SWoTSuite: A Toolkit for Prototyping End-to-End Semantic Web of Things Applications

Pankesh Patel, Amelie Gyrard, Soumya Kanti Datta, Muhammad Intizar

SWoTSuite: A Toolkit for Prototyping End-to-End Semantic Web of Things Applications

Pankesh Patel*, Amelie Gyrard**, Soumya Kanti Datta‡ and Muhammad Intizar Ali†
*ABB Corporate Research, India; **University of Lyon, France; ‡EURECOM, France;
†Insight Center for Data Analytics, Ireland
*pankesh.patel@in.abb.com, **amelie.gyrard@emse.fr, ‡dattas@eurecom.fr,
†ali.intizar@insight-centre.org

ABSTRACT

In this demonstration, we present our *S*emantic *W*eb *o*f *T*hings (SWoT) prototyping toolkit called *SWoTSuite*. It is a set of tools supporting an easy and fast prototyping of end-to-end SWoT applications. SWoTSuite facilitates - (i) automation of application development life-cycle, (ii) reducing the amount of time and effort required for developing WoT applications and (iii) an easy integration of semantic web technologies within WoT applications.

Keywords

Semantic Web of Things; Internet of Things; Semantic Web; Web of Things; Software Engineering; Cyber-Physical Systems; Toolkit.

1. INTRODUCTION

The Internet of Things (IoT) technologies have a great potential of bringing a very positive impact on many aspects of our day-to-day lives. Nowadays we can see IoT application in various areas such as agriculture and smart farming, health and fitness, smart home, smart cars and smart-x applications in smart cities. Currently, the development activities for various IoT applications are at their initial stages, and we can see availability of a few IoT applications development support tool-kits. However, with the growing popularity of IoT, we can easily foresee that in near future there will be a massive deployment of IoT devices in various domains, bringing tremendous challenges and opportunities for scientific and economic activities.

However, a true potential of IoT/WoT applications is yet to be realized and currently we see a major gap for applications connecting physical and cyber worlds. It requires a tremendous amount of manual efforts to integrate heterogeneous information, develop cross-domain IoT applications and design semantic Web of Things applications. Major challenges for fast development of IoT applications are as identified -

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3054736

Heterogeneity. IoT applications operate over an infrastructure consisting of heterogeneous devices (such as various types of sensors, user interfaces, network protocols etc.), heterogeneous interaction modes (e.g. event-driven, subscription based notifications or periodic) and heterogeneous platforms (e.g. Android, Java SE over computers, and microcontroller without any operating system).

Application development life-cycle phases. Developers face issues that are attributed to different life-cycles, including *design*, *implementation*, and *deployment*. At the design phase, the application logic has to be analyzed and separated into a set of distributed tasks for the underlying network consisting of heterogeneous entities. Then, these tasks have to be implemented for a specific platform of a device. At the deployment phase, the application logic has to be deployed over a network of devices.

Interoperability among heterogeneous IoT devices. IoT devices are often not interoperable, follow different network protocols, and store and exchange data using proprietary data formats. Additionally, mostly IoT devices only support raw sensor values lacking meta-data description, requiring specialized technology and domain knowledge before designing an IoT application.

Vertical silos of IoT applications. Currently, most of the available IoT solutions are designed keeping a single domain problem in view, resulting into vertical application development for targeted applications and lacking inter-operability, re-usability and resource sharing among IoT applications [2].

To address the above challenges, we developed SWoTSuite by combining few existing tools [1, 4], which is a complete toolkit for developing end-to-end SWoT application. More specifically, the objectives of this work is as follows:

- **Providing automation at different phases of application development life-cycle.** Our toolkit provides a set of high-level modeling languages to specify each development concern and abstracts the heterogeneity related complexity. It integrates code generation, task-mapping, and linking techniques. Code generation supports the application development phase by producing a programming framework that allows stakeholders to focus on the application logic, while our mapping and linking techniques together support the deployment phase by producing device-specific code to result in a distributed system collaboratively hosted by individual devices.
- **Reducing the time spent for developing WoT applications.** In order to create inter-operable and

cross-domain SWoT applications, developers have to perform various tasks such as designing an application, semantically annotating data and interpreting data. To perform these tasks, developers have to learn semantic web technologies and tools, which is a time consuming process and can take substantial amount of time. Reducing this gap as much as possible can be done by empowering a framework that assist developers in designing inter-operable applications with minimal knowledge of semantic web technologies.

- **Reducing the learning curve required by WoT developers to integrate semantic web technologies.** Fast prototyping of semantic-based WoT applications by hiding the use of semantic web technologies as much as possible is required to avoid the developers burden on designing ontologies, semantic annotators and reasoning mechanisms to enrich their data. An extensive work with Web frameworks (e.g., Drupal, Wordpress) has been done to design pre-defined templates to automatically generate web sites to avoid users dealing with Web technologies. Based on this idea, pre-defined templates to design SWoT applications can be created.

2. SWOTSUITE ARCHITECTURE

We envision a system that enables good decision making and actions. Figure 1 shows an architecture, inspired by our semantics-based IoT architecture [3] and IoTSuite [1]. The architecture is largely divided into three layers:

Physical layer: accessing things. It enables a device such that it can communicate information to the outside world. An IoT deployment may use gateways due to proprietary protocols. The gateways employ semantic Web technologies to explicitly describe sensor data to ensure interoperability among data.

Virtualization layer: deducing new knowledge. This layer takes data available in standard formats produced by the physical layer, infers high-level knowledge using reasoning engines, and exploits the web of knowledge available online. The suggestions as an output of this layer are provided to services and clients running in the application layer. This layer consists of the following components to deduce new knowledge from IoT data provided by the Physical layer:

−***Semantic annotator.*** It takes data from IoT devices and converts sensors metadata in a unified description to provide further reasoning in order to overcome heterogeneity issues. The sensor metadata is semantically annotated according to the M3 taxonomy [3] that is an extension of W3C Semantic Sensor Network (SSN) ontology.

−***Persistent storage.*** It stores the M3 ontologies & datasets, M3 rules, pre-written M3 compatible SPARQL queries, and unified sensor data received from the annotator. SWoTSuite uses a triple store to keep M3 ontologies, M3 datasets, and sensor data. M3 rules and SPARQL queries are stored as files.

−***Knowledge manager.*** It reuses, combines, and updates domain-specific knowledge in the persistent storage. We have been building datasets to reuse domain-specific knowledge (called as Linked Open Vocabularies for the Internet of Things (LOV4IoT)), which provides domain ontologies, datasets, and rules that could be reused to design cross-domain SWoT applications.

−***Reasoning engine.*** It infers new knowledge using a Jena inference engine and M3 rules that are extracted from LOV4IoT and redesigned to be compliant with the M3 taxonomy.

−***Query engine.*** The query engine is implemented using ARQ, a SPARQL processor for Jena. It loads the M3 ontologies & datasets, new knowledge derived using the reasoning engine, and executes SPARQL queries to provide suggestions.

Application layer: creating new applications. It builds meaningful IoT application on top of suggestions provided by the virtualization layer. The interaction of this layer with Virtualization layer is optional. More specifically, the application layer could be directly fuelled by the physical layer depends on application requirements.

IoTSuite. IoTSuite[1, 5] creates necessary infrastructure that enables SWoT applications. It takes high-level specifications as input, parses them and generates code that can be deployed on IoT sensors (e.g., temperature sensor, transportation devices) at the physical layer and IoT actuators (e.g., heater) and user interface devices (e.g., smart phone, dashboard) at the application layer. IoTSuite takes the following high-level specifications: (1) *Domain* includes specification of resources, which includes tags, sensors, actuators, and storage and third party web services. (2) *Architecture* consists of specification of computational services and interaction among them. (3) *User interaction* specifies data exchange between an application and use. (4) *Deployment specification* describes a device and its properties in a target deployment.

3. DEMONSTRATION

In this section, we present an application scenario and discuss steps involved in developing applications using SWoTSuite.

3.1 Application Scenario

Consider a real world scenario, where Alice a smart home owner, has a complete home automation system installed in her house. The home automation system is capable of performing various daily tasks automatically such as controlling heating system, lights, and burglar alarm system. The home automation system is designed, developed and deployed using IoTSuite. Alice also owns a smart car, which is well equipped with modern sensing technologies and a smart car supporting software (developed using IoTSuite) is also available for communication among various sensors within car as well as with external sources of information. Both the home automation and smart car applications are performing their required tasks within the specified domain of each application.

Now, consider a scenario, where both applications could leverage from data and information collected from each of these two applications as well external information sources. These combined rich sources of information can extend the functionality of existing applications by benefiting from the knowledge derived by another application in a complete different domain. For example, a smart home automation system at Alice home operates using a pre-planned schedule for home heating system after considering daily routine patterns of Alice arrival and departure times form home to work. On

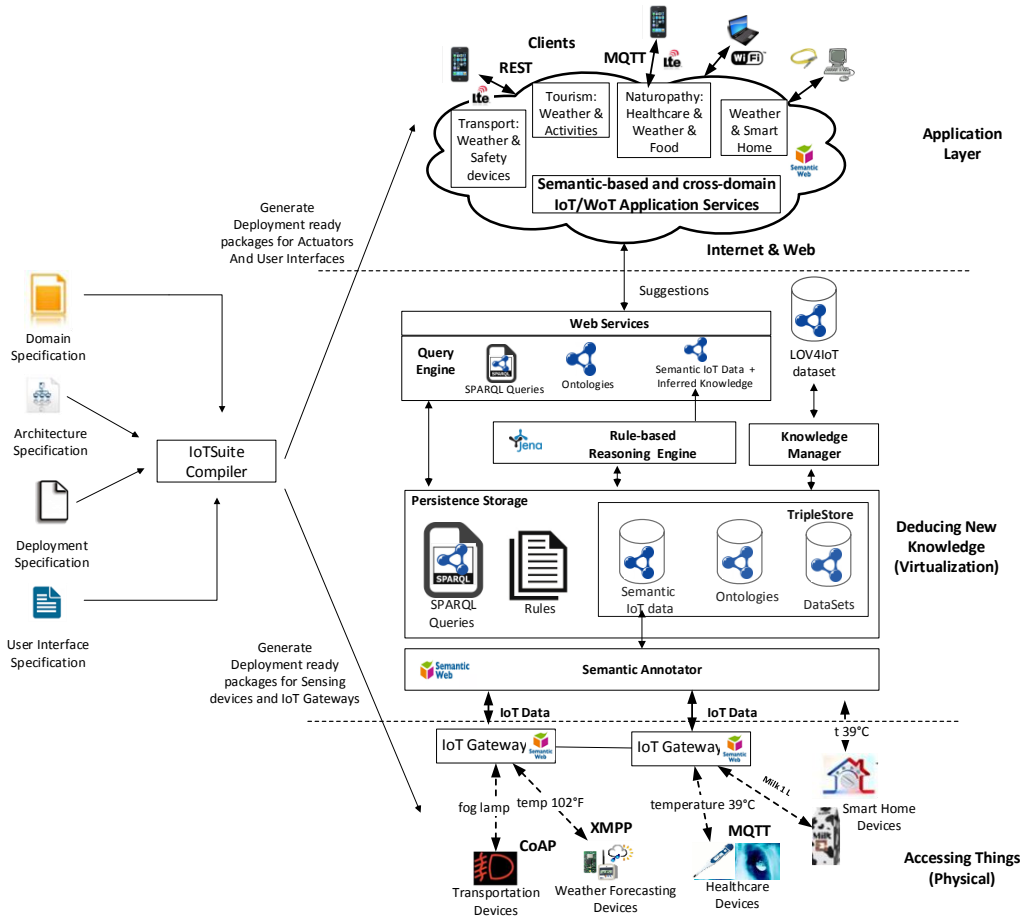

Figure 1: SWoTSuite architecture

a busy Monday evening on her way back to home from work, Alice is stuck into a severe traffic jam and her car automation system reports an expected delay of more than an hour than her usual arrival time at home. SWoTSuite builds applications could potentially process the information from car automation system to deploy actuation over the home automation system for delaying the triggering of automated heating system and thus conserving the energy consumption.

SWoTSuite can play a vital role by supporting a combination of IoTSuite and M3 Framework, where (i) *IoTSuite interacts with the physical layer and combines information collected from different smart applications*, (ii) *M3 Framework can process combined information, reason over it to extract additional knowledge, and provide useful suggestions for making knowledge-able better decisions*, and (iii) *IoTSuite application layer can take into account the extracted information from M3, to initiate the actuation process over the physical layer of SWoTSuite.*

3.2 Application Development using SWoTSuite

Our SWoTSuite demo consists of the following steps:

Step 1: Generating deployment ready packages for devices. This step of our demo is – how developers specify high-level specifications, compile them, and generate packages that can be deployed on devices at the physical layer and application layer. SWoTSuite provides an eclipse plugin[1] with an editor support to write these specifications. Figure 2 shows various editor features such as syntax coloring, error checking, auto completions, re-factoring, code-folding, and outline view.

Step 2: Annotating sensor data. This step of our demonstration is to show annotation process. It takes sensor data from the physical layer and transforms it into a unified description such as RDF/XML. Listing 1 illustrates a code snippet of temperature measurement in senML format from a temperature sensor. It shows sensed temperature value along with meta information such as unit of measurement, sensor name, and other information.

Step 3: Generating an appropriate template. This step of our demo is – generating appropriate template files that are required to build a SWoT application. This step extracts information such as sensor name and its domain from the annotated RDF/XML data and generates template files. The template consists of ontologies, datasets, rules, and SPARQL queries that are required to build a SWoT application.

Step 4: Executing reasoning engine. This step is to demonstrate the reasoning process of SWoTSuite. It deduces new knowledge from annotated IoT data using Linked Open

[1] `https://github.com/pankeshlinux/IoTSuite`

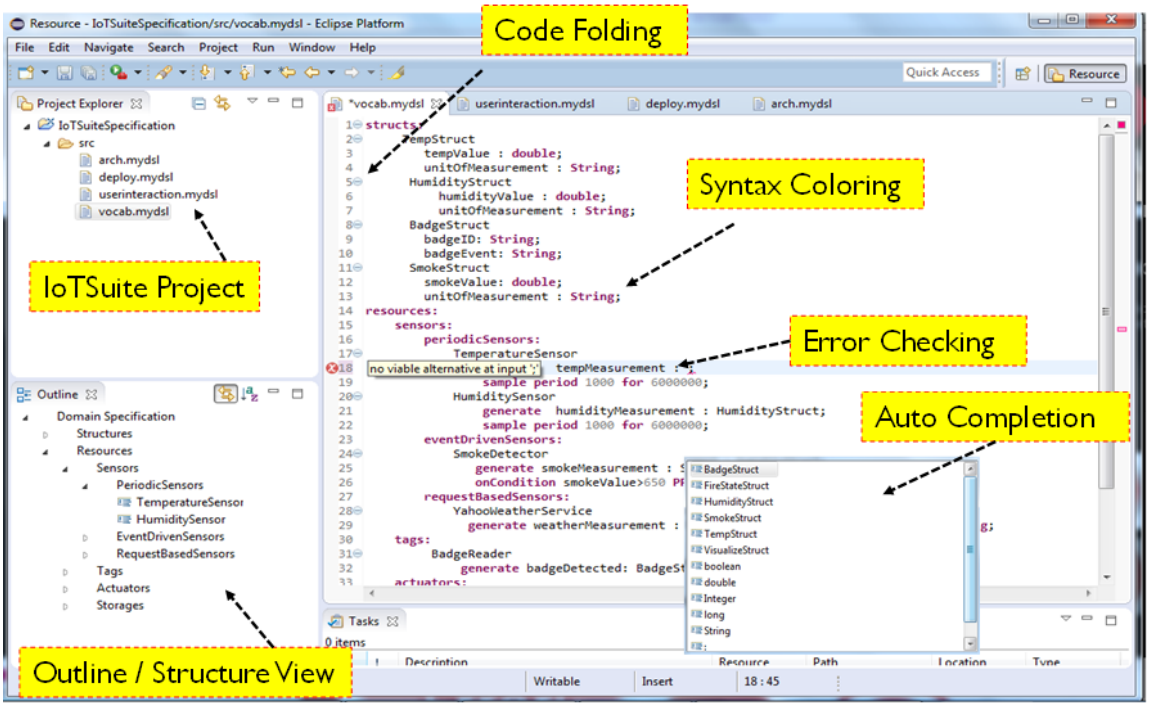

Figure 2: Editor support for writing High-level specification

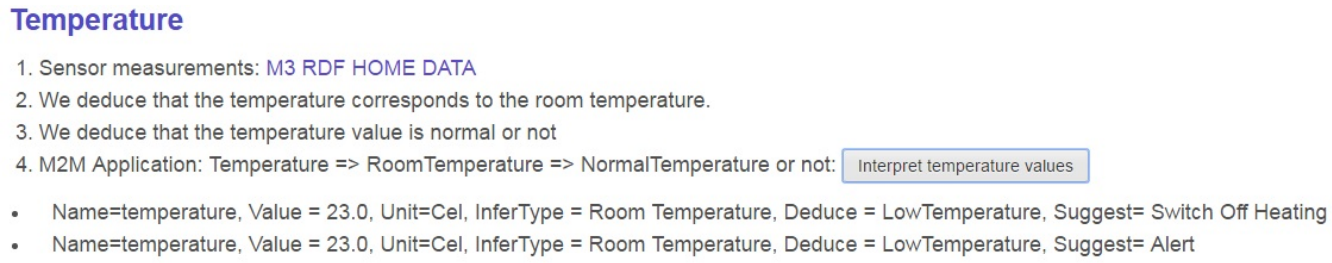

Figure 3: Suggestions from SWoTSuite

Rules available in the persistent storage. Listing 2 shows a rule example from the home-automation domain. A rule is "`if room temperature is less than 25°C, then low temperature in the room`". This rule deduces a condition based on the value from temperature sensor. Moreover, developers are guided through Java code in Eclipse IDE to load rules. Figure 4 illustrates a small code snippet to load the rules into Jena reasoning engine.

```
<senml bn="urn:thermometer:uuid:87e4fae-a8a6-79a-9abe-011c5043481d"><en="temperature" t="1392815153153" u="Cel" v="67"/>
<e n="temperature" t="1411376379709" u="Cel" v="21"/>
<e n="temperature" t="1411376392603" u="Cel" v="0"/>
</senml>
```

Listing 1: SenML/XML home temperature sensor data

```

[LowTemperature:
        (?measurement rdf:type m3:RoomTemperature)
        (?measurement m3:hasValue ?v)
        lessThan(?v,25)
        ->
        (?measurement m3:isRelatedTo home-dataset:LowTemperature)
]
```

Listing 2: Jena rule to deduce high level information

Step 5: Executing the query engine. This step is to show - how SWoTSuite generates suggestions. The query engine loads the M3 ontologies & datasets, annotated data generated at Step 2 and suggestions derived at Step 4 and SPARQL queries. Listing 3 illustrates the code snippet of a SPARQL query for the scenario in Section 3.1. This query derives recommendations over M3 ontologies, domain-specific home automation and car datasets.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX m3: <http://sensormeasurement.appspot.com/m3#>
```

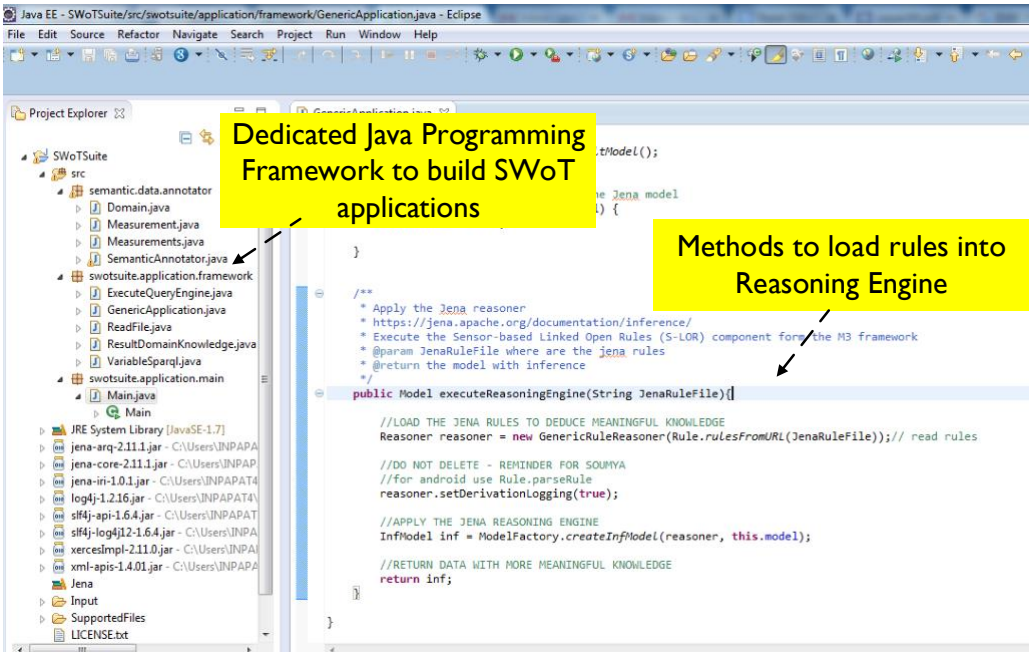

Figure 4: Java programming framework for Eclipse IDE

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX home-dataset: <http://sensormeasurement.
    appspot.com/dataset/home-dataset/>

SELECT DISTINCT ?name ?value ?unit ?inferType ?
    deduce ?suggest ?suggest_comment WHERE{
  ?measurement m3:hasName ?name.
  ?measurement m3:hasValue ?value.
  ?measurement m3:hasDateTimeValue ?time.
  ?measurement m3:hasUnit ?unit.

    # inferTypeUri = m3:RoomTemperature
    ?measurement rdf:type ?inferTypeUri.
    ?inferTypeUri rdfs:label ?inferType.

  ?inferTypeUri m3:isRelatedTo ?deduceUri.
    #deduceUri = home-dataset:LowTemperature
  ?deduceUri rdfs:label ?deduce.

    ?deduceUri m3:hasRecommendation ?suggestUri .
  ?suggestUri rdfs:label ?suggest.
    ?suggestUri dc:description ?suggest_comment.
}
```

Listing 3: Query to retrieve temperature data with enriched value

Step 6: Showing results. This step of the demonstration is to show final suggestions to a client application. The suggestions could be an actionable information such as sending alerts to a mobile application and/or controlling actuators. For our scenario described in Section 3.1, the actionable information is triggering an actuator at home with a notification to user, shown in Figure 3.

4. ACKNOWLEDGMENTS

This work is partially funded by a bilateral research convention with ENGIE Research & Development, the ANR 14-CE24-0029 OpenSensingCity project[2] and institutional collaboration supported by the Horizon 2020 Programme European project "Federated Interoperable Semantic IoT/cloud Testbeds and Applications" (FIESTA-IoT) from the European Union with the Grant Agreement No. CNECT-ICT-643943, Science Foundation Ireland (SFI) under grant No. SFI/12/RC/228, EU FP7 CityPulse Project under grant No.603095[3] and French ANR project DataTweet[4].

5. REFERENCES

[1] S. Chauhan, P. Patel, A. Sureka, F. C. Delicato, and S. Chaudhary. IoTSuite: A Framework to Design, Implement, and Deploy IoT Applications: Demonstration Abstract. In *Proceedings of the 15th International Conference on Information Processing in Sensor Networks*, pages 37:1–37:2, NJ, USA, 2016.

[2] S. K. Datta and C. Bonnet. Easing iot application development through datatweet framework. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 430–435, Dec 2016.

[3] A. Gyrard. *Designing Cross-Domain Semantic Web of Things Applications*. PhD thesis, Telecom ParisTech, Eurecom, April 2015.

[4] P. Patel, A. Gyrard, D. Thakker, A. Sheth, and M. Serrano. SWoTSuite: A Toolkit for Prototyping Cross-domain Semantic Web of Things Applications. In *Proceedings of the 15th International Semantic Web Conference(ISWC)*, 2016.

[5] P. Patel, A. Kattepur, D. Cassou, and G. Bouloukakis. Evaluating the Ease of Application Development for the Internet of Things. Technical report, Feb. 2013.

[2] `http://opensensingcity.emse.fr`
[3] http://www.ict-citypulse.eu
[4] http://www.agence-nationale-recherche.fr/?Projet=ANR-13-INFR-0008