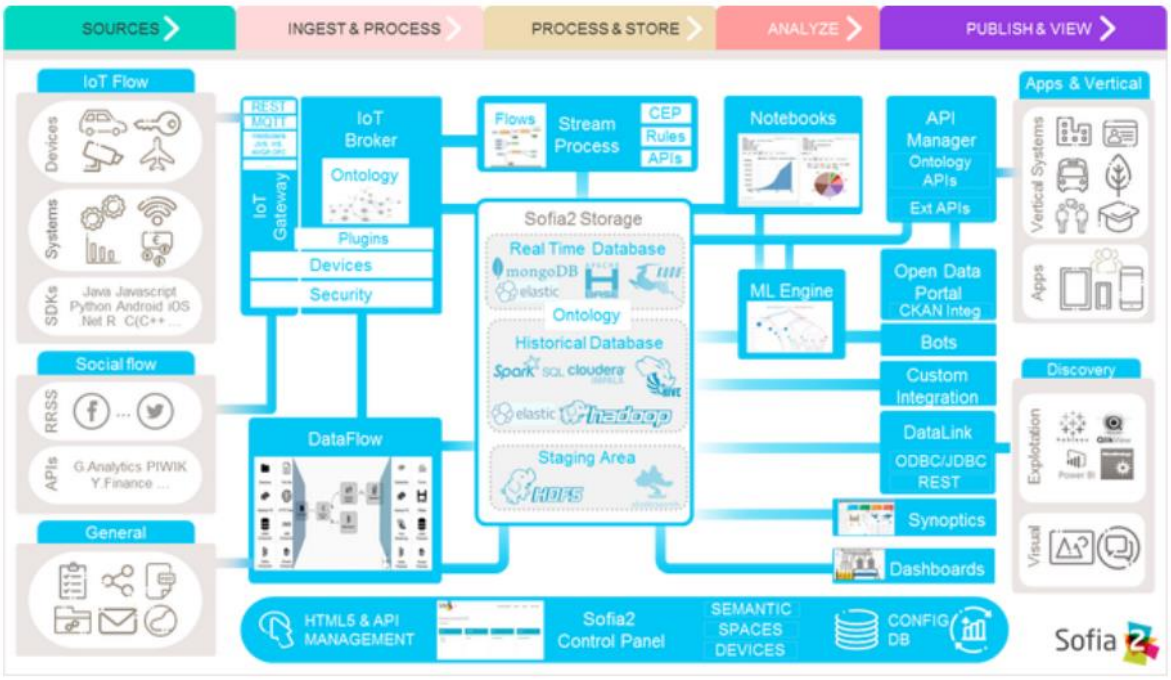


Figura 1.1. Componentes de la Plataforma Sofia2.

El enfoque principal de este trabajo es el proceso de ingestión, procesamiento y almacenamiento de la información, teniendo como núcleo el *Semantic Broker* que es el módulo de la plataforma que recibe, procesa y almacena toda la información de las aplicaciones, sensores y dispositivos conectados. Esta capa validará la corrección sintáctica y semántica del dato recibido gracias a la definición previa de la estructura de dato esperado, identificando de qué dato trata y aplicando la seguridad correspondiente al mismo.

En la plataforma se definen 4 conceptos elementales (figura 2):

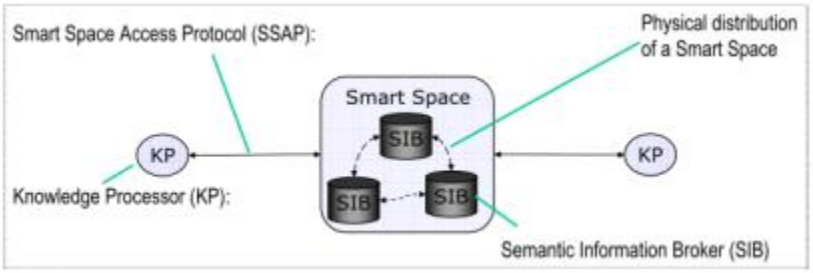


Figura 1.2. Conceptos elementales de Sofia2.

*Smart Space***:** Es el entorno virtual donde diferentes aplicaciones interoperan para ofrecer una funcionalidad compleja, su núcleo es el SIB (*Semantic Information Broker*). Los *Smart Spaces* pueden comunicarse entre ellos estableciendo relaciones de confianza.

SIB: Es el núcleo de la plataforma. Recibe, procesa y almacena toda la información de las aplicaciones conectadas a Sofia2, actuando de Bus de Interoperabilidad. En él se reflejan los conceptos existentes en el dominio (reflejados en las ontologías) y su estado actual (instancias particulares de ontologías).

KP (*Knowledge Processor*): Cada uno de los sistemas y/o aplicaciones que interoperan en el *Smart Space* a través del SIB deben estar definidos y configurados como KPs en el mismo. El KP es un elemento desplegado en el *Smart Space* que puede consumir y/o producir información. Cada aplicación trabaja con instancias de los conceptos relevantes del dominio para la que están diseñada.

SSAP (*Smart Space Access Protocol*): Es el lenguaje de mensajería estándar para la comunicación entre los SIBs y KPs. El lenguaje es independiente de la red subyacente.

Las ontologías son descripciones semánticas de un conjunto de clases, con ellas se modelan los diferentes sistemas de información que interoperan en el dominio de *Smart Space*. En Sofia2, estas ontologías están representadas en formato *JSON-Schema,* que las definen y validan.

Con el objetivo de garantizar que, para cada momento del ciclo de vida de la información, ésta se gestiona de la mejor manera, la plataforma plantea el uso de tres repositorios distintos (figura 3) que se complementan y comunican componiendo una solución de almacenamiento completa.

Este módulo ofrece optimización de tiempos de acceso a la información, soporte a diferentes tecnologías en función del patrón de accesos, altas y consultas de cada repositorio, soportan estándares y bases de datos SQL y NO-SQL. Los repositorios están integrados entre sí y con las demás capas de la plataforma.

Figura 1.3. Almacenamiento en Sofia2.

1.7. Data Streams

Los sensores, actuadores y dispositivos relacionados al IoT, se mantienen generando continuamente *data streams* relacionados con el estado actual del entorno en el que han sido desplegados. La enorme cantidad de datos generados tienen el potencial de proporcionar una perspectiva más significativa para la comprensión de fenómenos ambientales complejos si se analizan adecuadamente en tiempo real. Por tanto, la aplicación de tecnologías como el procesamiento de streams, en el campo de la monitorización de las variables ambientales puede ser beneficioso para predecir fenómenos ambientales complejos que influyen en el proceso de toma de decisiones efectivas en la agricultura.

El procesamiento de datos de *streaming* necesita dos capas: una de almacenamiento y una capa de procesamiento. La capa de almacenamiento debe ser compatible con la ordenación de registros y la alta coherencia para facilitar lecturas y escrituras de transmisiones de datos de gran tamaño de forma rápida, económica y que permita reproducciones adicionales. La capa de procesamiento es responsable del consumo de los datos de la capa de almacenamiento. Realiza tareas informáticas a partir de los datos y notifica a la capa de almacenamiento para que elimine los datos que ya no son necesarios. También es necesario planificar la escalabilidad, la durabilidad de los datos y la tolerancia a fallos tanto en la capa de almacenamiento como en la de procesamiento.

Los datos provenientes de sensores representan una medición u observación del mundo físico que requieren otros atributos descriptivos para que los datos sean más significativos. Utilizando descripciones semánticas se puede proporcionar descripciones para los datos en *stream* de los sensores que son interoperables e interpretados por máquinas. Los *streams* de sensores semánticos incluirán datos sensoriales sin procesar anotados con descripciones semánticas que especifican información espacial, temporal y otros atributos de los datos.

El modelo de *streaming* para los datos de los sensores debe considerar la cantidad, variedad y velocidad de cambio de los mismos, mientras describe los valores de observación y medición. Otro aspecto que debe ser tomado en consideración es cómo los datos serán usados y consultados.

1.8. Vocabularios ontológicos asociados a la anotación de streams de datos

1.8.1. Orígenes de SAO e IoT-Stream

Hace unos años, la principal contienda de las ontologías era describir en detalle el mundo real, anotando tanta información como fuera posible para representar el mundo real en la ontología. Sin embargo, con el crecimiento del número de sensores y datos, el tiempo para anotar y consultar las ontologías se ha convertido en un cuello de botella en el procesamiento en tiempo real de los datos provenientes de entornos de IoT. En los últimos años, algunos investigadores han concebido y aplicado la idea de modelos de información ligeros en el campo de IoT, como son el modelo IoT-Lite (Bermudez-Edo et al., 2017), y el modelo *Semantic Sensor Network* (SSN) (Compton et al., 2012). La ontología SSN es uno de los esfuerzos más significantes en el desarrollo de un modelo de información para los datos de sensores. La misma provee un vocabulario para describir conceptos como sensores, salidas, valores de observación y características de interés. Las extensiones más notables incluyen ontologías con características, servicios y roles para la respuesta a emergencias. Sin embargo, aunque la ontología SSN define un esquema de alto nivel para sistemas de sensores, no incluye una representación de los datos de observación y de medición (Kolozali et al., 2014).

SSN recientemente ha publicado una ontología ligera, SOSA (*Sensor, Observation, Sample and Actuation*) (Janowicz et al., n.d.), que se ha convertido en la ontología núcleo de la nueva versión del compendio de ontologías, SSN. Sin embargo, los modelos ligeros mencionados,

IoT-Lite y SOSA, están centrados en los dispositivos y no le prestan la atención necesaria a los *data streams* de IoT.

1.8.2. Stream Annotation Ontology

Representar *data streams* de IoT es un requerimiento importante en las aplicaciones semánticas de data streams, así como los entornos basado en conocimiento.

La *Stream Annotation Ontology* (SAO) (Kolozali et al., 2014) puede ser usada para expresar las características de un data stream. Es un modelo semántico ligero, que ha sido desarrollada en el marco del proyecto CityPulse que contiene 4 módulos principales. La reducción dimensional de los *data streams* o las transformaciones de los *streams* obtenidas a través de ventanas desplazadas puede resultar en una razón de datos, diferente de la frecuencia de muestreo de la observación del sensor original. Utilizando SAO, podemos describir un *data stream* y una instancia de línea de tiempo para enlazar la descripción del segmento con la extensión de tiempo de una unidad temporal que representa el data stream. Por tanto, podemos expresar un *data stream* como un intervalo de tiempo en la línea de tiempo universal, y también relacionar tal intervalo con el intervalo correspondiente en la línea de tiempo discreta junto con su frecuencia de muestreo discreta. Con respecto a las conceptualizaciones previas de los datos de sensores, la ontología SAO se ocupa de la representación de *stream data* y sus características temporales. Es libre de la organización taxonómica profunda y no intenta describir las interrelaciones o cálculos profundo del *stream data*.

SAO ha sido desarrollada sobre ontologías como SSN, PROV-O (Paolo, n.d.) y TimeLine (Raimond, 2008) e incluye conexiones con las ontologías Complex Event Processing y Quality. Los conceptos *StreamData, StreamEvent, StreamAnalysis, Observation, Sensor* y *Segment* están disponibles en esta ontología para describir conceptos temporales con precisión. Con la clase *StreamData*, SAO describe un *data stream* como un punto o segmento temporal y la salida de la observación como un evento con clase *StreamEvent*.

Esta ontología ha sido exitosamente utilizada en análisis forense y en algunos análisis casi en tiempo real. Sin embargo, cuando tiene que lidiar con una gran cantidad de datos, con una alta granularidad empezó a retrasar el proceso de análisis.

1.8.3. IoT-Stream.

IoT-Stream (Elsaleh et al., 2020a) es un modelo semántico ligero para la anotación de *data streams*, que está centrado alrededor del concepto de un *stream* de IoT y extiende la ontología SOSA. La principal idea detrás de IoT-Stream es la simplicidad del modelo de información y, especialmente, de los *streams* individuales, que son la parte más pesada de las anotaciones, ya que ellos representan la mayor parte de la información anotada. Por tanto, cada *stream* de observación está compuesto sólo por un valor y una marca de tiempo. Por eso se han segregado todos los metadatos necesarios con fines de búsqueda y rastreo, pero no requeridos para el procesamiento de datos en tiempo real. Además, permite anotar datos en bruto, así como datos procesados, ambos serán anotados como *streams*. Esta propuesta mejora la ontología SAO, reduciendo la cantidad de tripletas necesarias para consultar los *streams* de datos.

Los modelos semánticos para la anotación de *streams* son escasos, debido a que la semántica es pesada para procesar y no es ideal para entornos de IoT donde los datos son frecuentemente actualizados. IoT-Stream toma ventaja del conocimiento común compartido por la semántica, pero mantiene las interfaces y las consultas simples. El modelo ha sido diseñado acorde a las guías más reconocidas para desarrollar modelos semánticos y, especialmente, para entornos de IoT, donde la escalabilidad y tiempos de procesamientos cortos son esenciales. Con el concepto fundamental, *StreamObservation*, con simplemente propiedades de valor y tiempo, se aceleran las consultas al modelo de *streams* y se han creado el resto de conceptos necesarios para la búsqueda y rastreo alrededor de este concepto. Haciendo esto, se mejora el tiempo de procesamiento de las consultas a los *streams* (Elsaleh et al., n.d.).

1.9. Conclusiones Parciales del Capítulo.

- La Agricultura de Precisión tiene cada más presencia en el desarrollo de los cultivos en nuestro país, un ejemplo de esto es la UEB Agropecuaria Cubasoy que ya cuenta con una estación agrometeorológica para la monitorización de las condiciones climáticas.
- La Web Semántica resuelve el problema del IoT de tener un sinnúmero de datos provenientes de fuentes heterogéneas y logra clasificarlos y analizarlos mediante técnicas de *Big Data*.

- Los vocabularios para describir datos en *streaming* son escasos, la ontología predominante durante varios años ha sido SAO; pero recientemente se ha desarrollado la ontología Iot-Stream que es un modelo más ligero y mucho más simple.

CAPÍTULO 2. Modelo de streams e infraestructura IoT para la Agricultura de Precisión

Este capítulo estará dedicado a mostrar las características de las arquitecturas IoT, la elección de la plataforma adecuada y de las herramientas necesarias, así como el manejo de los datos en la ontología IoT-Stream.

2.1. Características físicas, productivas y tecnológicas de la Empresa Agropecuaria Cubasoy

La entidad Agropecuaria Cubasoy se dedica desde el año 2007 a la producción de soya, ya en el año 2010 se le incorpora el cultivo de maíz. Desde el año 2007 hasta el 2013 se ejecutan inversiones mejorando así la infraestructura administrativa y de servicios, sistemas de riego, viales, mecanización agrícola, industriales y de investigación y desarrollo. En el mes de julio del 2017 se traspasa esta entidad al Grupo Agrícola del MINAG, con la finalidad de producir viandas, cultivos varios, granos y hortalizas.

La Dirección de la Empresa se encuentra enclavada en el poblado de “La Carolina”, municipio Venezuela, al este de la carretera Ciego de Ávila – Venezuela, a 12 km de la cabecera provincial y a 6 km del poblado de Venezuela (Otero Barrera, 2019). Actualmente cuenta con más de 15 mil hectáreas en producción o fomento, en las que sobresalen maíz, yuca, plátano, calabaza, pepino y frutales, entre otros. Mantiene bajo riego unas 4700 hectáreas plantadas de viandas y hortalizas, gracias al funcionamiento de unas 96 máquinas de pivote central (Agencia Cubana de Noticias, 2020).

La UEB además cuenta con una red de sensores inalámbricos que posee nodos integrados por un transceptor de radio frecuencia (RF), sensores, un microcontrolador y la fuente de alimentación que brindan información de la humedad y temperatura del suelo. Además, posee una estación agrometeorológica que ofrece datos de velocidad y dirección del viento, precipitaciones, temperatura y humedad relativa del ambiente.

Los suelos de la zona, en su gran mayoría, son del tipo Ferralíticos, de ellos los Rojos en mayor cuantía, pero también posee Ferralíticos amarillentos y Ferralíticos pardos rojizos, de relieves llanos con profundidades de forma general desde 0.5 – 1.0 m (Otero Barrera, 2019).

La entidad cuenta con una infraestructura administrativa y productiva creada que tiene un buen funcionamiento, una fuerza de trabajo capacitada, una correcta asimilación de la ciencia y la técnica, se encuentra conectada a la red nacional de entidades agrícolas vía Wifi, se espera conectarla mediante fibra óptica en los años venideros, además en la misma existe conexión a Internet.

2.2. Arquitecturas de IoT

No existe un solo consenso de arquitectura en IoT que sea adoptado universalmente. Diferentes arquitecturas han sido propuestas por diferentes investigadores y éstas pueden tener distintas clasificaciones teniendo en cuenta en base a qué fueron creadas. En base a los protocolos utilizados las propuestas realizadas tienen analogías con los conocidos modelos TCP/IP y OSI (Maury Yera, 2015).

La distinción entre arquitecturas de protocolos y arquitecturas de sistemas no es muy nítida. Usualmente los protocolos y el sistema son codiseñados.

2.2.1. Arquitectura de 3 Capas

La arquitectura más básica es la arquitectura de 3 capas (figura 2.1 A). Fue introducida en las etapas iniciales de investigación en esta área. Se compone por la capa de percepción, la de red, y la de aplicación (Maury Yera, 2015).

- Capa de Percepción: Es la capa física, que posee sensores para medir y recolectar información sobre el entorno. Mide algún parámetro físico o identifica otros objetos en el entorno.
- Capa de Red: Es la responsable de conectar la capa de percepción con otras cosas inteligentes, los dispositivos de red y los servidores. Sus características son también utilizadas para transmitir y procesar los datos de los sensores.
- Capa de Aplicación: Es la responsable de converger entre las necesidades sociales de la IoT y tecnología industrial, es decir, puede ser considerado como el nivel

intermedio entre las tecnologías de la industria y la forma en que se puede controlar para cubrir las necesidades humanas.

La arquitectura de tres capas define la idea principal del Internet de las Cosas, pero no es suficiente para investigar en el IoT debido a que la investigación usualmente se centra en aspectos más finos del Internet de las Cosas.

2.2.2. Arquitectura de Cinco Capas

La arquitectura de cinco capas (figura 2.1 B), le adiciona las capas de procesamiento y de negocio al modelo de tres capas. Las cinco capas son percepción, transporte, procesamiento, aplicación y negocio. El papel de las capas de percepción y aplicación es el mismo que en la arquitectura de 3 capas (Sethi and Sarangi, 2017).

- Capa de Transporte: transfiere los datos de los sensores de la capa de percepción a la capa de procesamiento y viceversa a través de redes que pueden ser inalámbricas, 3G, LAN, Bluetooth, RFID y NFC.
- Capa de Procesamiento: También conocida como la capa *middleware*. Almacena, analiza y procesa una gran cantidad de datos que arriban desde la capa de transporte. Puede manejar y proveer varios servicios a las capas inferiores. Emplea tecnologías como bases de datos, computación en la nube y módulos de procesamiento de *Big Data*.
- Capa de Negocio: Maneja el sistema IoT completo, incluyendo sus aplicaciones, negocios, modelos de ganancias y privacidad del usuario. No corresponde con ninguna otra capa de las arquitecturas de redes existentes. A partir de análisis de resultados se determina las futuras acciones y estrategias de negocio.

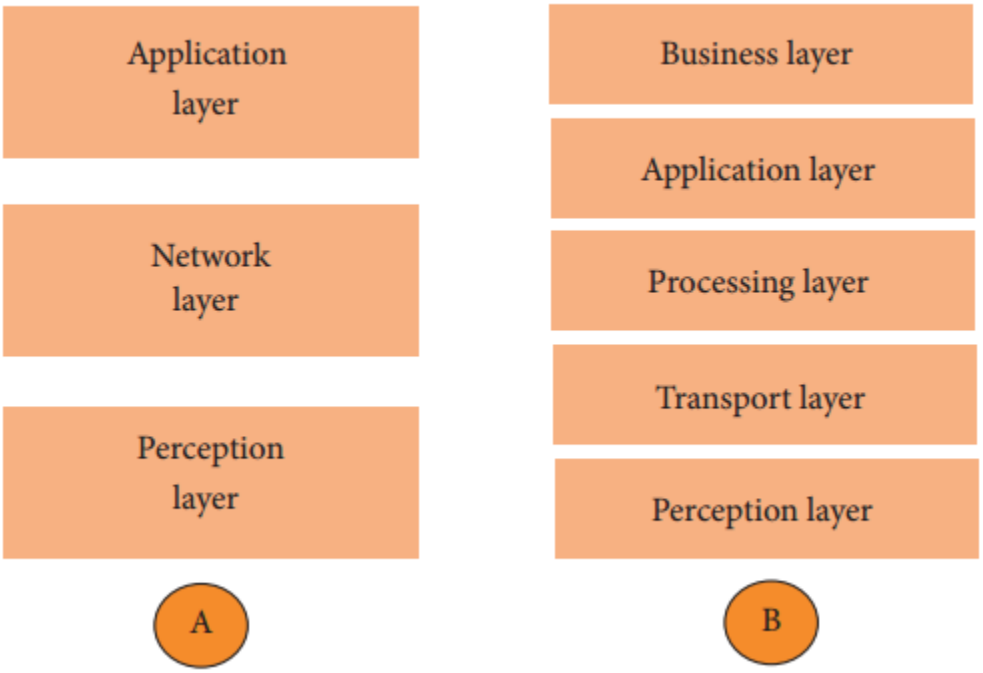


Figura 2.1. Arquitecturas en IoT (A: 3 capas) (B: 5 capas).

2.3. Arquitecturas basadas en la Nube y en la Niebla

En algunas arquitecturas el procesamiento de los datos es realizado en las muy conocidas computadoras en la nube. Esta arquitectura centrada en la nube mantiene a la nube en el centro, las aplicaciones sobre la misma y la red de cosas inteligentes debajo de ella. A la computación en la nube se le da primacía debido a que provee gran flexibilidad y escalabilidad. Ofrece servicios como el núcleo de la infraestructura, plataforma, software y almacenamiento (Sethi and Sarangi, 2017).

Recientemente, hay un movimiento hacia otra arquitectura del sistema llamada computación en la niebla (*fog computing*), donde los sensores y la puerta de enlace de la red hacen una parte del procesamiento y análisis de los datos. Una arquitectura en la niebla representa un enfoque en capas que inserta monitorización, preprocesamiento, almacenamiento y seguridad entre las capas físicas y de transporte. La capa de monitorización supervisa energía, recursos, respuestas y servicios. La capa de preprocesamiento realiza un filtrado, procesamiento y análisis de los datos de los sensores. La capa de almacenamiento temporal provee funcionalidades de almacenamiento tales como replicación, distribución y almacenamiento

de datos. Finalmente, la capa de seguridad realiza la encriptación/desencriptación y asegura la privacidad e integridad de los datos (Sethi and Sarangi, 2017).

Para el propósito de esta investigación se ha decidido tomar como arquitectura de protocolos a la arquitectura de cinco capas debido a que es más completa y describe mejor el funcionamiento de un sistema IoT y la arquitectura de sistema tomada es la computación en la niebla ya que convierte los flujos de datos de la red en información adecuada para su almacenamiento y procesamiento en los niveles superiores proveyendo un menor tiempo para acceder a estos servicios. Además, posee una respuesta en tiempo real y puede interactuar con la nube comunicándole solo los datos necesarios.

2.4. Selección de los protocolos de red en IoT

El IETF (*Internet Engineering Task Force* por sus siglas en inglés) ha desarrollado protocolos alternativos para la comunicación entre los dispositivos IoT utilizando IP debido a que es un estándar flexible y confiable. La Alianza IPSO (*Internet Protocol for Smart Objects* por sus siglas en inglés) ha publicado varios escritos describiendo protocolos alternativos y estándares para las capas del modelo IP y se ha adicionado una capa de adaptación que es utilizada para la comunicación entre los objetos inteligentes (Sethi and Sarangi, 2017).

2.4.1. Capa Física y Enlace

El protocolo IEEE 802.15.4 ha sido diseñado para permitir la comunicación entre dispositivos de bajo consumo embebidos que necesitan una larga duración de la batería. Define estándares y protocolos para la capa física y de enlace (MAC) del modelo IP. La transmisión requiere muy poca energía (máximo de un miliwatt) que es solo 1 % de la utilizada en las redes Wifi o celular. Esto limita el rango de la comunicación por lo que los dispositivos deben operar cooperativamente en función de realizar un ruteo multisalto para largas distancias. El esquema de codificación en IEEE 802.15.4 ha sido construido sobre la redundancia, lo que permite una comunicación robusta, detectar pérdidas y la retransmisión de paquetes perdidos (Vasseur et al., 2010).

2.4.2. Capa de Adaptación

IPv6 es considerado el mejor protocolo para la comunicación en el dominio IoT debido a su escalabilidad y estabilidad. El estándar 6LoWPAN conocido como *IPv6 over Low Power*

Wireless Personal Area Network soluciona el problema de transportar paquetes IPv6 sobre tramas IEEE 802.15.4. Es una alternativa al protocolo ZigBee, al implementar direccionamiento IPv6 en vez de IPv4, debido a que ofrece mayores ventajas (espacio, direccionamiento, escalabilidad, autoconfiguración, entre otras) y logra la interconexión directa de las Redes de Sensores Inalámbricos con otras redes externas. El motivo del surgimiento de este protocolo fue el gran tamaño de las direcciones IPv6, por lo que proporciona mecanismos de fragmentación y compresión de cabeceras para la transmisión de paquetes permitiendo adaptarse a las restricciones del protocolo 802.15.4 (máximo 127 bytes) llegando a alcanzar un tamaño de 4 bytes (Estevez Pérez, 2018).

2.4.3. Capa de Red

El organismo IETF desarrollo el protocolo de ruteo RPL, diseñado con el objetivo de cubrir las necesidades de enrutamiento en redes de baja potencia. RPL se caracteriza por ser un protocolo proactivo basado en vector de distancia, el cual define las rutas antes de que sean necesarias por los nodos de la red. Su modo de funcionamiento está basado en el intercambio de mensajes de control (enviados de forma periódica) para encontrar y propagar rutas en la red, estos mensajes pueden ser a nivel local o de enlace permitiendo el envío de información a los vecinos o a nivel global para propagar la información relacionada con la topología a todos los nodos de la red. El no tener una capa de enlace definida es una de las ventajas de RPL, por lo que está diseñado para soportar varias de ellas (Estevez Pérez, 2018).

2.4.4. Capa de Transporte

TCP no es una buena opción para la comunicación en entornos de baja potencia debido a que tiene una alta sobrecarga unido al hecho que es un protocolo orientado a la conexión. Por tanto UDP es preferido debido a que es un protocolo sin conexión previa y posee una baja sobrecarga (Sethi and Sarangi, 2017). Proporciona sumas de comprobación para la integridad de datos y números de puertos para direccionar diferentes funciones en la fuente y el destino del datagrama. Para el propósito de esta investigación se ha seleccionado el protocolo UDP ya que al procesar *streams* de datos se necesita una muy alta velocidad de comunicación.

2.4.5. Capa de Aplicación

En la búsqueda de la normalización e interoperabilidad del IoT, se presentan protocolos de comunicación muy prometedores que pueden convertirse en un estándar para aplicaciones de

este tipo. Ejemplo de ello, son los protocolos MQTT (*Message Queue Telemetry Transport*) y CoAP (*Constrained Application Protocol*), diseñados especialmente para dispositivos con recursos limitados en cuanto a procesamiento y ancho de banda.

CoAP es un protocolo M2M ligero que soporta tanto la arquitectura de solicitud/respuesta como la de recurso/observación. Se desarrolla principalmente para interoperar con HTTP y RESTful Web a través de *proxies* simples. El publicador publica datos en el URI y el suscriptor se suscribe a un recurso específico indicado por el URI. Cuando un publicador publica nuevos datos en el URI, entonces todos los suscriptores son notificados sobre el nuevo valor. CoAP utiliza UDP como protocolo de transporte DTLS para la seguridad. Utiliza mensajes "confirmables" o "no confirmables" para proporcionar dos niveles de QoS (Cárdenas Rivero, 2019).

MQTT es un protocolo de mensajería para publicar y suscribirse diseñado para comunicaciones M2M ligeras. Fue desarrollado originalmente por IBM y ahora es un estándar abierto. En cuanto a su arquitectura, MQTT tiene un modelo de cliente/servidor, donde cada sensor es un cliente y se conecta a un servidor, conocido como *broker*, sobre TCP. MQTT está orientado a mensajes, donde cada mensaje es un segmento discreto de datos. Cada mensaje se publica en una dirección, conocida como tema (*topic*). Los clientes pueden suscribirse a varios temas. Cada cliente suscrito a un tema recibe todos los mensajes publicados en el tema. El modelo de publicación/suscripción permite a los clientes comunicarse uno a uno, uno a muchos y muchos a uno.

MQTT se enfoca a una mensajería confiable, por lo que incluye búferes de mensajes y niveles de QoS controlados por el *broker*. Los clientes MQTT pueden registrar un mensaje personalizado que podrá ser enviado por el *broker* si se interrumpe la conexión, indicando a los suscriptores cuando un dispositivo se desconecta. A pesar de que MQTT está diseñado para ser ligero, tiene dos inconvenientes para dispositivos muy limitados. Cada cliente de MQTT debe soportar TCP y tendrá típicamente una conexión abierta al *broker* en todo momento. En algunos entornos donde la pérdida de paquetes es alta o los recursos de cómputo son escasos, esto es un problema. El otro problema es que los nombres de temas de MQTT son a menudo cadenas largas que los hacen imprácticos para el estándar IEEE 802.15.4. Ambas deficiencias son resueltas por el protocolo MQTT-SN, que define un mapeo

UDP de MQTT y añade soporte del *broker* para la indexación de temas (Cárdenas Rivero, 2019).

MQTT-SN es un protocolo de publicación/suscripción para Redes de Sensores Inalámbricos considerado una versión de MQTT que se adapta a las particularidades de un ambiente de comunicación inalámbrica y utiliza UDP como protocolo de transporte. Los enlaces de radio inalámbricos tienen en general tasas de fallo más altas que las cableadas debido a su susceptibilidad a la atenuación y a la interferencia, presentan una tasa de transmisión más baja y sus paquetes de datos tienen una longitud muy corta para hacerlos resistentes a errores de transmisión. MQTT-SN está diseñado para asemejarse lo más posible a MQTT, pero se adapta a las peculiaridades de un entorno de comunicación inalámbrica con poco ancho de banda, fallas de enlace, longitud corta de mensajes, etc. Está optimizado para la implementación en dispositivos de bajo costo, alimentados con baterías y con recursos de procesamiento y almacenamiento limitados (Cárdenas Rivero, 2019).

Para el caso específico de esta investigación se utiliza el protocolo MQTT-SN, debido a que permite el protocolo de transporte UDP, es compatible con WSN, el tamaño del encabezado es más pequeño y al ser muy similar a MQTT tiene 3 niveles de QoS y tiene mayor soporte por parte de la comunidad.

2.5. Middleware

La interoperabilidad entre dispositivos heterogéneos necesita estándares muy bien definidos, pero la estandarización es una dificultad debido a la variedad de requerimientos de los diferentes dispositivos y aplicaciones. Para estas aplicaciones heterogéneas la solución es tener una plataforma *middleware,* que abstraerá los detalles de los dispositivos para las aplicaciones. Actúa como un software puente entre los dispositivos y las aplicaciones. Le provee los servicios requeridos a los desarrolladores de aplicaciones para que se puedan enfocar más en los requerimientos de la aplicación que en tratar de interactuar con el hardware. En resumen, el *middleware* abstrae el hardware y provee APIs (*Application Programming Interface*) para la comunicación, manejo de datos, computación, seguridad y privacidad (Sethi and Sarangi, 2017).

Los *middlewares* pueden ser clasificados según la base de su diseño:

- Basado en Eventos: Todos los componentes interactúan entre ellos a través de eventos. Cada evento tiene un tipo y algunos parámetros. Estos eventos son generados por los productores y recibidos por los consumidores; esto puede ser visto como una arquitectura de publicación/suscripción, donde las entidades pueden suscribirse para algún tipo de evento y obtener notificaciones de esos eventos.
- Orientado al Servicio: Están basados en la Arquitectura Orientada al Servicio (SOA), en donde se tienen módulos independientes que proveen servicios a través de interfaces accesibles. Un *middleware* orientado al servicio ve a los recursos como proveedores de servicios y abstrae los recursos subyacentes a través de una serie de servicios que son utilizados por las aplicaciones. Hay un repositorio de servicios donde los proveedores suscriben los servicios y los consumidores los descubren a través de este repositorio y de este modo se enlazan.
- Orientado a la Base de Datos: En este enfoque, red de dispositivos IoT es considerada como un sistema virtual de base de datos relacional. Esta base de datos puede ser consultada por las aplicaciones utilizando un lenguaje de consultas, además existen interfaces fáciles de usar para extraer los datos de la base de datos. Este enfoque posee problemas con la escalabilidad debido a que es un modelo centralizado.
- Semántico: Un *middleware* semántico se enfoca en la interoperación de diferentes tipos de dispositivos que se comunican utilizando diferentes formatos de datos. Incorpora dispositivos con diferentes formatos de datos y ontologías y los ata a todos en un *framework* común. En este *middleware* una capa semántica es introducida que es donde ocurre un mapeo de cada recurso a una capa de *software* para ese recurso. Las capas de *software* se comunican entre ellas utilizando un lenguaje mutuamente inteligible. Esta técnica permite a varios recursos físicos comunicarse, aunque no implementen o entiendan los mismos protocolos.
- Específicos de aplicación: Este tipo de *middlewares* es utilizado específicamente para el dominio de una aplicación para el que es desarrollado debido a que toda la arquitectura de este *middleware* ha sido diseñada en base a los requerimientos de determinada aplicación.

Para los propósitos de esta investigación que se centra en el uso de ontologías para el manejo de datos en tiempo real se ha decidido implementar un *middleware* semántico. A pesar de

existir varias opciones en el mercado, el *middleware* utilizado será el propietario de la plataforma IoT seleccionada, la cual se explicará con más detalle en el siguiente epígrafe.

2.6. Plataforma IoT Sofia2

Sofia2 ha sido pensada para facilitar y acelerar la construcción de nuevos sistemas y soluciones digitales y así lograr la transformación y disrupción en los negocios. Su propósito es lograr la interoperabilidad entre diferentes aplicaciones que comparten conceptos semánticos.

Es una plataforma integrada, con seguridad integrada en todos los elementos, con un panel de control centralizado, permite ingesta, análisis y procesamiento de *Big Data*, ingesta y análisis de información de las redes sociales, es multidispositivo, adaptable a las necesidades, posee capacidad de representación a través de un visor holístico, un modelado visual de la semántica, escalabilidad horizontal, puede ser instalado en el cliente o en la nube y posee versiones *OpenSource* y Comercial (Cebrián and Manuel, 2017).

2.6.1. Características IoT

FEEP IoT *Platform* Sofia2, es un *Middleware* y repositorio que permite la interoperabilidad en tiempo real entre sistemas, redes sociales, dispositivos y sensores. Posee conectores de comunicación para diversos clientes y protocolos de comunicación ligeros (REST, WebSockets, MQTT, WS, JMS, AMQP, etc.). Es extensible en Java (APIs, Protocolos, *Plugins*) y APIs de desarrollo de clientes proporcionados en diversos lenguajes. Posee auditoría de la actividad de integración, configuración de reglas sencillas y complejas ejecutadas en tiempo real, gestión y configuración integrado en Sofia2 *Control Panel* (HTML5), API *Manager* integrado basado en estándares (JSON, REST, RESTful) que incluye control completo del ciclo de vida de las APIs. Además, posee seguridad integrada con el resto de elementos de la plataforma (autenticación, autorización, cifrado), integración transparente de APIs de terceros y el control de *Throtling* (gestión del número de peticiones que podrá realizar cada usuario por minuto).

2.6.2. Módulos IoT

Estos son los módulos necesarios para dar soporte a sistemas IoT:

- SDK: La plataforma provee un set de herramientas para desarrolladores que facilita el desarrollo de clientes en diferentes lenguajes y sobre una gran variedad de protocolos.
- Sofia2 Panel de Control: La plataforma ofrece una completa web de administración/configuración que permite gestionar todos los conceptos que maneja la plataforma. El resto de módulos de la plataforma se operan/configuran desde este módulo, que persiste su configuración en la BDC (Base de Datos de Configuración) del Repositorio de Sofia2. Esta consola es accesible para los diferentes roles de la plataforma, permitiendo operar de una u otra forma en función de este rol. Mediante interfaces normalizadas, permite una representación estructurada de la información y un diseño de operación destinado a sacar el máximo rendimiento de los usuarios del sistema, facilitando el aprendizaje y reduciendo el tiempo de respuesta.
- IoT *Gateway*: capa de abstracción del protocolo de comunicación, que implementa el protocolo SSAP, sobre diferentes protocolos y facilita la incorporación de nuevos protocolos gracias al despliegue de nuevos *Plugins*. De esta manera, la información gestionada por las subsiguientes capas de la plataforma es completamente agnóstica del protocolo tecnológico usado para el envío del dato, dando lugar a su gestión desde un punto de vista semántico. Este módulo se especializa en el esquema de comunicación con dispositivos, sensores sistemas en un contexto IoT, donde se debe facilitar el uso de protocolos de comunicación ligeros en un entorno tecnológico heterogéneo. El protocolo SSAP proporciona tanto la ligereza del mensaje como su homogeneización a nivel de aplicación.
- *Semantic Broker*: módulo de la plataforma que recibe, procesa y almacena toda la información de las aplicaciones, sensores, y dispositivos conectados, actuando como Bus de Interoperabilidad. Esta capa validará la corrección sintáctica y semántica del dato recibido gracias a la definición previa de la estructura del dato esperado (ontología), identificando de qué dato se trata y aplicando la seguridad correspondiente al mismo. La plataforma ofrece el concepto de *plugin* como mecanismos de extensión de la plataforma que permite incorporar nuevas funcionalidades en esta de forma sencilla. Mediante el despliegue de *Plugins* se podrá ampliar o adaptar la funcionalidad por defecto de este componente. El motor de

Plugins permite crear nuevos conectores, cambiar autenticación, auditar información, persistir en otros repositorios, generar KPIs, monitorizar, etc., es decir, dotan de una flexibilidad máxima a la plataforma.

- *Process*: módulo que incluye 2 motores para la definición de reglas a aplicar sobre la información que entra en la plataforma: el motor de Reglas (Sofia2-Rules) y el motor CEP (Sofia2-CEP). El motor de reglas permite ampliar el funcionamiento de la plataforma permitiendo definir reglas que se ejecuten ante ciertas condiciones (inserción de un nuevo dato o cada cierto tiempo). Estas reglas dan la capacidad de definir, en base a *Scripting*, acciones que ejecutan la plataforma, gracias a ellas se pueden gestionar y tratar los datos de la plataforma. El motor CEP permite definir reglas en las que interviene el tiempo (por ejemplo, que no ha llegado una cierta medida en 1 día). A los eventos generados por el motor CEP pueden suscribirse los clientes o servir como entrada al motor de reglas.
- Sofia2 *Storage*: módulo de almacenamiento de la información de la plataforma. Se compone de 3 repositorios: Base de Datos Tiempo Real (BDTR), Base de Datos Histórica (BDH) y Área de *Staging* (HDFS). BDTR almacena la información recibida en tiempo real, como instancias de ontologías, siendo, por lo tanto, el primer repositorio en el que almacena la información recibida de sensores y dispositivos integrados con la plataforma en un contexto IoT típico, esta información se valida automáticamente, garantizando corrección de la estructura según la definición previa de las ontologías. Por cada ontología se puede configurar una ventana de tiempo a partir de la cual la información ya no se considera en tiempo real, de manera que será migrada automáticamente al repositorio de información histórica. BDH almacena la información histórica para su posterior explotación analítica. El repositorio *Staging* almacena información en diferentes estados (estructurada, semiestructurada y no estructurada) temporalmente, para facilitar procesos complejos de transformación, ingestión y exposición de datos que requieran la persistencia temporal de estados intermedios del proceso.
- Sofia2 API *Manager*: permite publicar la información gestionada por la plataforma como APIs REST ya su vez permite la búsqueda de estas APIs, la suscripción por parte de clientes y la gestión del versionado y ciclo de vida de cada una de ellas.

Además, este API *Manager* permite la disponibilidad de Servicios REST externos a la plataforma, lo que permite ofrecer un punto único de acceso a APIs internas y externas de la plataforma.

- *Holystic Viewer*: módulo de visualización avanzada de la plataforma, que soporta diferentes motores. Se trata de un sistema integral de visualización avanzada e interactiva que permite una gestión de información geolocalizada asociándola a un entorno de visualización tridimensional y multimedia. Sus capacidades de visualización avanzada resultan un plus de interés a la creación de cuadros interactivos.

2.7. Arquitectura del sistema y manejo de datos con la ontología IoT-Stream

Para utilizar IoT-Stream en sistemas orientados al IoT es necesario definir las entidades del sistema y los requerimientos necesarios relacionados a la anotación, publicación, persistencia, consulta y subscripción a la ontología (Elsaleh et al., 2020a).

2.7.1. Entidades del sistema

Para que un sistema soporte la adopción de IoT-Stream, las entidades del sistema requeridas, que se muestran en la figura 4, serían:

- Registro: responsable primario de almacenar información sobre *IoTStream* en una tripleta y exponer un punto final SPARQL para manejar consultas. También puede ser empleado por un productor de *IotStream* para almacenar *StreamObservations*, asumiendo así el papel de un servicio IoT.
- Productor: responsable de registrar *IotStreams* produciendo las *StreamObservations* generadas por sus sensores. Si puede almacenar y exponer *StreamObservations*, puede actuar como un servicio IoT.
- Consumidor: una aplicación o servicio que descubra *IotStreams* a través del registro y consuma *StreamObservations*. En el contexto del análisis de datos, el consumidor puede consumir datos procesados o *StreamObservations* pre-procesada (como agregación o filtración).
- *Broker*: un agente alternativo persistente de *StreamObservations*, en el cual los Consumidores y los Servicios pueden suscribirse a las *StreamObservations* en

tiempo real publicadas por el Productor a un *broker* de mensajes. En este caso, la persistencia depende del Consumidor.

- Servicio de Análisis: empleado por el Consumidor o el Productor para consumir o generar *IotStreams* analizados, aplicando una técnica específica de análisis de datos con un método específico o un conjunto de métodos. Puede ser parte del sistema del Consumidor o un microservicio externo que se enfoca en un tipo de análisis particular.

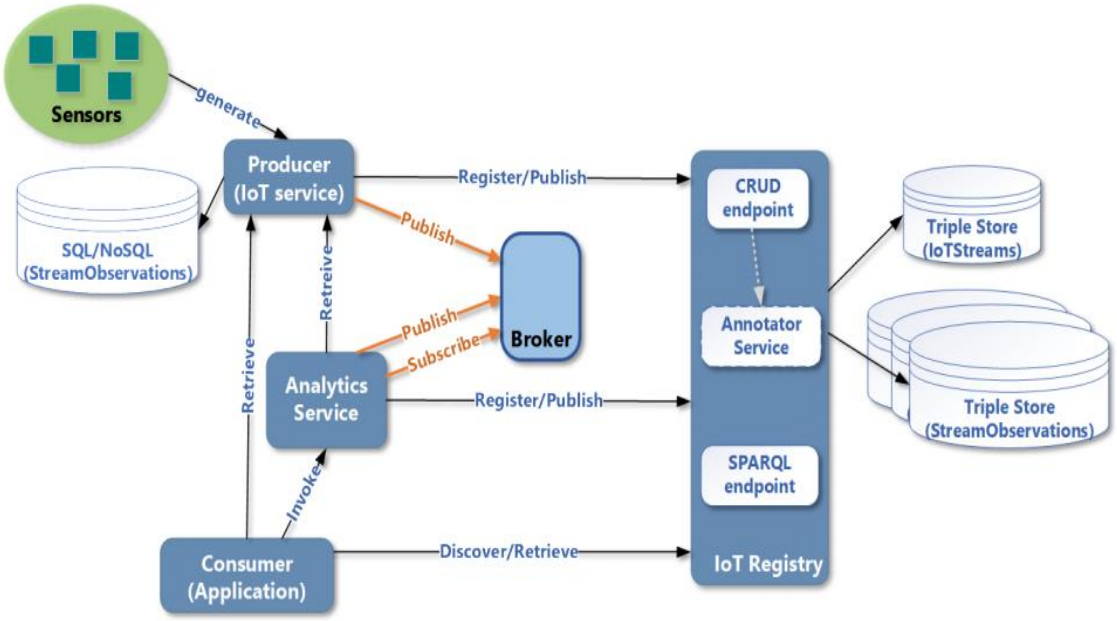


Figura 2.2. Entidades del sistema y su interacción para la adopción de *IotStream*.

2.7.2. Flujo de datos entre las entidades del sistema

Un Productor normalmente tendría un modelo de información propietaria para modelar los datos y metadatos de su sensor. Para compartir los datos externamente, el Productor necesita registrar y publicar sus datos en un Registro accesible externamente. El Registro puede tener dos roles; el primero como un directorio y el segundo, como un repositorio. Como un directorio, el Registro solo almacenará información sobre los *streams* y no las observaciones de los *streams* en una tripleta RDF. Aquí, el Productor transforma sus descripciones propietarias e instancia un *IotStream* utilizando un anotador semántico. Un *IotStream* individual es creado, con el tiempo de *streamStart*. También crea una instancia de *sosa:*

Sensor individual con su correspondiente *qu: QuantityKind* y *qu: Unit* y, además, agrega la localización del *IotStream*. Por último, adiciona el *iot: Service* que define el punto final y la interfaz por la que serán proporcionadas las *StreamObservations*. El servicio puede estar alojado en el dominio del Productor o en el *broker* delegado. Si el Registro asume el rol de repositorio, entonces los Productores pueden delegar el servicio al mismo Registro. Entonces el Productor comienza a publicar *StreamObservations* al servicio IoT delegado, al cual el Consumidor puede suscribirse y así recuperar las *StreamObservations*. Si el Consumidor no desea consumir las *StreamObservations* como han sido generadas originalmente por los sensores, puede emplear un Servicio de Análisis para recuperar o suscribirse a ellas. Aquí, el Consumidor registra un nuevo *IotStream* en el Registro y envía al Servicio de Análisis el Servicio IoT para obtener las *StreamObservations* y llama de nuevo al Servicio para publicar las nuevas *StreamObservations* ya analizadas. El Consumidor puede emplear otro Servicio de Análisis para consumir el *IoTStream* analizado para detectar Eventos (Elsaleh et al., 2020a).

2.7.3. Anotaciones

La ontología *IotStream* ha sido diseñada para permitir flexibilidad en la forma que las *StreamObservations* son anotadas y almacenadas. La anotación puede ser realizada en el nivel atómico, por lo cual una observación corresponde a un dato y puede ser no atómica, donde las observaciones corresponden a un conjunto de datos representados como un vector de cadenas. Alternativamente, se puede anotar el servicio que provee los *streams*, pero mantener los *streams* sin anotaciones semánticas. En este caso, las *StreamObservations* son recibidas desde un servicio IoT externo. Este servicio puede servir *StreamObservations* utilizando formatos de datos simples como CSV o JSON. De lo contrario, si el Consumidor los requiere en RDF, entonces se puede utilizar anotador semántico "*on-the-fly*" para proveer *StreamObservations* en la variante RDF al ser requeridas, como son *Turtle* y JSON-LD (Elsaleh et al., 2020a).

Cada uno de estos tres enfoques vienen con sus ventajas y desventajas. El enfoque atómico para anotar proveerá menos carga de procesamiento del lado del Consumidor cuando se hace la consulta, pero más en el lado Registro. El no atómico producirá más procesamiento en el lado del Consumidor para analizar las *StreamObservations* provocando un alivio en el

Registro por no tener que buscar en más instancias. El último enfoque requerirá que el Consumidor haga una petición a otro punto externo de las *StreamObservations*, lo que requerirá que el Consumidor haga una segunda petición con una interfaz distinta, pero aliviará al Registro de almacenar esas *StreamObservations*.

En el caso de que el Productor sea alojado en un dispositivo de propiedades restringidas, un servicio anotador dedicado externo al dispositivo puede ser empleado. Este servicio típicamente requerirá las observaciones de los *streams* en un formato predefinido. Por ejemplo, un anotador dedicado puede transformar el formato CSV en JSON-LD, que sigue el modelo de IoT-Stream (Elsaleh et al., 2020a).

2.7.4. Almacenamiento y consultas

Las tripletas son adecuadas para almacenar información finita sobre entidades, pero no para datos temporales asociadas a ellas. En el caso de los datos temporales, como más datos son acumulados en un conjunto de datos, a las tripletas les resulta más difícil responder a las consultas en periodos razonables. Si una anotación atómica es requerida, entonces cada *StreamObservation* debe ser separada del resto de los metadatos, excepto su enlace con el *IotStream* al cual pertenece. En este caso, una consulta SPARQL puede ser utilizada para descubrir *IotStreams* y a su vez, recuperar las observaciones de otro conjunto de datos utilizando el *iot-lite: endpoint*. Si la recuperación de una *StreamObservation* es realizada sin utilizar SPARQL, *iot-lite: interfaceDescription* también se puede recuperar para saber qué parámetros pasar para recuperar una observación instantánea o un conjunto de observaciones dentro de una ventana definida. Por ejemplo, en un servicio RESTful, normalmente se utilizaría un WADL (*Web Application Description Language* por sus siglas en inglés) para este propósito (Elsaleh et al., 2020a).

Otro aspecto a tener en cuenta es la variabilidad de los metadatos asociados con un *IotStream*, como la ubicación y la calidad. En el caso de la ubicación, si el sensor generando los *IotStreams* es adjuntado a una entidad móvil, entonces la información de ubicación necesitará ser capturada y enlazada a cada *StreamObservation*. Una consulta en este caso necesitará incluir una verificación del estado de movilidad del sensor utilizando *iot-lite: isMobile*. El propio *IotStream* solo tendría la información de la ubicación actual vinculada a él. Para la calidad de la información, métricas como *qoi: Timeliness* pueden cambiar durante la vida útil

de un *IotStream*, debido a que un sensor experimenta problemas computacionales o de conectividad ya sea internamente, es decir, en el dispositivo, o causada externamente por un nodo intermediario como un *Gateway*. En estos casos, es necesario almacenar nuevas instancias de QoI con cada *StreamObservation*, o enlazarlas a cada una.

2.8. Herramientas y aplicaciones útiles para la adopción de IoT-Stream

2.8.1. Herramienta para el Análisis de Datos

Un sistema que consume *streams* de datos de IoT necesita emplear alguna forma de análisis de datos para manejar el grado de volumen, velocidad, intermitencia, irregularidad y dimensionalidad que los acompaña. Las librerías y *frameworks* para los lenguajes de programación populares han permitido la creación de herramientas para manejar datos de su naturaleza y los conocimientos esperados que se obtengan. Dependiendo de la aplicación, las herramientas implicarían algún tipo de preprocesamiento, aprendizaje automático, o correlación. El resultado de estas técnicas puede ser utilizado para enriquecer un grafo de conocimiento semántico. El servicio web *Knowledge Acquisition Toolkit* (KAT) (Ahrabian et al., 2017) proporciona una instalación que permite al consumidor que primero experimentar con las fuentes de datos IoT remotas con diferentes cascadas de métodos para estudiar cuál funciona mejor para ellos. Al exponer una interfaz RESTful, KAT puede consultar los *streams* de datos de un almacenamiento de *streams* de datos IoT semántico al aceptar una consulta SPARQL con un formato predefinido para las variables de salida. A su vez, el servicio generará un nuevo *data stream* basado en los métodos seleccionados y sus parámetros correspondientes. Las nuevas *StreamObservations* son anotadas y enlazadas a un nuevo *IotStream*, con los detalles del Análisis empleado y luego es enviado de vuelta al Consumidor. En la figura 5 se ilustra el proceso completo (Elsaleh et al., 2020a).

Muchas de las herramientas de análisis existentes ofrecen sistemas como *software* que requiere que el usuario o consumidor de los datos descargue el producto. Sin embargo, tales sistemas pueden requerir que el usuario aprenda un lenguaje específico del dominio (como Python o Java) para el uso de dichos sistemas, mientras también requiere de actualizaciones de *software* para acceder a las últimas técnicas de análisis de datos. La ventaja que ofrece KAT es que es una herramienta de análisis de datos de acceso abierto para los consumidores que se presenta como un servicio web. Además, le permite a los consumidores novatos o

principiantes analizar y obtener información útil de los datos. Mientras que para los usuarios más experimentados o avanzados proporciona herramientas más efectivas para un conjunto de datos dado (Ahrabian et al., 2017).

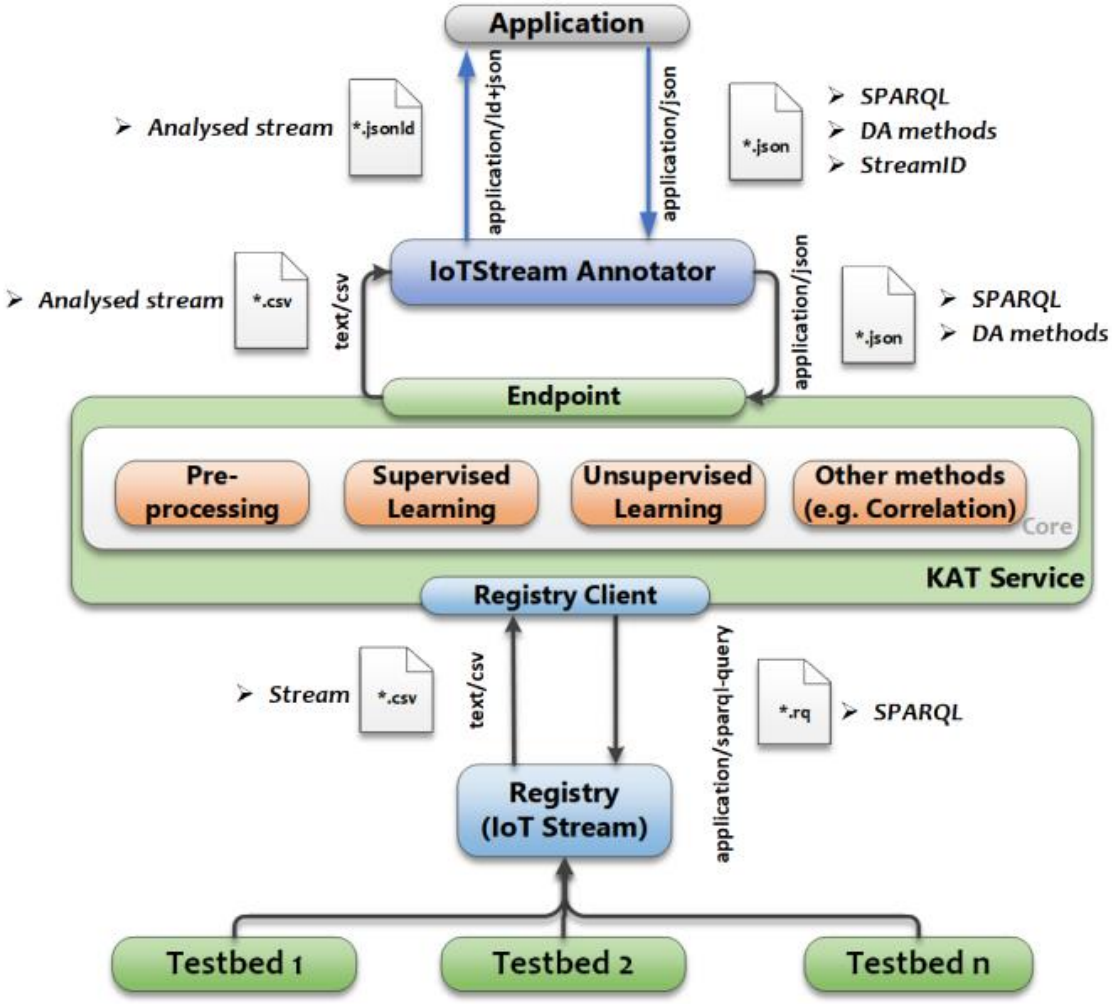


Figura 2.3. IoT-Stream analizado utilizando el servicio KAT.

2.8.2. Motor de Búsqueda y Rastreo para *data streams* en IoT

En contraste con los motores de búsqueda de la web, un motor de búsqueda para el IoT es utilizado principalmente por otras máquinas o aplicaciones que necesitan información para funcionar correctamente. Mientras que los usuarios humanos tienen la habilidad de evaluar la usabilidad del resultado de una búsqueda según sus necesidades, una máquina no es capaz de hacerlo. Es esperado que todos los resultados de la búsqueda satisfagan la consulta, debido a que no hay manera objetiva de decidir entre ellos. Por tanto, un motor de búsqueda de IoT

debe clasificar y organizar los resultados de antemano, incluso sin los requisitos específicos de la consulta. Por esto, debe utilizar toda la información disponible sobre el dispositivo IoT como confiabilidad y disponibilidad a largo plazo. Una máquina, además, requiere formatos de datos predefinidos y debe saber de antemano cómo interpretar un resultado de búsqueda recibido (Iggena et al., 2021).

Al igual que con cualquier motor de búsqueda convencional, buscar por los recursos disponibles en el momento de la solicitud de búsqueda no es factible. Para proporcionar resultados de búsqueda de manera oportuna, un repositorio o base de datos sobre las fuentes de los datos debe construirse con anticipación. Para disminuir el tiempo de búsqueda, los datos dentro del repositorio deben configurarse con los índices apropiados. Antes de todo, el motor de búsqueda debe conocer los dispositivos IoT. Esta es probablemente la tarea más desafiante ya que existe una gran variedad de dispositivos y posibilidades de configuración en IoT. Además, el dominio IoT es más dinámico que la *World Wide Web*. Si bien los servidores web permanecen en línea y estacionarios durante un largo período de tiempo, los dispositivos de IoT pueden aparecer y desaparecer con frecuencia. Por lo tanto, una vez que se ha identificado un dispositivo de IoT e integrado en la base de datos del motor de búsqueda, es necesario supervisar su disponibilidad y la calidad en la transmisión (Iggena et al., 2021).

Para el *framework* IoTCrawler se ha adoptado un concepto de búsqueda en dos pasos: presentando la Capa de Rastreo y Procesamiento e introduciendo una solicitud de búsqueda entrante a la Capa de Búsqueda y Orquestación.

La Capa de Rastreo y Procesamiento es la parte “en línea” del *framework*. Está en constante funcionamiento y es el responsable de la integración de nuevos datos en el *framework*. En el primer paso, las fuentes de datos de diferentes tipos se encuentran e integran en el nivel MDR (*Distribuited Metadata Repository* por sus siglas en inglés). El MDR es el punto de anclaje del *framework* IoTCrawler y contiene información de metadatos para todos los *data streams* disponibles en el *framework*. En el segundo paso, se aplica el modelo de información del IoTCrawler que está basado en el estándar NGSI-LD (*Next Generation Service Interface for Linked Data* por sus siglas en inglés) y centrado en el concepto de IotStreams. El modelo proporciona las bases para la información almacenada en los MDR y la integración de fuentes de datos heterogéneas. Después de la integración de nuevas fuentes de datos, el SE (*Semantic*

Enrichment) entra en juego para agregar nueva información a las nuevas fuentes de datos. El SE enriquece las fuentes de datos conocidas con nueva información extraída de los datos recibidos. El SE incluye un componente de análisis de calidad que agrega QoI, así como un PE (*Pattern Extractor*), que analiza los datos y proporciona información de nivel superior.

En paralelo, los datos enriquecidos se monitorean para habilitar las soluciones del *framework* de Detección de Fallas y Recuperación de Fallas. El componente de Monitorización asegura una experiencia de usuario constante al detectar *streams* defectuosos y proporcionar mecanismos de recuperación de datos. En el último paso dentro de esta capa, se crean índices de búsquedas lo que permite que las fuentes de datos se encuentren en el proceso de búsqueda de una manera más rápida (Iggena et al., 2021).

La capa de Búsqueda y Orquestación contiene componentes para manejar la búsqueda y solicitudes de suscripción provenientes de aplicaciones de IoT o usuarios individuales. El Orquestador es el principal punto de entrada para cualquier usuario o aplicación que desee buscar dispositivos IoT. Organiza el proceso de búsqueda y provee los *streams* de datos necesarios. El Orquestador convierte solicitudes GraphQL a solicitudes NGSI-LD y, por tanto, proporciona una interfaz fácil de usar, ocultando los mecanismos de consulta NGSI-LD. Para solicitudes de suscripción procedentes de aplicaciones de IoT, el Orquestador puede procesar la información recopilada y puede proporcionar un punto para recibir notificaciones sobre las propiedades de transmisión, por ejemplo, fallas detectadas. Las solicitudes NGSI-LD son redirigidas al componente de clasificación, que utiliza los índices, las restricciones dadas por el usuario y la información enriquecida para jerarquizar las fuentes de datos encontradas antes de que se envíen de vuelta al usuario o aplicación (Iggena et al., 2021).

El *framework* IoTCrawler (Skarmeta et al., 2018) proporciona un motor de búsqueda y rastreo para descubrir fuentes de *data streams* de IoT pertenecientes a varios dominios. El rastreador extrae metadatos de las fuentes de datos y los empuja a través de una Capa de Adaptación que anota semánticamente los metadatos como instancias de *IotStream*, y se almacena en un repositorio de metadatos RDF. Los metadatos capturados aquí incluyen una identificación para el *IotStream* descubierto y el *sosa: Sensor* que lo genera, y el *qu: QuantityKind* que se está midiendo y la *qu: Unit* utilizada para medir. La información de geolocalización si se proporciona tendrá el *geo: Point* conteniendo la posición absoluta y/o

la información de *iot-lite: relativeLocation*. El objetivo del motor de búsqueda es proporcionar a los Consumidores resultados que incluyen información de cómo llegar e interactuar el *data stream* de IoT *iot-lite: Service*. El sistema central adopta el *broker* de datos perteneciente al *framework* FIWARE (Cirillo et al., 2019), que emplea la API NGSI-LD. Esta API utiliza un meta-modelo para encapsular modelos de datos que son específicos de una plataforma en particular, en este caso, IoT-Stream. Los metadatos recopilados se distribuyen entre múltiples *brokers*, conocido como MDR. El *framework* emplea componentes de procesamiento que descubren y analizan *stream data* del MDR. Uno de los componentes, el módulo de Enriquecimiento Semántico, aplica análisis de datos para evaluar la *qoi: Quality*, se retroalimenta esta información al MDR y se adjunta al *IotStream* correspondiente. El módulo también aloja extractores de patrón que buscan por patrones en los *streams* de datos para un dominio específico, ya sean ciudades inteligentes, casas inteligentes, agricultura de precisión, etc. Ciertos patrones que se detectan en el *IotStream*, luego se traducirán en *Events*, que son luego empujados al MDR. Son estos *Events* los que servirán como palabras claves para que los Consumidores las utilicen para buscar el *data stream* de su interés entre todos los *data streams* y sensores. Para habilitar esto, los componentes relacionados con el motor de búsquedas indexarán y clasificarán los *IotStream*s, *sosa: Sensors, qu: QuantityKind* y *Events* basados en la localización y las métricas de Calidad de Información. Esto permitirá a los Consumidores realizar búsquedas instantáneas o suscribirse a actualizaciones de servicios de *data stream* basado en sus preferencias. La figura 6 muestra los componentes arquitectónicos en el *framework* IoTCrawler y las anotaciones de *streams* utilizando IoT-Stream (Elsaleh et al., 2020a).

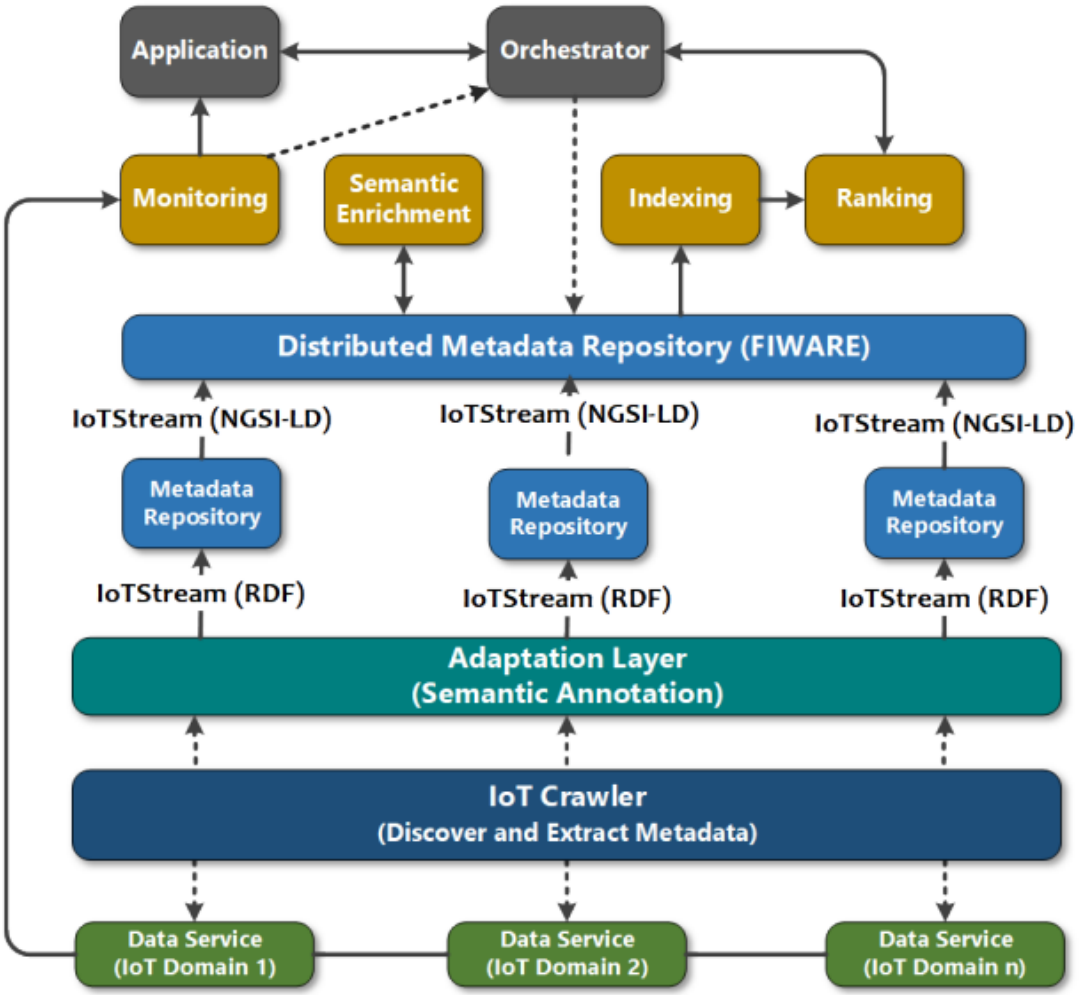


Figura 2.4. Arquitectura de IoTCrawler con el uso IoT-Stream.

2.9. Conclusiones Parciales del Capítulo.

La arquitectura propuesta responde a un modelo de cinco capas, basada en la Niebla ya que posee un *middleware* propietario de la plataforma que se encarga de realizar un preprocesamiento y filtrado de los datos. Se utilizan protocolos a diferentes niveles que permiten un correcto aprovechamiento del bajo consumo de los sensores y la rapidez que requiere una infraestructura de IoT como lo son IEEE 802.15.4, 6LoWPAN, RPL, UDP y MQTT-SN. Para monitorizar y controlar el sistema es utilizada la plataforma IoT Sofia2, la cual posee una versión *OpenSource* y utiliza semántica para manejar los datos de los sensores y otros dispositivos. La ontología encargada de manejar los datos en *streaming* sería IoT-Stream que al ser utiliza con herramientas para el análisis de los datos como el servicio KAT y un motor de búsquedas para IoT como el *framework* IoTCrawler proporciona los

mecanismos necesarios para extraer el conocimiento y lograr un desarrollo eficiente de los cultivos en la empresa agropecuaria.

CAPÍTULO 3. Aplicación y validación del modelo de streams.

Este capítulo estará dedicado a dar una detallada descripción de la ontología diseñada con sus clases, sus propiedades de datos y sus propiedades de los objetos, así como evaluar su desempeño mediante la ejecución de consultas SPARQL y reglas SQWRL a la misma. Los datos del experimento fueron obtenidos de la estación agrometeorológica ubicada en el municipio de Venezuela durante el mes de abril del 2020.

3.1. Modelo de información

El modelo de información principal se enfoca en modelar observaciones de los *streams*, su análisis y los eventos que se detectan de los mismos, así como actuar en consecuencia. La clase central de la cual las otras clases se enlazan directamente es *IotStream*. Esta abstracción representa un *data stream* originado de una fuente de datos de IoT. Posee propiedades que capturan el tiempo de vida del *stream* de datos de IoT que principalmente será utilizado como referencia en lugar de un consumo real por una aplicación. Estas propiedades de anotación son *streamStart* y *streamEnds*. Un *IotStream* puede ramificarse en otros *streams*, y por lo tanto *derivedFrom* otros *IotStreams*. Esto puede ser el resultado de algún tipo de procesamiento del *IotStream* del que ha sido derivado, como puede ser filtrado, remuestreo o agregación (Elsaleh et al., 2020).

La clase que será de mayor interés y consumo es el *StreamObservation* que pertenece al *IotStream*. El valor de una observación puede ser tomada de muchas formas. La primera es un dato fijo y la segunda puede ser un vector con un conjunto de datos. También contiene una marca de tiempo instantánea que mantiene un registro de cuándo la información fue capturada. Al considerar la reutilización de otras ontologías populares, la ontología SOSA proporciona una clase que cumple con los requisitos de capturar una observación de sensor que es la clase *sosa: Observation* y sus propiedades de datos, *sosa: hasSimpleResult* y *sosa: ResultTime*. Como uno de los principales objetivos de esta ontología es inhibir una manifestación ligera, la clase *sosa: Observation* se ha ampliado con una subclase - *StreamObservation*, para incluir directamente propiedades de datos para representar ventanas temporales. Estos se capturan en las propiedades de los datos *windowStart* y *windowEnd*, que

representan el inicio y el final de la ventana, respectivamente. Otra consideración importante para las *StreamObservations* es segregarlas del resto de los metadatos, ya que el número de las instancias creadas sería significativamente mayor en proporción con respecto al número de *IotStreams* y, por esta razón, la extensión de la instanciación de la clase *sosa: Observation* se mantiene intencionalmente al mínimo (Elsaleh et al., 2020b).

Las *StreamObservations* que *belongsTo IotStreams* pueden ser la salida de las lecturas del sensor o la salida de un proceso de análisis. En el caso de que los *IotStreams* sean *analysedBy* un proceso de análisis, la clase *Analytics* captura los métodos de las técnicas de análisis de datos aplicadas en *IotStream*. Puede ser un solo proceso o una cascada de procesos y, por tanto, se representa como un vector de cadenas con propiedades de datos *methods*, *parameters* y *paramValues*. La propiedad de datos *methods* captura los diferentes métodos y algoritmos con los que se ha analizado el flujo, esta propiedad de datos es un vector con los algoritmos analíticos aplicados al flujo. La propiedad de datos *parameters* establecidos para esos métodos también se capturan como una cadena de vectores, por lo que el primer elemento en el vector de métodos corresponde con el primer elemento del vector de parámetros. Vale la pena señalar que, por supuesto, los métodos pueden establecer múltiples parámetros, por lo que el elemento correspondiente en la propiedad de datos *parameters* puede ser una matriz de parámetros. Para cada parámetro, los valores que se establecen también se capturan en la propiedad de datos *paramValues*. Un proceso de análisis que se aplica a un *IotStream* posiblemente puede estar activo durante una ventana temporal con la vida útil de un *IotStream*. Por lo tanto, las propiedades de datos *windowStart* y *windowEnd* se utilizan para este caso. La clase *Analytics* también se puede utilizar exclusivamente para definir el proceso de análisis de datos que se utiliza para generar *Events* que se detectedFrom un *IotStream*, que sería aplicable en casos como clasificación o agrupamiento. El *Event* contiene propiedades que capturan la propiedad de datos *label* que es utilizado para describir el *Event*, y el intervalo temporal del *Event* también es relevante. Esta puede ser información útil para que los científicos de datos comprendan cómo el evento fue generado (Elsaleh et al., 2021).

La ontología geográfica del W3C proporciona un conjunto de conceptos básicos que representa la ubicación de una entidad. El principal concepto de interés es el *geo: Point* que contiene propiedades geoespaciales (latitud, longitud y altitud). La ontología de IoT-Lite

(Bermudez-Edo et al., 2017) extiende las propiedades para incluir la ubicación relativa y la altitud relativa. Para mantener el contexto histórico para las *stream observations*, especialmente en el caso de movilidad, un *geo: Point* puede ser enlazado a cada observación.

El siguiente subconjunto de conceptos adoptados se relaciona con la fuente generadora de *IotStream* y la medición de sus informaciones. Como los *streams* en el mundo real son generados por sensores, el concepto *Sensor* de SOSA (Janowicz et al., 2019) está vinculado. A través de las propiedades de objetos definidas por IoT-Lite, los conceptos *QuantityKind* y *Unit* de la ontología QU también son enlazados.

Aunque es el sensor quien genera el *stream* de IoT, a través de Internet, el *stream* de datos normalmente los proporciona un servicio de capa de aplicación TCP/IP. IoT-Lite proporciona la clase *Service* que contiene campos relacionados con la dirección del *endpoint* del servicio, el tipo de interfaz y el enlace a la descripción de la interfaz, que proporciona detalles sobre cómo interactuar con el servicio (Elsaleh et al., 2021).

Finalmente, a lo largo de la vida útil de un *stream* de IoT, la calidad de las observaciones del *stream* puede cambiar con el tiempo. Para el análisis de datos, el conocimiento de la calidad es muy importante para que las medidas de adaptación se puedan ir aplicando cuando sea necesario. La ontología *Quality of Information* (QoI) proporciona el concepto *qoi: Quality* que tiene subclases que se centran en un aspecto particular de la calidad, para los propósitos de esta investigación se ha tomado la subclase *Frecuency* la cual muestra la frecuencia con que se realizan las observaciones (Elsaleh et al., 2021).

Para la aplicación de la ontología en la Agricultura de Precisión es necesario agregarle axiomas que representen una granja con sus respectivos cultivos y productos elaborados por la misma, siendo de mayor interés los cultivos, los cuales tendrán una localización geográfica, representada por un *geo: Point*. Para optimizar el desarrollo de los cultivos se realizan observaciones sobre las condiciones meteorológicas, medioambientales y de tráfico para contemplar la necesidad del regado de los mismos. Este riego se puede realizar con vehículos aéreos y terrestres, los mismos cuentan a su vez con sensores para determinar su velocidad, aceleración y posición geográfica, lo cual también se encuentra contemplado en la ontología desarrollada.

En el **anexo I** se puede apreciar las relaciones entre las clases, las propiedades de los objetos y las propiedades de datos de cada clase. Así como el dominio al que deben pertenecer las mismas.

3.2. Estructura de la ontología

3.2.1. Clases

Los conceptos más básicos en un dominio deben corresponder a las clases que son las raíces de varios árboles taxonómicos. Cada individuo en el mundo OWL es un miembro de la clase owl:Thing. Por tanto, cada clase definida es implícitamente una subclase de owl:Thing. Esto se evidencia en la figura 3.1, donde además se puede apreciar la jerarquía de clases de la ontología.

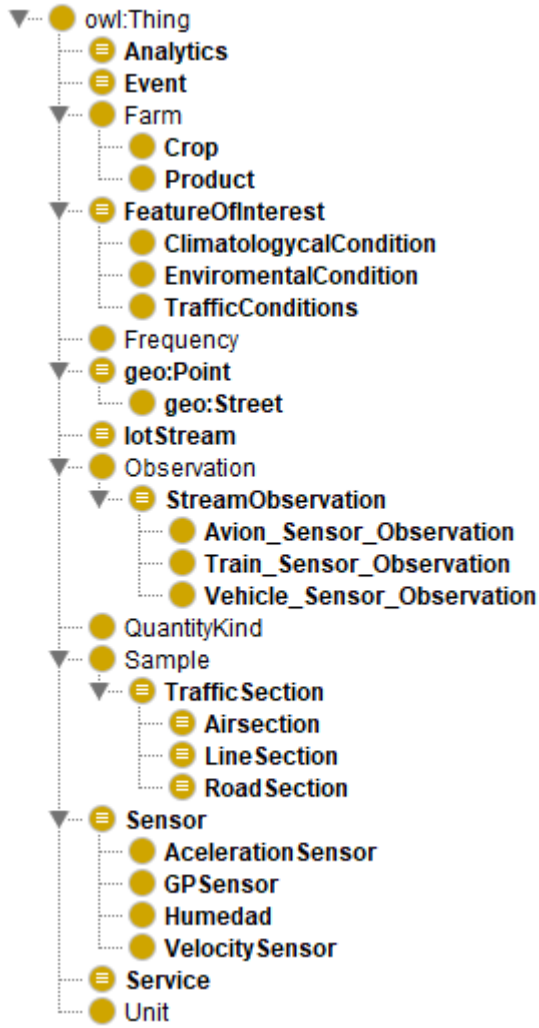


Figura 3.1: Jerarquía de Clases de la Ontología.

A continuación, se describen las clases principales que posee la ontología.

Analytics: Los análisis de datos que han sido aplicados al *stream* de datos de IoT. Esta clase ha sido heredada de Iot-Stream.

Event: El evento que ha sido detectado de un *data stream* de IoT y generado por un proceso analítico. Heredada de también de Iot-Stream.

Farm: Representa una granja.

Crop: Representa un cultivo de la granja.

Product: Es el producto obtenido a partir del cultivo.

FeatureOfInterest: La característica que esta siendo observada por un Sensor para arribar a un resultado. Por ejemplo, cuando se mide la altura de un árbol, 20m puede ser el resultado de la medición y el árbol es la *FeatureOfInterest*. Es una clase derivada de la ontología SOSA.

ClimatologycalCondition: Representa las condiciones climáticas de la zona.

EnvironmentalCondition: Representa las condiciones medioambientales del lugar donde se encuentra el cultivo.

TrafficConditions: Representa las condiciones del tráfico de una zona.

Frecuency: Máximo intervalo de tiempo entre 2 sets de datos. Como es un intervalo de tiempo debe ser mayor que 0. Es una clase de la ontología *Quality* utilizada para representar la calidad de la información de los *streams* de datos.

geo:Point: Un punto típicamente descrito utilizando un sistema de coordenadas relativas a la Tierra, como lo es WGS84. Únicamente es identificado por su longitud, latitud y altura.

geo:Street: Es una subclase de *geo:Point* que representa una calle.

IotStream: Una abstracción que representa un *data stream* de IoT que es generado por una fuente de IoT. Se deriva de la ontología Iot-Stream y es su concepto central, por lo cual las observaciones continuas de *streams* pertenecen a ella. Tiene un tiempo de vida definido por la hora de inicio del *stream* y la hora de finalización.

Frecuency: Define la frecuencia con la cual se actualiza la información.

QuantityKind: Se define como un aspecto común para cantidades mutuamente comparables. Representa la esencia de una cantidad sin ningún valor numérico o unidad.

Sample: Es uno de los conceptos núcleo en la ontología SOSA. Las muestras son artefactos de una estrategia observacional y no tienen una función significante fuera de su papel en el proceso de observación. Es una característica sobre la cual se pueden hacer observaciones, que pretende ser representativa de una *FeatureOfInterest* que no es completamente accesible.

TrafficSection: Se refiere a la sección del tráfico muestreada.

LineSection: Representa la sección de la línea muestreada.

AirSection: Representa la sección del espacio aéreo muestreado.

RoadSection: Representa la sección de la carretera a ser muestreada.

Sensor: Dispositivo, agente (incluidos los humanos), o *software* (simulación) envuelto en el proceso de medición. Los sensores responden a estímulos, por ejemplo, un cambio en el entorno y genera un resultado. Los sensores pueden ser montados en plataformas. Es uno de los conceptos núcleo de la ontología SOSA.

AcelerationSensor: Es una representación de un sensor de aceleración.

GPSensor: Representa un receptor de GPS (Sistema de Posicionamiento Global).

Humedad: Representa un sensor que es capaz de medir la humedad del aire.

VelocitySensor: Representa un sensor utilizado para medir la velocidad.

Service: Un servicio IoT provisto por un dispositivo IoT.

StreamObservation: Es heredada de la ontología IoT-Stream y es una observación hecha por un dispositivo de sensado capturado como un punto de datos durante un instante de tiempo, o como un subconjunto de puntos de datos durante un intervalo de tiempo definido.

Observation: Representa la observación realizada por un sensor.

Avion_Sensor_Observation: Representa la observación realizada por los sensores de un avión.

Train_Sensor_Observation: Es la observación que realizan los sensores de un tren.

Vehicle_Sensor_Observation: Representa la observación realizada por los sensores de un vehículo agrícola.

Unit: Representa la unidad de medida de la observación hecha por el sensor.

3.2.2. Propiedades de los objetos

Las principales propiedades de los objetos en la ontología se unen de la siguiente forma: *classes*/*individuals*. Ellos son los elementos más importantes en la ontología y actúan para gestionar el tráfico de los datos en tiempo real de los diferentes sensores. En la figura 3.2 se listan las propiedades de los objetos de la ontología.

En el **anexo II** se presenta una tabla más detallada con las propiedades de los objetos de la ontología, su dominio, su rango y su descripción.

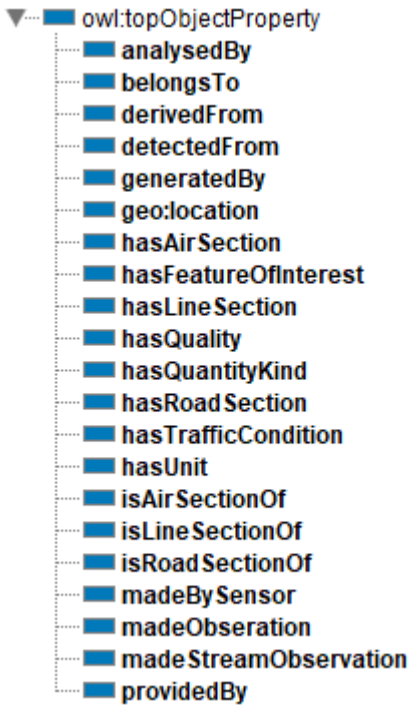


Figura 3.2: Propiedades de los objetos.

3.2.3. Propiedades de los datos

Las propiedades de los datos en la ontología representan todos los atributos de las clases con las que estos interactúan. Éstas se diseñan teniendo en cuenta las exigencias de los sistemas *Smart* y las propuestas de los diferentes lenguajes del consorcio de la web semántica. Son

atributos que se les define un tipo de valor y facilitan el análisis de las clases. En la figura 3.3 se muestran las propiedades de datos, a cada una de las mismas se le asigna un dominio que son las clases que poseen esa propiedad y el rango es el tipo de valor que pueden presentar los datos.

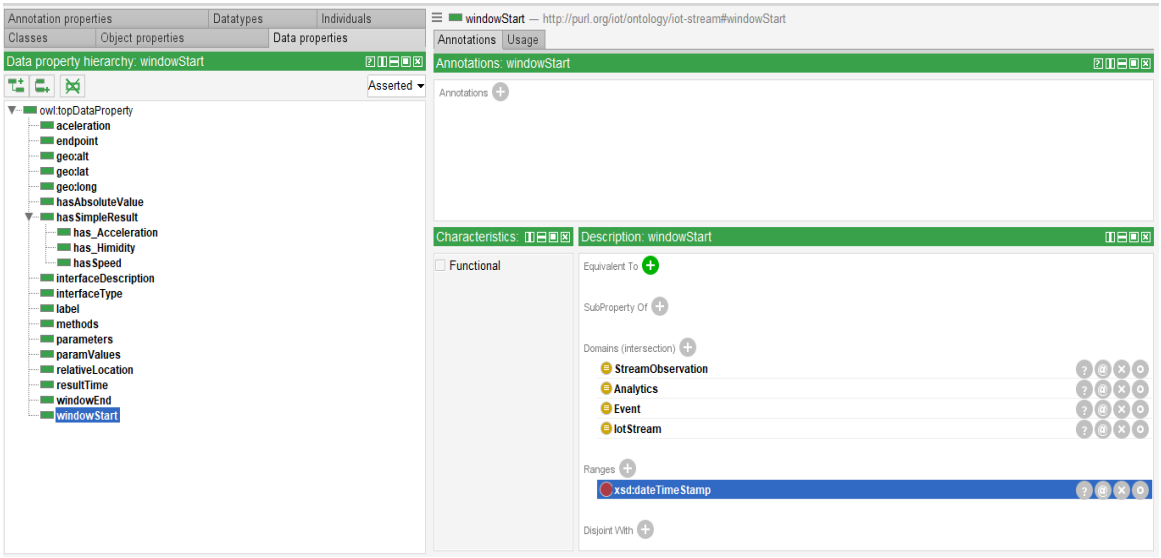


Figura 3.3: Propiedades de los datos de la Ontología.

3.3. Evaluación de la ontología mediante consultas SPARQL

3.3.1. SPARQL

Al lanzar modelo RDF se presentó el problema natural de consultar datos RDF. Desde entonces, se han propuesto varios diseños e implementaciones de lenguajes de consulta RDF. En 2004, el Grupo de Trabajo de Acceso a Datos RDF publicó un primer borrador de un artículo público de un lenguaje de consulta para RDF, llamado SPARQL. Desde entonces SPARQL se ha adoptado rápidamente como el estándar para consultar datos en la Web Semántica. En enero del 2008, SPARQL se convirtió en una recomendación del W3C (Pérez et al., 2009).

RDF es un formato de datos de grafos etiquetado y dirigidos para representar la información en la web y, por lo tanto, SPARQL es esencialmente un lenguaje de consulta de coincidencia de grafos. Las consultas SPARQL está compuestas por tres partes. La parte de coincidencia de patrones incluye varias características interesantes de patrones de grafos, como partes opcionales, unión de patrones, anidamiento, filtrado de valores de posibles coincidencias, y

la posibilidad de elegir la fuente de datos para que coincida con un patrón. Los modificadores de solución, una vez la salida del patrón ha sido calculada, permite modificar estos valores aplicando operadores clásicos como proyección, distinción, orden y límite. Finalmente, la salida de una consulta SPARQL puede ser diferentes tipos: preguntas de sí o no, selecciones de valores de las variables que coinciden con los patrones, construcción de nuevos datos RDF a partir de estos valores y descripciones de recursos (Pérez et al., 2009).

3.3.2. Consultas SPARQL

Este tipo de evaluación es para los usuarios finales y sustenta consultas que ningún motor de búsqueda de internet podría soportar en estos momentos.

Primeramente, se realiza una consulta para determinar los servicios de tipo *RESTful* que proveen los *IotStreams* generados por los sensores. Esto se logra utilizando la función FILTER en conjunto con una expresión regular, la letra i significa que en la búsqueda ignorará las mayúsculas y minúsculas. La respuesta puede ser observada en la figura 3.5.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX iot-lite: <http://purl.oclc.org/NET/UNIS/fiware/iot-lite#>
PREFIX iot-stream: <http://purl.org/iot/ontology/iot-stream#>
PREFIX sosa: <http://www.w3.org/ns/sosa/>
SELECT *
        WHERE { ?s ?p ?o . FILTER(regex(?o, "restful","i"))}
```

s	p	o
servicePrecipitacion	interfaceType	"RESTful"^^<http://www.w3.org/2001/XMLSchema#string>
servicePresion	interfaceType	"RESTful"^^<http://www.w3.org/2001/XMLSchema#string>
serviceTemperatura	interfaceType	"RESTful"^^<http://www.w3.org/2001/XMLSchema#string>
humService	interfaceType	"RESTful"^^<http://www.w3.org/2001/XMLSchema#string>

Figura 3.4: Consulta de los servicios tipo RESTful.

La siguiente consulta se interceptan 4 variables que son cada *IotStream* con el *Sensor* que lo genera y sus propiedades de *windowStart* y *windowEnd* de esta manera se observa la duración de cada uno como se aprecia en la figura 3.6.

Es necesario agregar que esta consulta genera con extrema precisión los elementos necesarios para que un usuario, desde su teléfono móvil, pueda acceder a los datos del sistema. Los valores obtenidos permitirían en un futuro realizar estudios de tendencia mediante otras aplicaciones de la plataforma IoT y el tráfico de datos a los gestores de la agricultura, puesto que, desde buscadores tradicionales esto no podría lograrse de ninguna manera.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX iot-lite: <http://purl.oclc.org/NET/UNIS/fiware/iot-lite#>
PREFIX iot-stream: <http://purl.org/iot/ontology/iot-stream#>
PREFIX sosa: <http://www.w3.org/ns/sosa/>
SELECT ?IotStream ?Sensor ?windowStart ?windowEnd
        WHERE { ?IotStream iot-stream:windowStart ?windowStart .  ?IotStream iot-stream:windowEnd ?windowEnd . ?IotStream iot-stream:generatedBy ?Sensor}
```

IotStream	Sensor	windowStart	windowEnd
hum-raw-04	1IEE1000	"2020-04-12T14:20:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:30:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
presion-raw-01	Barometro	"2020-04-12T13:20:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T13:30:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
presion-raw-05	Barometro	"2020-04-12T14:00:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:10:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
precip-raw-05	Pluviometro	"2020-04-12T14:40:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:50:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
hum-raw-02	1IEE1000	"2020-04-12T13:40:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T13:50:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
presion-raw-03	Barometro	"2020-04-12T13:40:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T13:50:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
precip-raw-03	Pluviometro	"2020-04-12T14:20:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:30:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
temp-raw-01	Termometro	"2020-04-12T14:00:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:10:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
viento-raw-03	Anemometro	"2020-04-12T13:50:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T13:40:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
temp-raw-03	Termometro	"2020-04-12T14:20:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:30:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
viento-raw-05	Anemometro	"2020-04-12T14:00:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:10:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>
precip-raw-01	Pluviometro	"2020-04-12T14:00:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>	"2020-04-12T14:10:00Z"^^<http://www.w3.org/2001/XMLSchema#dateTimeStamp>

Figura 3.5: Consulta de los IotStream, Sensores, WindowStart y WindowEnd.

En la figura 3.7 se observa la consulta en la cual encuesta todas las observaciones en las cuales las precipitaciones hayan sido menores a 30mm con la asociación del sensor que realizó la medición. Esto permite analizar si el comportamiento de las precipitaciones durante el desarrollo del cultivo fue favorable o fue necesario un regado constante para mantener la humedad del producto. Para obtener este resultado se realizó una restricción de los sensores a fin de que cada uno este asociado a una o más observaciones, lo que garantiza la homogeneidad en el diseño semántico y mejora la cardinalidad computacional.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX iot-lite: <http://purl.oclc.org/NET/UNIS/fiware/iot-lite#>
PREFIX iot-stream: <http://purl.org/iot/ontology/iot-stream#>
PREFIX sosa: <http://www.w3.org/ns/sosa/>

SELECT ?StreamObservation ?Sensor ?hasSimpleResult
        WHERE {
                FILTER(?hasSimpleResult <30)

                ?StreamObservation sosa:hasSimpleResult ?hasSimpleResult .  ?StreamObservation sosa:madeBySensor ?Sensor
}
```

StreamObservation	Sensor	hasSimpleResult
precip-stream-05	Pluviometro	"20"^^<http://www.w3.org/2001/XMLSchema#int>
precip-stream-03	Pluviometro	"20"^^<http://www.w3.org/2001/XMLSchema#int>
precip-stream-04	Pluviometro	"20"^^<http://www.w3.org/2001/XMLSchema#int>

Figura 3.6: Consulta de las precipitaciones menores a 30mm.

3.3.3. Uso de Geolocalización

Todos los sistemas relacionados al IoT tienen varios elementos comunes, uno de ellos es la geolocalización. En la figura 3.7 se puede observar el resultado de relacionar cada cultivo y los sensores de la estación agrometeorológica con su localización, estos valores son leídos por un script de Python y mediante la librería Folium se representan todos los cultivos en un

mapa quedando los marcadores como se muestran en la figura 3.8. De esta forma se puede visualizar de manera sencilla la distribución de los cultivos en un área determinada. En el anexo III se muestra el código de Python utilizado para generar este mapa.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX iot-lite: <http://purl.oclc.org/NET/UNIS/fiware/iot-lite#>
PREFIX iot-stream: <http://purl.org/iot/ontology/iot-stream#>
PREFIX sosa: <http://www.w3.org/ns/sosa/>
PREFIX wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT ?Crop ?lat ?long
        WHERE
{
        ?Crop wgs84_pos:location ?Point . ?Point  wgs84_pos:lat ?lat  . ?Point  wgs84_pos:long ?long
}
```

Crop	lat	long
Calabaza	"21.831601"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.965899"^^<http://www.w3.org/2001/XMLSchema#string>
temp-raw-05	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
hum-raw-05	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
presion-raw-03	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
viento-raw-01	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
presion-raw-05	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
viento-raw-03	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
precip-raw-01	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
viento-raw-05	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
temp-raw-02	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
hum-raw-01	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>
precip-raw-03	"21.864042"^^<http://www.w3.org/2001/XMLSchema#string>	"-78.931615"^^<http://www.w3.org/2001/XMLSchema#string>

Figura 3.8: Ubicación geográfica de los cultivos.

Figura 3.8: Ubicación geográfica de los cultivos.

3.4. Validación mediante reglas SQWRL

3.4.1. SQWRL

La habilidad de extraer información de ontologías OWL es un requerimiento básico. Mientras SPARQL y sus extensiones son utilizados como un lenguaje de consulta de OWL es en el mejor de los casos incompleto. Por lo que se presenta una necesidad de un lenguaje de consulta de OWL conciso, legible y semánticamente robusto. SQWRL (*Semantic Query-enhanced Web Rule Language*) se basa en el lenguaje de reglas SWRL. SQWRL toma un antecedente de regla SWRL estándar y lo utiliza de manera efectiva como una especificación de patrón para una consulta. Reemplaza la regla consecuente con una especificación de recuperación. SQWRL las funciones incorporadas de SWRL como punto de extensión. Usando las características incorporadas, define un conjunto de operadores que se pueden usar para construir especificaciones de recuperación. El atractivo de este enfoque es que no se requieren extensiones sintácticas para SWRL. Por lo tanto, los editores SWRL existentes se pueden utilizar para generar y editar consultas SQWRL. Además, se pueden utilizar mecanismos de serialización SWRL estándar para que las consultas se puedan almacenar en ontologías OWL (O'Connor and Das, 2009).

3.4.2. Consultas con reglas SQWRL

Para la primera regla se hace una consulta que muestre todos los *IotStreams* de la base de datos con sus respectivas hora y fecha de inicio y final, pero con la particularidad de que los mismos se encuentran ordenados cronológicamente desde el más actual hasta el más antiguo. En la figura 3.9 se muestran los resultados de la consulta.

La regla se enuncia de la siguiente manera:

iot-stream:IotStream(?s) ^ iot-stream:windowStart(?s, ?st) ^ iot-stream:windowEnd(?s, ?e) -> sqwrl:select(?s, ?st, ?e) ^ sqwrl:orderByDescending(?st, ?e)

s	st	e
:hum-sum-05	"2020-04-12T16:00:00Z"^^xsd:dateTimeStamp	"2020-04-12T16:10:00Z"^^xsd:dateTimeStamp
:hum-sum-04	"2020-04-12T15:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:50:00Z"^^xsd:dateTimeStamp
:eventSequia	"2020-04-12T15:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:50:00Z"^^xsd:dateTimeStamp
:hum-sum-03	"2020-04-12T15:20:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:30:00Z"^^xsd:dateTimeStamp
:analytics-05	"2020-04-12T15:20:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:30:00Z"^^xsd:dateTimeStamp
:analytics-04	"2020-04-12T15:10:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:20:00Z"^^xsd:dateTimeStamp
:hum-sum-02	"2020-04-12T15:00:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:10:00Z"^^xsd:dateTimeStamp
:analytics-03	"2020-04-12T15:00:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:10:00Z"^^xsd:dateTimeStamp
:analytics-02	"2020-04-12T15:00:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:temp-sum-05	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:temp-raw-05	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:presion-sum-05	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:precip-sum-05	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:precip-raw-05	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp

Figura 3.9: Resultados consulta ordenar IotStreams.

Para la siguiente consulta se requiere conocer cuándo la humedad de la zona ha sido menor del 60%, para lo cual se buscan todos las *StreamObservations* que hayan tenido un valor de *hasSimpleResult* menor de 60 y que hayan sido realizadas por el sensor de humedad y se muestra además la hora en la que mostraron ese valor, obteniéndose como resultado la figura 3.10. Esto se logra mediante la siguiente regla:

iot-stream:StreamObservation(?s) ^ sosa:madeBySensor(?s, autogen0:IEE1000) ^ sosa:hasSimpleResult(?s, ?t) ^ swrlb:lessThan(?t, 60) ^ sosa:resultTime(?s, ?r) -> sqwrl:select(?s, ?t, ?r)

s	t	r
:hum-stream-03	"50"^^xsd:int	"2020-04-12T13:40:00Z"^^xsd:dateTimeStamp
:hum-stream-02	"30"^^xsd:int	"2020-04-12T13:30:00Z"^^xsd:dateTimeStamp

Figura 3.10: Resultados de la consulta de los valores humedad menores de 60%.

Otra regla que se puede aplicar para el análisis en tiempo real de las variables climatológicas es la visualización de los eventos meteorológicos ocurridos en un período de tiempo determinado. Esto se logra mediante la regla:

iot-stream:Event(?s) ^ iot-stream:detectedFrom(?s, ?r) ^ iot-stream:windowStart(?s, ?st) ^ iot-stream:windowEnd(?s, ?e) -> sqwrl:select(?s, ?r, ?st, ?e)

Esto toma todos los eventos climáticos y los relaciona con el *IotStream* resultado de un análisis que dispara la ocurrencia del evento y muestra, además, el tiempo de duración del mismo. Los resultados de esta búsqueda se muestran en la figura 3.11.

s	r	st	e
:eventTormentaTropical	:precip-sum-01	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:40:00Z"^^xsd:dateTimeStamp
:eventOlaDeCalor	:temp-sum-01	"2020-04-12T14:00:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:30:00Z"^^xsd:dateTimeStamp
:eventHumedadAlta	:hum-sum-01	"2020-04-12T14:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:50:00Z"^^xsd:dateTimeStamp
:eventHuracan	:presion-sum-01	"2020-04-12T13:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T14:00:00Z"^^xsd:dateTimeStamp
:eventSequia	:hum-sum-02	"2020-04-12T15:40:00Z"^^xsd:dateTimeStamp	"2020-04-12T15:50:00Z"^^xsd:dateTimeStamp

Figura 3.11: Resultados de la búsqueda de eventos climatológicos.

Mediante el agrupamiento se puede discernir de forma rápida y sencilla la cantidad de observaciones que realiza cada sensor y cuáles son estas observaciones. La regla siguiente agrupa a todos los sensores y muestra la cantidad de *StreamObservations* que realizó el mismo, lo que lleva como resultado lo mostrado en la figura 3.12.

sosa:Sensor(?p) ^ iot-stream:madeStreamObservation(?p, ?d) . sqwrl:makeSet(?st, ?d) ^ sqwrl:groupBy(?st, ?p) . sqwrl:size(?n, ?st) -> sqwrl:select(?p, ?n)

p	n
:Anemometro	5
:Barometro	5
:Pluviometro	5
:Termometro	5
autogen0:IEE1000	5

Figura 3.12: Número de observaciones realizadas por cada sensor.

Por último, se creó una regla que agrupa todas las observaciones realizadas por cada sensor de la estación agrometeorológica y sus resultados se muestran en la figura 3.13.

sosa:Sensor(?s) ^ iot-stream:madeStreamObservation(?s, ?r) . sqwrl:makeSet(?st, ?r) ^ sqwrl:groupBy(?st, ?s) -> sqwrl:select(?s, ?r)

s	r
:Anemometro	:viento-stream-01
:Anemometro	:viento-stream-02
:Anemometro	:viento-stream-03
:Anemometro	:viento-stream-04
:Anemometro	:viento-stream-05
:Barometro	:presion-stream-01
:Barometro	:presion-stream-02
:Barometro	:presion-stream-03
:Barometro	:presion-stream-04
:Barometro	:presion-stream-05
:Pluviometro	:precip-stream-01
:Pluviometro	:precip-stream-02
:Pluviometro	:precip-stream-03
:Pluviometro	:precip-stream-04

Figura 3.13: Relación entre cada sensor con sus observaciones.

Los procesos de evaluación por reglas demuestran la calidad de la ontología y sus consultas inteligentes a los *streams*. Uno de los elementos más favorables de este tipo de proceder, es que se constata que la ontología describe semánticamente la gran cantidad de *streams* generados continuamente y responde a las consultas con gran rapidez y precisión. Finalmente se entrega el .csv para que el personal encargado tome las decisiones permitentes en cuanto al desarrollo del cultivo.

En la figura 3.14 se muestra un segmento en Excel del dataset utilizado para la realización de las pruebas de la ontología. El mismo posee los datos de lectura de los sensores de la estación meteorológica ubicada en el municipio de Venezuela en Ciego de Ávila en el mes de abril del año 2020.

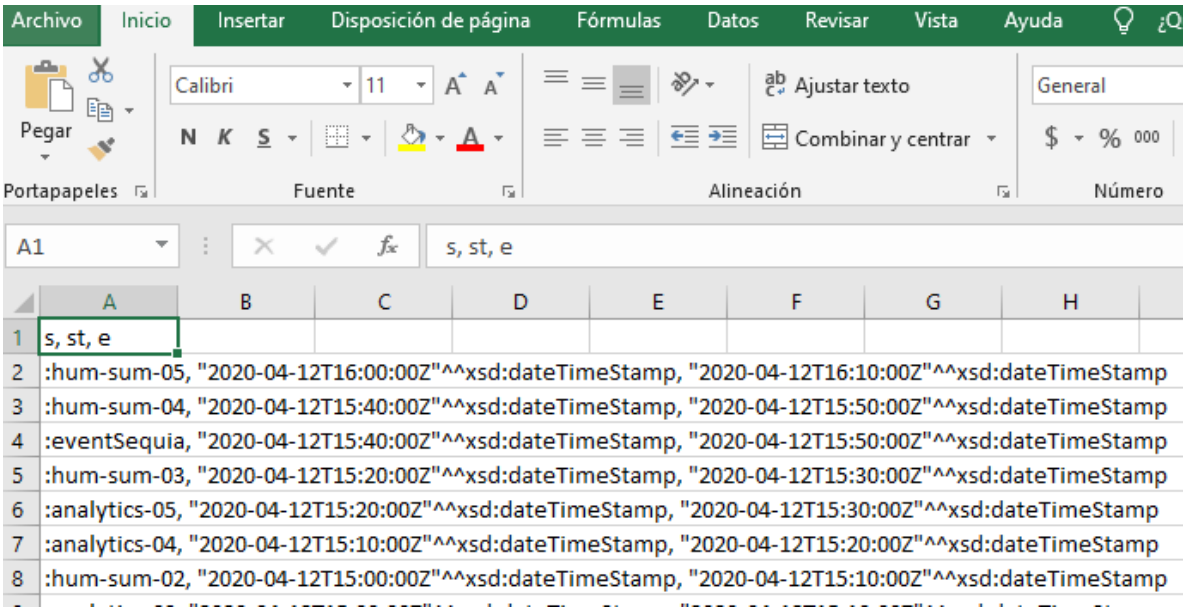


Figura 3.14: Dataset en Excel para la ejecución de pruebas.

3.5. Análisis Económico y Medioambiental

El modelo diseñado tiene una gran repercusión en el ahorro de los recursos utilizados en la agricultura debido a que se tiene acceso en tiempo real a toda la información brindada por los sensores. Todos estos datos están enriquecidos para que su análisis y clasificación ocurra de manera inmediata, además de ser este un modelo muy ligero que no supone ninguna demora en su transmisión. Por otra parte, tiene un impacto ambiental considerable, ya que contribuye al ahorro de un recurso natural fundamental, el agua, disminuyendo el estrés del manto freático que conlleva el proceso de irrigación, y reduciendo la erosión de los suelos cultivables.

Con toda la información climática a disposición de los agricultores y de las máquinas de regado en tiempo real se permite una toma de decisiones que logre que el cultivo se desarrolle de manera eficaz, trayendo consigo un aumento de la producción y un ahorro de recursos hidráulicos.

3.6. Conclusiones Parciales del Capitulo

La ontología desarrollada es una ontología ligera cuya clase principal es *IotStream*, las demás clases se enlazan a la misma mediante las propiedades de objeto. Cada clase posee sus propias propiedades de datos lo cual le permite describir sus características. El modelo se encarga de

describir los *streams* de datos, analizarlos y desencadenar eventos y reaccionar de alguna de forma a los mismos.

Se realizaron distintos experimentos para demostrar la eficacia de la misma mediante consultas SPARQL, el cual es un lenguaje de consulta que permite búsquedas que ningún motor de búsqueda es capaz de realizar. Otra forma de búsqueda utilizada fueron las consultas SQWRL, el cual permite una búsqueda más elaborada y precisa, así como la creación de grupos de datos.

CONCLUSIONES

- Iot-Stream es un modelo semántico para las anotaciones de *streams* de datos que facilita la implementación de aplicaciones de IoT que lidian datos de sensores en tiempo real. Con el concepto principal, *StreamObservation*, con propiedades simples de valores y tiempo, se logra una aceleración de las consultas al modelo de *streams* y se creó el resto de los conceptos necesarios para la búsqueda e indexación alrededor de este concepto principal. Al hacer esto, se mejora el tiempo de procesamiento de las consultas a los *streams*.
- El modelo diseñado ha sido desarrollado acorde a las guías más reconocidas para desarrollar modelos semánticos y, especialmente, las guías para entornos IoT, donde la escalabilidad y los cortos tiempos de procesamiento son esenciales. Con estas restricciones se ha creado un modelo semántico ligero totalmente compatible con el con la gran conocida ontología SSN y su reciente núcleo ligero SOSA.
- Las ontologías necesitan herramientas y guías para utilizarlas efectivamente y facilitar la adopción del modelo de datos. Para esta ontología se ha creado una arquitectura de sistema con las entidades necesarias para anotar y consumir los datos en *stream*. Esta arquitectura facilitará la adopción y replicación de la propuesta. Se utilizan protocolos a diferentes niveles que permiten un correcto aprovechamiento del bajo consumo de los sensores y la rapidez que requiere una infraestructura de IoT. Para el control y monitorización del sistema se utiliza la plataforma Sofia2 en su versión básica, la cual es *OpenSource*. Los procesos de análisis expuestos como un servicio web se han ido incrementando por lo cual la arquitectura del sistema ha sido basada en esos servicios, incorporando la anotación y consumo de los datos en *stream*.
- La ontología y su sistema de integración ha quedado validada por diversos experimentos con reglas y modelos de integración, lo que garantiza la calidad del modelo de *streams* para ser usado en la práctica.

RECOMENDACIONES

- Aplicar este modelo ontológico a los datos generados por los sensores físicos del sector de la agricultura en Cuba.
- Mostrar este proyecto a las autoridades competentes a fin de normalizar los procesos de manejo de datos para sentar las bases para la automatización de los procesos de la agricultura.
- Continuar desarrollando los procesos de investigación en IoT relacionados con las ontologías y los sensores como vía de automatización de la agricultura en Cuba.

REFERENCIAS BIBLIOGRÁFICAS

Agencia Cubana de Noticias, 2020. Empresa agropecuaria Cubasoy, decisiva en el aporte de alimentos - Televisión Avileña [WWW Document]. URL https://www.tvavila.icrt.cu/empresa-agropecuaria-cubasoy-decisiva-en-el-aporte-de-alimentos/ (accessed 11.9.21).

Agtech, 2017. Cómo la Big Data está revolucionando la agricultura y la cadena de abastecimiento [WWW Document]. Redagrícola Chile. URL https://www.redagricola.com/cl/la-big-data-esta-revolucionando-la-agricultura-la-cadena-abastecimiento-2/ (accessed 4.23.21).

Ahrabian, A., Kolozali, S., Enshaeifar, S., Cheong-Took, C., Barnaghi, P., 2017. Data analysis as a web service: A case study using IoT sensor data, in: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Presented at the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 6000–6004. https://doi.org/10.1109/ICASSP.2017.7953308

Barber, E.E., Mostaccio, M.R., de Pedro, G., Blanco, N., Romagnoli, S., Gregui, C., Pisano, S., 2018. Metodologías para el diseño de ontologías Web. Inf. Cult. Soc.

Barnaghi, P., Wang, W., Dong, L., Wang, C., n.d. A Linked-Data Model for Semantic Sensor Streams.

Bermudez-Edo, M., Elsaleh, T., Barnaghi, P., Taylor, K., 2017. IoT-Lite: a lightweight semantic model for the internet of things and its use with dynamic semantics. Pers. Ubiquitous Comput. 21, 475–487. https://doi.org/10.1007/s00779-017-1010-8

Berryman, A.A., 2020. La revolución de los datos en la agricultura de precisión: aplicaciones en cultivos de vid [WWW Document]. Redagrícola Chile. URL https://www.redagricola.com/cl/la-revolucion-de-los-datos-en-la-agricultura-de-precision-aplicaciones-en-cultivos-de-vid/ (accessed 4.23.21).

Callís Flaqué, E., 2019. Evaluación de las prestaciones de la aplicación Matlab Mobile para su uso en la agricultura de precisión (Thesis). Universidad Central “Marta Abreu” de Las Villas, Facultad de Ingeniería Eléctrica, Departamento de Automática y Sistemas Computacionales.

Cárdenas Rivero, A.J., 2019. Diseño de una aplicación IoT para casas de cultivo (Thesis). Universidad Central “Marta Abreu” de Las Villas, Facultad de Ingeniería Eléctrica, Departamento de Automática y Sistemas Computacionales.

Cebrián, V., Manuel, V., 2017. Interconexión de dispositivos IoT (Internet of Things) con plataforma Sofia2 (Proyecto/Trabajo fin de carrera/grado). Universitat Politècnica de València.

Cirillo, F., Solmaz, G., Berz, E.L., Bauer, M., Cheng, B., Kovacs, E., 2019. A Standard-Based Open Source IoT Platform: FIWARE. IEEE Internet Things Mag. 2, 12–18. https://doi.org/10.1109/IOTM.0001.1800022

Codina, L., Rovira, C., 2006. La Web semántica, in: Tramullas, J. (Ed.), Tendencias En Documentación Digital. Trea.

Compton, M., Barnaghi, P., Bermudez, L., Garcia-Castro, R., 2012. The SSN ontology of the W3C semantic sensor network incubator group. J. Web Semant. 17, 25–32. https://doi.org/10.1016/j.websem.2012.05.003

Elsaleh, T., Bermudez-Edo, M., Enshaeifar, S., Acton, S.T., Rezvani, R., Barnaghi, P., 2021. IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams.

Elsaleh, T., Bermudez-Edo, M., Enshaeifar, S., Acton, S.T., Rezvani, R., Barnaghi, P., n.d. IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams.

Elsaleh, T., Enshaeifar, S., Rezvani, R., Acton, S.T., Janeiko, V., Bermudez-Edo, M., 2020a. IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams and Its Use with Data Analytics and Event Detection Services. Sensors 20, 953. https://doi.org/10.3390/s20040953

Elsaleh, T., Enshaeifar, S., Rezvani, R., Acton, S.T., Janeiko, V., Bermudez-Edo, M., 2020b. IoT-Stream: A Lightweight Ontology for Internet of Things Data Streams and Its Use with Data Analytics and Event Detection Services. Sensors 20, 953. https://doi.org/10.3390/s20040953

Estevez Pérez, A.A., 2018. Diseño de red de sensores inalámbricos para aplicación de internet de las cosas en casas de cultivos (Thesis). Universidad Central “Marta Abreu” de Las Villas, Facultad de Ingeniería Eléctrica, Departamento de Automática y Sistemas Computacionales.

Evans, D., 2011. Internet de las cosas. Cómo Próxima Evol. Internet Lo Cambia Todo Cisco Internet Bussiness Solut. Group-IBSG 11, 4–11.

Ganzha, M., Paprzycki, M., Paw\lowski, W., Szmeja, P., Wasielewska, K., 2017. Streaming semantic translations, in: 2017 21st International Conference on System Theory, Control and Computing (ICSTCC). IEEE, pp. 1–8.

García, E., Flego, F., 2008. Agricultura de precisión. Rev. Cienc. Tecnol. Recuperado Httpwww Palermo Eduingenier IaC Ienciayt Ecnolog Iacienciaytecno8 Html.

Garcia-Sanchez, A.-J., 2011. Wireless sensor network deployment for integrating video-surveillance and data-monitoring in precision agriculture over distributed crops. Comput. Electron. Agric. 75, 288–303. https://doi.org/10.1016/j.compag.2010.12.005

Guarino, N., 1998. Formal Ontology in Information Systems: Proceedings of the First International Conference (FOIS’98), June 6-8, Trento, Italy. IOS Press.

Gutiérrez Junco, D.F., Díaz Caro, D.F., Cruz Forero, M.S., Ruge, I.A.R., 2015. Agrometeorological monitoring station based microcontroller and bluetooth communication, in: 2015 IEEE 2nd Colombian Conference on Automatic Control

(CCAC). Presented at the 2015 IEEE 2nd Colombian Conference on Automatic Control (CCAC), pp. 1–5. https://doi.org/10.1109/CCAC.2015.7345228

Iggena, T., Bin Ilyas, E., Fischer, M., Tönjes, R., Elsaleh, T., Rezvani, R., Pourshahrokhi, N., Bischof, S., Fernbach, A., Xavier Parreira, J., Schneider, P., Smirnov, P., Strohbach, M., Truong, H., González-Vidal, A., Skarmeta, A.F., Singh, P., Beliatis, M.J., Presser, M., Martinez, J.A., Gonzalez-Gil, P., Krogbæk, M., Holmgård Christophersen, S., 2021. IoTCrawler: Challenges and Solutions for Searching the Internet of Things. Sensors 21, 1559. https://doi.org/10.3390/s21051559

Janowicz, K., Haller, A., Cox, S.J.D., 2019. SOSA: A lightweight ontology for sensors, observations, samples, and actuators - ScienceDirect.

Janowicz, K., Haller, A., Cox, S.J.D., n.d. SOSA: A lightweight ontology for sensors, observations, samples, and actuators - ScienceDirect.

Khan, F.A., Abubakar, A., Mahmoud, M., Al-Khasawneh, M.A., Alarood, A.A., 2019. Cotton crop cultivation oriented semantic framework based on IoT smart farming application. Int. J. Eng. Adv. Technol. 8, 480–484.

Kolozali, S., Bermudez-Edo, M., Puschmann, D., Ganz, F., Barnaghi, P., 2014. A Knowledge-Based Approach for Real-Time IoT Data Stream Annotation and Processing. Presented at the IEEE Inernational Conference on Internet of Things, University of Surrey, Guildford, Surrey, United Kingdom.

Le Phuoc, D., Hauswirth, M., 2019. Semantic Stream Processing.

Llanes, K.R., Casanova, M.A., Lemus, N.M., 2016. From Sensor Data Streams to Linked Streaming Data: a survey of main approaches. J. Inf. Data Manag. 7, 130–130.

MARQUÉS, M.P., 2015. BIG DATA - Técnicas, herramientas y aplicaciones. Alfaomega Grupo Editor.

Maury Yera, L., 2015. Estándares para la realización práctica de la Internet de las Cosas (Thesis). Universidad Central “Marta Abreu” de Las Villas.

Morales Machado, R., 2016. Sistemas inteligentes de irrigación en la agricultura (Thesis). Universidad Central “Marta Abreu” de Las Villas. Facultad de Ingeniería Eléctrica. Departamento de Automática y Sistemas Computacionales.

Navarro, E., Costa, N., Pereira, A., 2020. A Systematic Review of IoT Solutions for Smart Farming. Sensors 20, 4231. https://doi.org/10.3390/s20154231

Nuñez-Agurto, D., Benavides-Astudillo, E., Rodríguez, G., Salazar, D., 2020. Propuesta de una Plataforma de Bajo Costo Basada en Internet de las Cosas para Agricultura Inteligente. Cumbres 6, 53–66. https://doi.org/10.48190/cumbres.v6n1a5

O’Connor, M.J., Das, A.K., 2009. SQWRL: a query language for OWL., in: OWLED.

Otero Barrera, G., 2019. Sistemas de administración para la agricultura de precisión (Thesis). Universidad Central “Marta Abreu” de Las Villas, Facultad de Ingeniería Eléctrica, Departamento de Automática y Sistemas Computacionales.

Paolo, n.d. Applied Data Science and Engineering, with a topping of Data Provenance | Paolo Missier, School of Computing. URL https://blogs.ncl.ac.uk/paolomissier/ (accessed 5.6.21).

Pérez, J., Arenas, M., Gutierrez, C., 2009. Semantics and complexity of SPARQL. ACM Trans. Database Syst. TODS 34, 1–45.

Pinto, F. de A. de C., Queiroz, D.M. de, Chartuni, E., Ruz, E., 2007. Agricultura de precisión : nuevas herramientas para mejorar la gestión tecnológica en la empresa agropecuaria. Rev. Palmas 28, 29–34.

Pozo Pérez, J.R., 2017. Diseño de un sistema automatizado para riego por goteo en la empresa agroindustrial Victoria de Girón (Thesis). Universidad Central “Marta Abreu” de Las Villas. Facultad de Ingeniería Eléctrica. Departamento de Automática y Sistemas Computacionales.

Raimond, Y., 2008. A distributed music information system (Ph.D.). Queen Mary, University of London.

Rodríguez Rodríguez, R., Izaguirre Castellanos, E., Pérez, P., Rubén, J., 2017. Diseño de sistema automatizado para riego por goteo empleando red inalámbrica de sensores de humedad y estación agrometeorológica.

Rogushina, J., Gladun, A., Прийма, С.М., Прийма, С.Н., Pryima, S., 2018. Use of Ontologies for Metadata Records Analysis in Big Data.

Sánchez, A.F.A., 2019. LA AGRICULTURA DE PRECISIÓN EN LA EMPRESA AGRARIA DE CUBA. Rev. Direito Paz 1, 257–285. https://doi.org/10.32713/rdp.v1i40.1133

Sethi, P., Sarangi, S.R., 2017. Internet of Things: Architectures, Protocols, and Applications. J. Electr. Comput. Eng. 2017, e9324035. https://doi.org/10.1155/2017/9324035

Silva, F. da, Cristina, A., Hirmer, P., 2020. Models for Internet of Things Environments—A Survey. Information 11, 487. https://doi.org/10.3390/info11100487

Skarmeta, A.F., Santa, J., Martínez, J.A., Parreira, J.X., Barnaghi, P., Enshaeifar, S., Beliatis, M.J., Presser, M.A., Iggena, T., Fischer, M., Tönjes, R., Strohbach, M., Sforzin, A., Truong, H., 2018. IoTCrawler: Browsing the Internet of Things, in: 2018 Global Internet of Things Summit (GIoTS). Presented at the 2018 Global Internet of Things Summit (GIoTS), pp. 1–6. https://doi.org/10.1109/GIOTS.2018.8534528

Smith, B., Kusnierczyk, W., Schober, D., Ceusters, W., 2006. Towards a coherent terminology for principles-based ontology, in: Proceedings of the 2nd International Workshop on Formal Biomedical Knowledge Representation, KRMED. pp. 54–70.

Soldatos, J., 2016. Building Blocks for IoT Analytics. River Publishers.

Stuart, D., 2016. Practical Ontologies for Information Professionals. Facet Publishing.

Tran, B., 2020. Development of an Internet of Things platform for Smart Agriculture solutions.

Valdés Valle Alberto, 2020. Diseño de un sensor de datos para una estación agrometeorológica usando Semantic Sensor Network (Thesis). Universidad Central

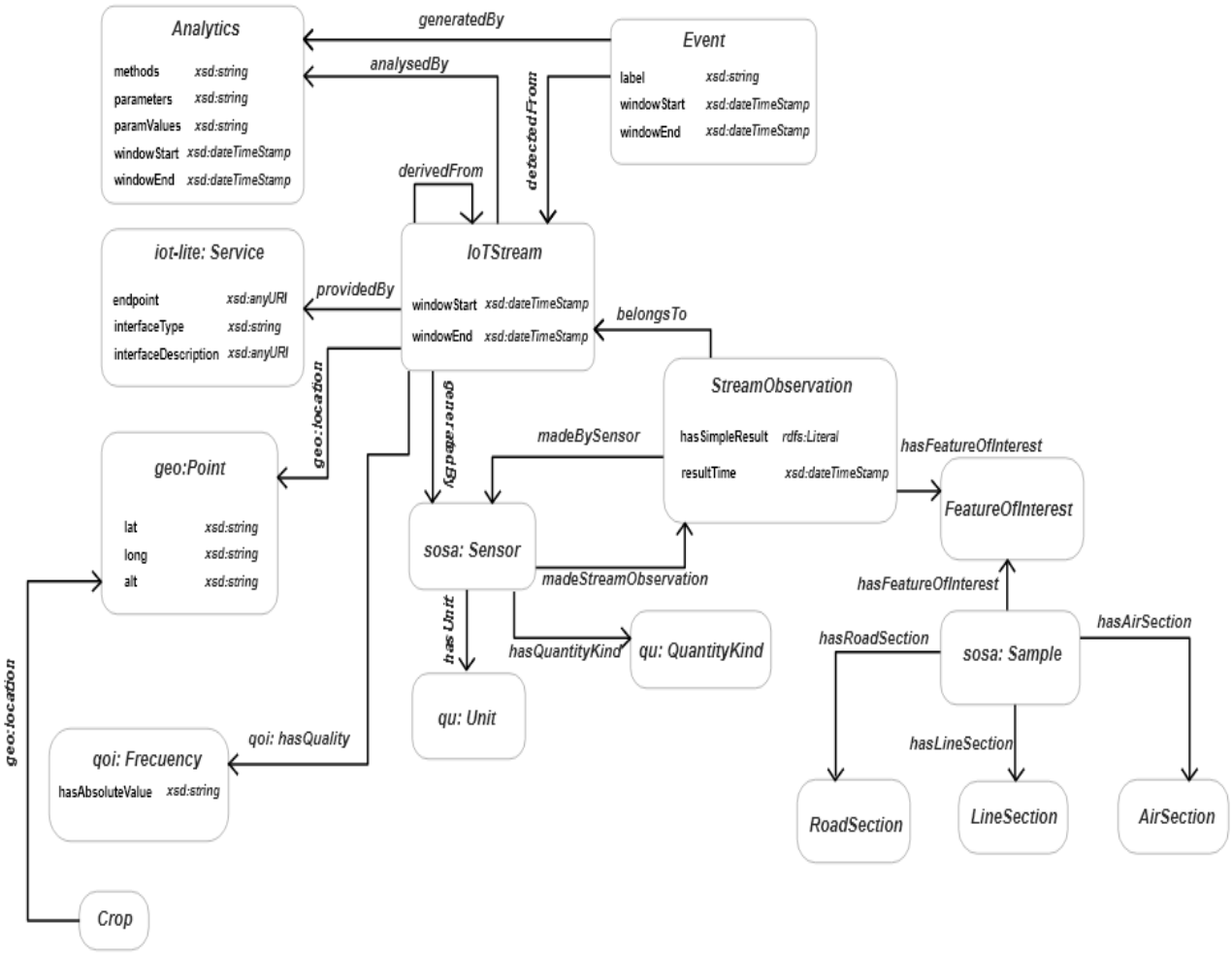
"Marta Abreu" de Las Villas. Facultad de Ingeniería Eléctrica. Departamento de Automática y Sistemas Computacionales.

Vasseur, J., Bertrand, C.P., Aboussouan, B., Gnoske, E., Pister, K., Culler, D., Acra, R., Huotari, A., 2010. A survey of several low power link layers for IP smart objects. Internet Protoc. Smart Objects IPSO Alliance.

Wang, F., Hu, L., Zhou, J., Zhao, K., 2015. A Data Processing Middleware Based on SOA for the Internet of Things [WWW Document]. J. Sens. URL https://www.hindawi.com/journals/js/2015/827045/ (accessed 5.6.21).

ANEXOS

Anexo I Clases y propiedades de la ontología.



Anexo II Propiedades de los objetos, dominio, rango y descripción.

Object Property	***Domain***	***Range***	***Description***
analysedBy	*IotStream*	*Analytics*	Relaciona cada *IotStream* con el servicio que

			realiza el análisis del mismo.
belongsTo	*StreamObservation*	*IotStream*	Relaciona una *StreamObservation* con el *IotStream* al cual pertenece.
derivedFrom	*IotStream*	*IotStream*	Un *IotStream* puede ramificarse en otros *IotStream* por lo tanto es necesario relacionar cada uno con el ancestro al cual pertenece.
detectedFrom	*Event*	*IotStream*	Relaciona un evento con el *IotStream* que lo genera.
generatedBy	*IotStream or Event*	*Sensor or Analytics*	Relaciona un *IotStream* o un evento con el sensor o el servicio de análisis

			correspondiente que lo genera.
geo:location	*Crop* *IotStream*	*geo:Point*	La relación entre algo y el punto, u otra cosa geométrica en el espacio, donde se encuentra.
hasAirSection	*Sample*	*AirSection*	Relaciona una muestra con la sección aérea a la cual pertenece.
hasFeatureOfInterest	*Observation*	*FeatureOfInterest*	Es la relación entre una observación y la entidad cuya cualidad fue observada.
hasLineSection	*Sample*	*LineSection*	Relaciona la muestra con la sección de la línea a la cual pertenece.
hasQuality	*StreamObservation*	*Frecuency*	Relaciona la una fuente de datos con su calidad.

hasQuantityKind	*Sensor*	*QuantityKind*	Relaciona un sensor con su tipo de cantidad.
hasRoadSection	*Sample*	*hasRoadSection*	Relaciona una muestra con la sección de la carretera a la cual pertenece.
hasTrafficCondition	*Train_Sensor_Observation* *Avion_Sensor_Observation* *Vehicle_Sensor_Observation*	*TrafficConditions*	Relaciona las observaciones de los vehículos con las condiciones del tráfico.
hasUnit	*Sensor*	*Unit*	Relaciona un sensor con la unidad de medida de la magnitud sensada.
isAirSectionOf	*Sample*	*AirSection*	Es el inverso de *hasAirSection*.
isLineSectionOf	*Sample*	*LineSection*	Es el inverso de *hasLineSection*.
isRoadSectionOf	*Sample*	*RoadSection*	Es el inverso de *hasRoadSection*.

madeBySensor	*Observation*	*Sensor*	Relaciona una observación con el sensor que la produce.
madeObservation	*Sensor*	*Observation*	Es el inverso de *madeBySensor*.
madeStreamObservation	*Sensor*	*StreamObservation*	Relaciona una *StreamObservation* con el sensor que la produce.
providedBy	*IotStream*	*Service*	Relaciona un *IotStream* con el servicio provisto por un dispositivo de IoT.

Anexo III Código de Python necesario para generar un mapa con la ubicación geográfica de los cultivos.

map.py

```python
import folium

#Create map object
m = folium.Map(location=[21.811154, -78.871074], zoom_start=12)

#Create markers
tooltip = "Click me!"

folium.Marker(
    [21.787103, -78.907168], popup='<b>Cultivo: Boniato</b>', tooltip=tooltip, icon=folium.Icon(color="green")
).add_to(m)
folium.Marker(
    [21.831601, -78.965899], popup="<b>Cultivo: Calabaza</b>", tooltip=tooltip, icon=folium.Icon(color="red")
).add_to(m)
folium.Marker(
    [21.864042, -78.931615], popup="<b>Cultivo: Frijoles</b>", tooltip=tooltip, icon=folium.Icon(color="purple")
).add_to(m)
folium.Marker(
    [21.811154, -78.871074], popup="<b>Cultivo: Maiz</b>", tooltip=tooltip, icon=folium.Icon(color="orange")
).add_to(m)
folium.Marker(
    [21.755576, -78.833067], popup="<b>Cultivo: Soya</b>", tooltip=tooltip, icon=folium.Icon(color="blue")
).add_to(m)

#Generate the map
m.save('map.html')
```