**3. Related work**

Using the semantic web technology is not a new topic for IoT/WoT. Many authors have addressed the interoperability problem from different point of view over the years. Some focused on the shareability layer [13][18] while others focused on the accessibility layer [16]. Some perceived the entire system under the semantic web [17]. But the common term which was used by all of them was linked-data and the application of linked data in IoT/WoT domain. Several works have been focused on using linked data technologies in IoT domain [18].

In case of our work best motivation was found from the comprehensive Recommendation by Prof. Harth in [14]. Here authors described how WoT can be seen in the lenses of linked data. Moreover, Authors recommended SoLiD to be used in the shareability layer.

## 4. Demo: WoT Application on top of SoLiD

### **4.1 The Concept**

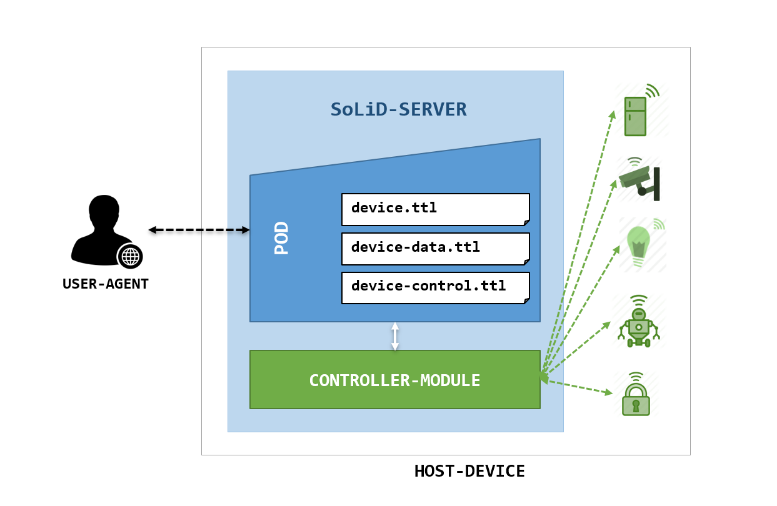
Our background study and related works have lead us to develop a concept which utilizes SoLiD ecosystem to foster WoT applications. **Figure-1** best describes our concept of integrating **Things** inside the SoLiD environment to make them accessible, controllable and sharable from the web through a semantic interface. Here, the SoLiD server acts the glue between the two entities User Agent and the Host Device. We describe these entities in the following subsections.

#### **4.1.1 Host-Device**

The Host-Device refers to the computing unit which hosts an instance of a SoLiD server. It brings all the smart things under the SoLiD ecosystem through a **controller-module**. The controller module has been developed in such a way so that it can act as the interface between the SoLiD server and the things. The POD facilitates necessary files to discover the things (device.ttl), to control them (device-control.ttl), or to share data (device-data.ttl). The host device authenticates user-agents and authorizes actions as described in the SoLiD specification. Rasberry Pi, Ardino etc. are the examples of such device.

#### **4.1.2 User-Agent**

The user-agent can be thought of as a piece of software by which users will be able to interact with the things over the web. Since it is possible to execute semantic query over these things, an intelligent machine or an application can also act as a user agent. A user agent can discover things from the POD, fetch the data and can even send control signals.



### **4.2 The Demo**

As described in the previous sub sections Figure-3 represents the Host device. We initiate our device by starting the SoLiD Server. The controller-module synchronizes all the things with the POD. Whenever any change is detected it updates the POD. On the other hand, Figure-1 represents a simple web app which simulates a user agent. Whenever, the agent presses the discover button it performs two actions: (1) Synchronizes the UI with the current sate of the POD, (2) Subscribes the POD for further updates through the Web Socket. Whenever any changes occur to any device it gets the immediate update from the POD. Figure-2 shows some example transactions between the user agent and the host device.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Figure-1** | **Figure-2** | **Figure-3** |

#### **4.2.1 Used technologies:**

We have used the Node.js development environment to develop our user agent and to simulate the host device. We have used the implementation of node-solid-server [1]. For realtime communication we have used WebSocket protocol and for semantic query we have use Rdflib.js [2]

#### **4.2.2 Things Description**

|  |
| --- |
| device.ttl |
| @prefix : <#>.  @prefix td: <https://www.w3.org/2019/wot/td#>.  @prefix ex: <https://example.org/vocab#>.  : td:thing <#1>.  : td:thing <#2>.  <#1> ex:status "on".  <#2> ex:status "off". |
|  |
| device-control.ttl |
| @prefix : <#>.  @prefix td: <https://www.w3.org/2019/wot/td#>.  <>  td:thing  “””{“id”:1, “status”:”on”}”””, “””{“id”:2, “status”:”on”}”””. |
|  |
| device-data.ttl |
| @prefix : <#>.  @prefix td: <https://www.w3.org/2019/wot/td#>.  <>  td:thing  “””{“id”:1, “status”:”on”}”””, “””{“id”:2, “status”:”on”}”””. |

### **4.3 Challenges**

The first challenge was to describe things. To overcome that we consulted with the W3C Things Descripting Ontology [1] which provides a very comprehensive set of classes and attributes to deal with the WoT things. Secondly, actuating things was the bigger problem. We solved that using the WebSocket protocol supported by the SoLiD server.

## 5. Conclusion

WoT is an application paradigm involving a set of self-independent devices capable of communicating with each other. On the other hand, SoLiD project has been developed to support the concept of decentralization and user data independence. Our Proof of concept demonstrates that SoLiD can foster WoT to a great extent. Enormous research opportunity lies here.

SoLiD was developed to support the Web infrastructure in general. It can already foster WoT application. In future we wish to develop a light-weight SoLiD Server only for WoT to ensure better performance and credibility.