IEICE TRANS. INF. & SYST., VOL.E99–D, NO.1 JANUARY 2016

PAPER Special Section on the Architectures, Protocols, and Applications for the Future Internet

A Design of GS1 EPCglobal Application Level Events Extension for IoT Applications*

Chao-Wen TSENG[†a)], ***Student Member***, **Yu-Chang CHEN**[†b)], ***and*** **Chua-Huang HUANG**[†c)], ***Nonmembers***

SUMMARY EPCglobal architecture framework is divided into identify, capture, and share layers and defines a collection of standards. It is not fully adequate to build IoT applications because the transducer capability is lacking. IEEE 1451 is a set of standards that defines data exchange format, communication protocols, and various connection interfaces between sensors/actuators and transducer interface modules. By appending IEEE 1451 transducer capability to EPCglobal architecture framework, a consistent EPC scheme expression for heterogeneous things can be achieved at identify layer. It is benefit to extend the upper layers of EPCglobal architecture framework seamlessly. In this paper, we put our emphasis on how to leverage the transducer capability at the capture layer. A device cycle, transducer cycle specification, and transducer cycle report are introduced to collect and process sensor/actuator data. The design and implementation of GS1 EPCglobal Application Level Events (ALE) modules extension are proposed for explaining the design philosophy and verifying the feasibility. It will interact with the capture and query services of EPC Information Services (EPCIS) for IoT applications at the share layer. By cooperating and interacting with these layers of EPCglobal architecture framework, the IoT architecture EPCglobal+ based on international standards is built.

key words: *EPCglobal ALE, EPCIS, IEEE 1451, Internet of Things, RFID tag, transducer*

1. Introduction

Many scholars point out the research challenges of IoT in recent years. The concept of IoT is given by K. Ashton of the MIT Auto ID Center in 1999 [1]. The US National Intelligence Council (NIC) foresees the general idea that things that are readable, recognizable, locatable, and controllable [2]. Fleisch presents a technical overview and economic perspective in [3]. Unlike the internet, there is no single global set of standards for the IoT. Atzori et al. propose the "one paradigm, many visions"of IoT and shape the IoT from different points of view by syntax and semantic approaches such as internet-oriented, thing-oriented, and semantic-oriented [4]. Stankovic addresses "one vision of the future is that IoT becomes a utility with increased sophistication in sensing, actuation, communications, control, and in creating knowledge from vast amounts of data"[5]. They all seem to agree that the IoT will bring a smart world and give us a qualitative change in the near future.

There are many heterogeneous objects and devices are connected to communicate and share information in IoT world. L'opez et al. explain how to add sensing functions to make IoT applications and propose the smart object conceptual framework to establish the IoT architecture [6]. Kortuem et al. propose the smart objects as building blocks for IoT applications [7]. Things on the IoT will have full interconnectivity and computation resources, being natural to consider connecting these things to the web using current paradigms. An IoT application processes objects identification, environmental monitoring, actions triggering, and intelligent management with advanced network technologies usually. Although there are many enabling technologies proposed to make the IoT architecture, it is difficult to make a common IoT architecture to fit all applications.

Mitsugi et al. provide the fundamental classification of sensor integration to GS1 EPCglobal architecture framework [8]. They provide a comprehensive reference model to integrate sensor to EPCglobal network. Hada and Mitsugi point out an Electronic Product Coding (EPC) based internet of things architecture [9]. Sung et al. propose the EPC sensor network for RFID and WSN integration [10]. The issues of RFID and sensor integration standards are discussed in [11], [12]. They try to integrate RFID and sensing technology based on EPCglobal architecture framework to achieve a cornerstone of IoT architecture. However, a detailed specification and implementation is not fully supported.

The rest of this paper is organized as follows. In Sect. 2, we describe the whole picture of IoT framework based on EPCglobal architecture framework and IEEE 1451 (has been ratified by ISO/IEC and coded ISO/IEC/IEEE 21451). We combine the EPCglobal architecture framework and IEEE 1451 and explain why these standards are suitable to be integrated and extended for IoT applications. The design issues of ALE extension for adding transducer capability such as device cycle definition, transducer cycling specification, and report design are introduced in Sect. 3. ALE modules extension is also proposed. The user interface, event process, device control, data process, and report generation modules are discussed. Section 4 shows Spec Design Tool and the implementation of ALE modules extension. The experimental results are generated to verify and validate the feasibility of ALE extension. We make a brief conclusion and point out the future work in Sect. 5.

Manuscript received April 7, 2015.
Manuscript revised July 22, 2015.
Manuscript publicized October 21, 2015.

[†]The authors are with the Department of Information Engineering and Computer Science, Feng Chia University, Taichung, Taiwan.

*This work was supported in part by the Ministry of Science and Technology, Taiwan under Grant 102-2218-E-035-006-.

a) E-mail: ttw@rfidlab.iecs.fcu.edu.tw (Corresponding author)
b) E-mail: ycchen@rfidlab.iecs.fcu.edu.tw
c) E-mail: chh@fcu.edu.tw

DOI: 10.1587/transinf.2015NTP0002

2. An IOT Architecture Based on EPCglobal Architecture Framework

There are many standards related to IoT enabling technology. RFID systems play an important role in identification of IoT. Many research institutes have insisted on the IoT architecture design and related integration issues in recent years. The IoT-A, ASPIRE, BRIDGE, SENSEI projects, sponsored by the outstanding firms, focus on IOT-related research and have focused on development, demonstration, and deployment activities. As the concept of IoT is given by Auto ID Center [1], the extension and integration issues that are based on EPCglobal architecture framework are usually discussed [8]–[10].

GS1 EPCglobal releases a collection of standards and uses Electronic Product Code (EPC) as tag data format. There are three layers introduced in EPCglobal architecture framework [13]. In EPCglobal architecture framework, C-1 Gen-2 tags are used to identify objects [14]. The raw data such as identification data is gathered at the identify layer. The identification data of tags is gathered at the identify layer. The tag data coding schemes and related document are specified and translated in Tag Data Standards (TDS) [15] and Tag Data Translation (TDT) [16].

ALE [17] middleware has the functions of filtering and grouping the EPC and related data from one or more data sources. It receives the EPC and related data, gathers data over intervals of time, filters to eliminate data that are not of interest and duplicates, groups and counts to reduce the volume of data, and reports in various forms including identifier formats. EPCIS [18] has capture service and query service. Capture applications are used to analyze event reports that are submitted from ALE middleware. Query application sends events to form real time event stream immediately and continuously. The core business vocabulary [19] is used to describe the metadata. EPCIS repository plays the role of data storage center, provides information sharing capabilities, uses web service technology, and allows other applications systems or trading partners to query and update data through standard interface.

According to our design philosophy, it is an efficient and effective way to build a framework based on the standards which are widely adopted. For a whole picture, we present an IoT framework based on the standards of EPCglobal architecture framework. For adding transducer capability to EPCglobal architecture framework, IEEE 1451 [20] is an alternative solution for supporting plug and play facility to fulfill standardization of legacy sensors and actuators.

IEEE 1451 is a series of smart transducer standards to provide a unified way to access any type, manufacturer, and wired or wireless sensors and actuators. It can be divided into Network Capable Application Processor (NCAP) and Transducer Interface Module (TIM). NCAP [21] performs network communications, TIM communications, and data conversion or other processing functions. TIM is a module that contains Transducer Electronic Data Sheet

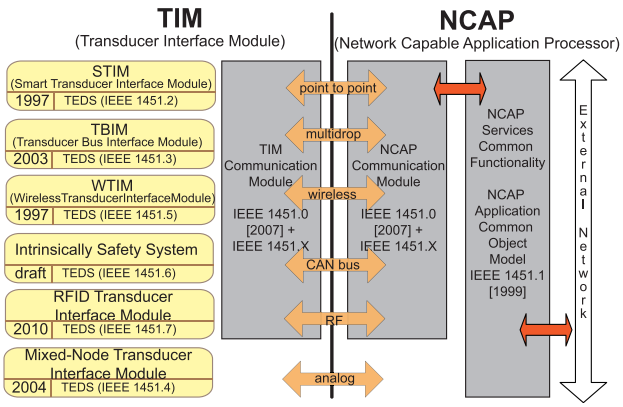

Fig. 1 IEEE 1451 reference model

(TEDS) [22], approach to implement the transducer interface, the transducer(s) or connection to the transducer(s), and any signal conversion or signal conditioning.

It features various connection interfaces between sensors/actuators and transducer interface modules, including serial, multi-drop, wireless, RFID, etc. The communication between NCAP and TIM is through IEEE 1451.X. IEEE 1451 standard defines different types of transducer communication, such as point to point, Distributed Multi drop Bus, wireless, CAN-bus and RFID [23]. IEEE 1451 reference model is shown in Fig. 1.

We propose an IoT framework based on adding IEEE 1451 transducer capability to EPCglobal architecture framework. This mechanism plays an important role on gathering raw data of heterogeneous sensors and actuators.

By referring to EPCglobal architecture framework and rotating IEEE 1451 reference model shown in Fig. 1, Fig. 2 shows the mapping mechanism of these two standards and presents a conceptual IoT framework based on extended EPCglobal architecture framework. The transducer capability such as sensing and actuating is integrated as in Fig. 2. There are three layers in EPCglobal architecture framework: identify, capture, and share layer. For supporting the transducer capability, ALE and EPCIS are extended.

Derived from the conceptual IoT framework, we can realize it by integrating IEEE 1451 to EPCglobal architecture framework extension. The proposed IoT framework based on adding transducer capability to extended EPCglobal architecture framework is shown in Fig. 3. It is divided into the following layers: *identify and transducing layer*, *capture layer*, *share layer*, and *complex event process layer*. This mechanism will help us to derive the concept of RFID applications to build IoT applications.

In this paper, we adopt the EPCglobal architecture framework [13] and IEEE 1451 standards [20] as the cornerstone of our IoT architecture framework. In addition to extend the original layers of EPCglobal architecture framework, a complex event process layer is introduced. By integrating IEEE 1451 transducer capability on extended EPCglobal architecture framework, the IoT applications can handle more versatile things and achieve more powerful func-

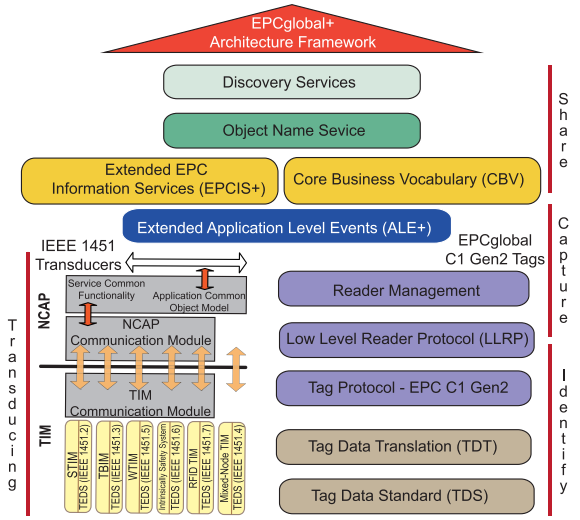

Fig. 2 Conceptual IoT framework based on GS1 EPCglobal

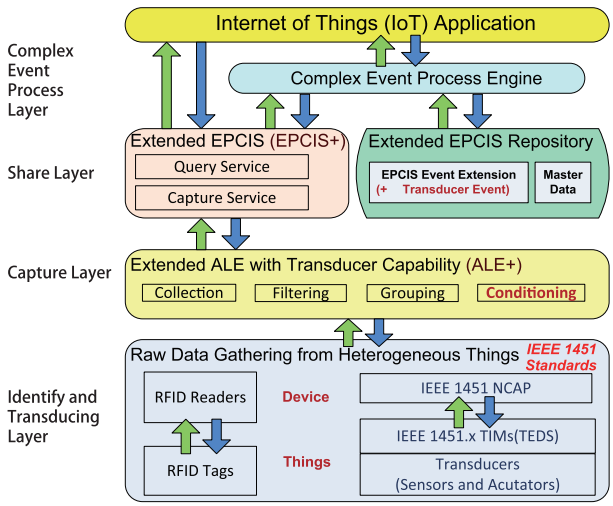

Fig. 3 IoT application framework

tions.

2.1 Identify and Transducing Layer

The RFID and smart transducer technology are adopted to identify items and sense the environment. In real deployment of IoT applications, they usually process raw data that is gathered from physical things such as tags, sensors, and actuators. These things are naturally heterogeneous and dynamic to communicate with each other.

The original EPCglobal architecture framework supports C1Gen2 RFID tags to identify objects. As we know, most of the IoT applications usually handle the raw data of identification and sensor. Transducer technology plays a crucial role of awareness in IoT. How to resolve the heterogeneity and dynamic characteristics of this variety of things is an essential issue for IoT applications.

By integrating IEEE 1451 compatible transducers to the identify layer of the extended EPCglobal architecture framework, we can process more versatile data of different things in IoT applications. According to Transducer Electronic Data Sheet (TEDS) definition of IEEE 1451, we combine TIM UUID and transducer serial number to derive the unique EPC scheme transducer ID for transducers. The applications read EPC in Pure Identity EPC URI format. At the identity layer, we introduce extended SGTIN [24] and SGXIN [25] mechanisms to make a consistent EPC scheme for heterogeneous things. The uniform EPC scheme of heterogeneous things will make a consistent expression of things transparent to capture layer. . This mechanism plays a fundamental role to support transducer capability in EPCglobal architecture framework. It is beneficial to process the gathered raw data easily and transparently at capture and share layers of extended EPCglobal architecture framework.

2.2 Capture Layer

A transducer cycle specification and report are extended in ALE to collect and process sensor/actuator data at the capture layer. As a middleware, ALE [26] supports the capture application to form events based on ECSpecs and CCSpecs definition. The raw data is sent to the capture layer and filtered by extended ALE middleware. In this layer, a middleware is used to communicate between interrogator and application that is responsible for collecting, filtering, grouping, and conditioning things data. Then, it stores events in EPCIS repository.

There is no significant data handling mechanism described in NCAP of IEEE 1451. For a consistent expression of things in identify and transducing layer, filtering the sensor data stream is considered as an alternative way. We put our emphasis on ALE extension in this paper.

2.3 Share Layer

EPCIS is in charge of receiving the events that are sent from capture application. A key feature of EPCIS is its ability to be extended to adapt to particular business situations. There are four event types on EPCIS to form business transactions. They are object event, aggregation event, transaction event, and quantity event. A new event type may be added in Data Definition Layer for transducers. At the share layer, a transducer event (such as sensor event and actuator event) is introduced to extend EPCIS for adding transducer capability and cooperates with capture and query services of original EPCIS.

These events are stored at EPCIS repository. EPCIS repository plays the role of data storage center, provides information sharing capabilities, uses web service technology, and allows the other application systems or trading partners to query and update data through standard interface. EPCIS event repository must also be extended to store transducer events.

For the consideration of semantics and ease of use for IoT applications, we append transducer event to EPCIS at the share layer. EPCIS processes events and stores them

into EPCIS repository for query service. Finally, IoT applications can handle events by query service of EPCIS.

2.4 Complex Event Process Layer

In addition to extend the original layers of EPCglobal architecture framework, a complex event process layer is introduced too. A complex event is an event that is generated or triggered by other events. This means that when multiple events occur, you can identify significant events from the event stream, analyze their impact, and handle them by pre-defined event pattern.

By combing with EPCIS events and services, the defined event patterns or reasoning rules, complex event process mechanism can decide whether an action will be triggered when a significant event occurs. The mechanism of handling complex sensing events which are generated and aggregated from things such as tags, sensors, and actuators is also proposed. By this mechanism, the objective of event handling without human intervention will be achieved.

3. Design Issues of EPCglobal ALE Extension

For handling transducer data, EPCglobal architecture framework must be extended. Like reader cycle in original EPCglobal ALE, a device cycle is introduced. Furthermore, the design concept of transducer cycle that includes sensor and actuator cycle specifications is introduced. A transducer report is proposed too. EPCglobal ALE is not suitable for operating, managing, and filtering transducers and their data. We extend the capability and functions of ALE to support the transducer capability.

After extension and modification, extended ALE can operate and manage transducers, filter the data collected from transducers, and cooperate with identify and share layers to build the IoT applications.

3.1 Device Cycle Definition and Operation

In EPCglobal ALE, a reader cycle is the basic unit for gathering data from RFID readers. An event cycle means the data collecting cycle and a command cycle represent actions execution cycle. In IoT environment, not only tags but also transducers data must be collected, grouped, and filtered, so that reader cycle cannot be applied in these situations. We introduce a device cycle as the basic unit for heterogeneous things. In a device cycle, an ALE client can gather data from tags, sense the environment by sensors, and do some actions by the state of actuators. Like the event and command cycle in ALE, we extend the transducer cycle for transducers. For semantics, the sensor and actuator cycle are introduced for different types of transducers. After these cycles are specified, reports is generated and sent back to the client.

We examine the event and command cycle of EPCglobal architecture framework at the capture layer and add the transducer cycle on it. Figure 4 shows the raw data handling of heterogeneous things on dynamic deployment

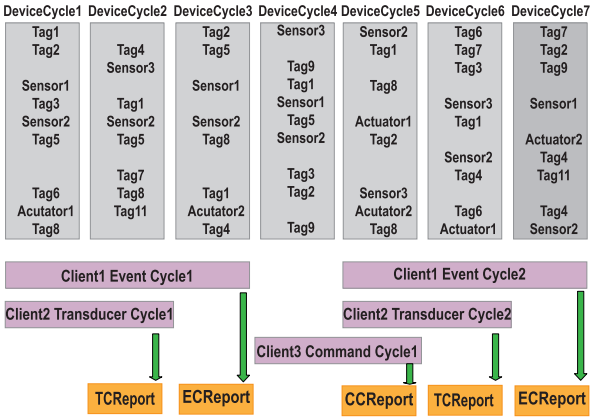

Fig. 4 Raw data handling in ALE middleware

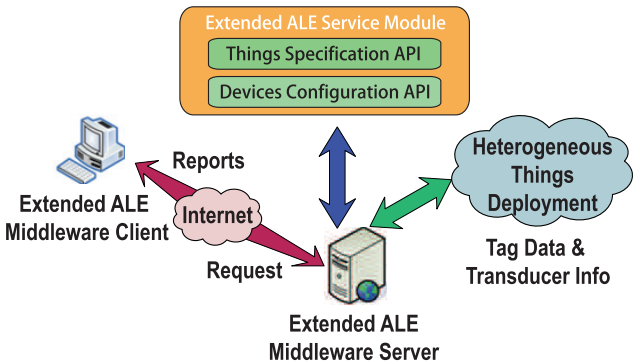

Fig. 5 ALE extension for transducer

in ALE middleware. TCReport is generated like ECReport and CCReport.

ALE is used to collect and filter tag and transducer data. As a middleware, It produces output data which are sent to capture applications. A capture application generates events which are stored in EPCIS repository. Data operations for transducers can be extended to others than collecting, filtering and aggregation in EPCglobal ALE. For IoT application, EPCglobal ALE have the ability to process transducer ID and corresponding data.

Figure 5 uses ECSpecs/CCSpecs/TCSpecs as inputs of ALE middleware client. It shows data handling process and reports generation.

ALE client specifies the fieldspec and describes ECSpecs/CCSpecs/TCSpecs for each data fields. The fieldspec is a combination of fieldname, data, and format. For adding transducer capability, we can store the required TEDS data using the fieldspec on User Bank and get it by specific API. There are some formats to specify the fieldname of fieldspec, such as:

- *fixed fieldname:@bank.length.[offset]*
- *variable fieldname: @bank.oid*
- *generic symbolic fieldname: @userbank.length.[offset]*
- *built-in fieldname: such as epc,killpwd, accesspwd, epcbank, tidbank, userbank, etc.*

ALE 1.1 also provides facilities for collecting, filtering, grouping, and reporting of non-EPC tag data by defining *TMSpec(specname:string, spec: TMSpec):void*. TMSpec is abstract class. We can use this mechanism to collect, filter, group, and report transducer ID and corresponding data. The tag and transducer data are gathered in the identify layer. By the extended definition of PrimaryKeyFields, ALE client can view the transducer ID and transducer data as a KeyField. It describes the EPCglobal ALE extension for adding transducer capability. A capture application generates events according to system requirements and stores events to EPCIS repository for future use.

3.2 Transducer Cycling Specification (TCSpec) and Report (TCReport) Design

The events on EPCglobal architecture framework are derived from specifications such as ECSpecs and CCSpecs. ECSpecs and CCSpecs are enough to handle the read and write operation for RFID tags. For extending the transducer capability, it is an intuitive alternative to design TCSpecs for transducer data collection. TCSpecs and TCReport are extended in ALE to collect and process sensor/actuator data and a transducer event type is enhanced in EPCIS to record sensor/actuator events which can be queried for IoT applications.

TCSpecs definition is compatible with EPCglobal specifications like ECSpecs and CCSpecs. We can define TCSpecs by ALE Reading API with methods of subscribe, poll, and immediate. TCSpecs must specify logicalTransducers, boundarySpec, reportSpec, cmdSpec, includeSpecInReport, etc.

For semantics, we derive the concept of TCSpecs and implement Sensor Cycling Specifications (SCSpecs) and Actuator Cycling Specifications (ACSpecs) for sensors and actuators. The design of SCSpecs and ACSpecs is similar to ECSpecs and CCSpecs.

3.2.1 SCSpec Design

The major fields in SCSpec are logicSensors, boundarySpec, and reportSpecs. Figure 6 shows the design of SCSpecs.

- *logicalSensors*: Users can specify which sensors should apply this specification. Some sensors and sensor modules can be viewed as logical sensors to be applied.
- *boundarySpec*: It specifies temporal constraints of this spec. The duration, repeat period, and when data is available are specified in this field.
- *reportSpecs*: It specifies the report will be generated by which SCReportspec. The SCReportSpec contains: reportName, reportSet, filterSpec, and groupSpec etc. The original set values of reportSet such as Current, Addition, and Deletion are reserved and have similar meaning. For the characteristic of sensor, Active and Sleep are added.
- *includeSpecInReports*: It specifies whether the content of SCSpecs will be appended in report or not.
- *extension*: For example, a user can set "dangerous temperature reading" to express the warm temperature in reading cycle.

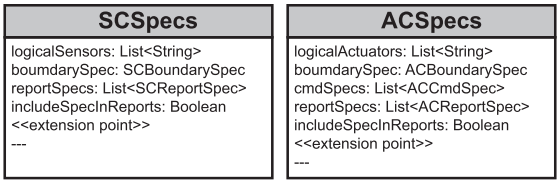

Fig. 6 SCSpecs and ACSpecs design

3.2.2 ACSpec Design

The design of ACSpecs is shown in Fig. 6. The major fields in ACSpecs are logicActuators, boundarySpec, cmdSpecs, and reportSpecs.

- *logicalActuators*: A lot of actuators and actuator module can be treated as a logical unit to apply the specification.
- *boundarySpec*: It specifies temporal constraints of this spec. The duration, repeat period, start and stop trigger list, and stop after error count are specified in this field.
- *cmdSpecs*: It specifies which actions will be triggered by an ACCmdSpec.
- *includeSpecInReports*: It specifies whether the content of ACSpecs will be appended in report or not.
- *extension*: It is reserved for users to add different fields for describing various situations.

3.3 Transducer Reports for ALE Client

ECSpecs/CCSpecs/TCSpecs generate reports in ALE middleware client. Like ECReport, TCReport contains the transducer data that are defined in TCReportOutputFieldSpec. According to SCSpecs and ACSpecs, one or more reports are generated from it.

Figure 7 shows the SCReport and ACReport Design. The *initiationCondition* indicates the device cycle beginning condition such as repeat period or trigger. The *reports* and *cmdReports* store the content of reports after ALE middleware client execution.

ALE middleware client will generate the SCReport and ACReport according to SCSpecs and ACSpecs. Figure 8 shows XML templates of SCReport and ACReport.

3.4 ALE Module Extension for the Transducer

EPCglobal ALE is not suitable for operating, managing, and filtering transducers and their data. For ALE extension data, we must increase setting transducer devices, define transducer communication parameters, and define transducer data format. User Interface Module, Event Process

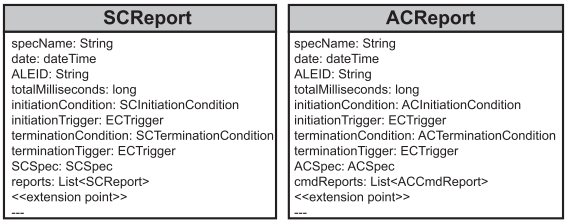

Fig. 7 SCReport and ACReport design

```
<ns2:SCReports xmlns:ns2="…"
    specName="…" date="…" ALEID="…"
    …… "schemaVersion=" ">
   <reports>
      <report reportName="…"/>
   </reports>
   <SCSpec>...</SCSpec>
   <extension>...</extension>
</ns2:SCReports>
```

```
<ns2:ACReports xmlns:ns2="…"
    specName="…" date="…" ALEID="…" ……>
    <CmdReports>
      <CmdReport>
        <sensorReports>…</sensorReports>
        <actuatorReports>…</actuatorReports>
      </CmdReport>
    </CmdReports>
    <ACSpec>...</ACSpec>
    <extension>...</extension>
</ns2:ACReports>
```

Fig. 8 Templates of SCReport and ACReport

Module, Device Control Module, Data Process Module, and Report Generator Module must be extended. Figure 9 shows interactions of these modules.

ECSpecs/CCSpecs/SCSpecs/ACSpecs are specified by users and are sent to ALE middleware client. The User Interface Module analyzes specifications by state machine. The raw data is gathered by Device Control Module. Event Process Module handles the raw data based on the specs. After the functions of Data Process Module, reports are generated by Report Generator Module.

Some sub modules related to transducer must be modified such as Device Configuration Module, Things Spec Management Module, Collecting Module, Filtering Module, Grouping Module, and Report Generation Module. The Sensing Module, Actuating Module, Transducer Module, Sensor Cycle Process Module, Actuator Cycle Process Module, and Conditioning Module must be designed for adding transducer capability on EPCglobal ALE. Figure 10 shows ALE modules extension.

For adding transducer capability on EPCglobal ALE, we extended the configuration and specification management modules in User Interface Module. Device Configuration Module can configure the interrogator devices such as readers and transducers. The Sensing and Actuator Modules are designed for sensor and actuator. Analysis Spec and Database functions are extended for TCSpecs analysis and SCSpecs and ACSpecs storage.

Besides User Interface Module, Event Process Module must be extended too. We add Sensor and Actuator Process Module for transducers. The Analysis XML block in Event Process Module is modified for handling the criteria temporal constraints such as duration and repeatPeriod, which are specified in boundarySpec.

We use the word device instead of reader for broad usage. Device Control Module has the capability of gathering data from sensors and actuators. For IEEE 1451 compatible transducers, we can add a Transducer Module such as IEEE

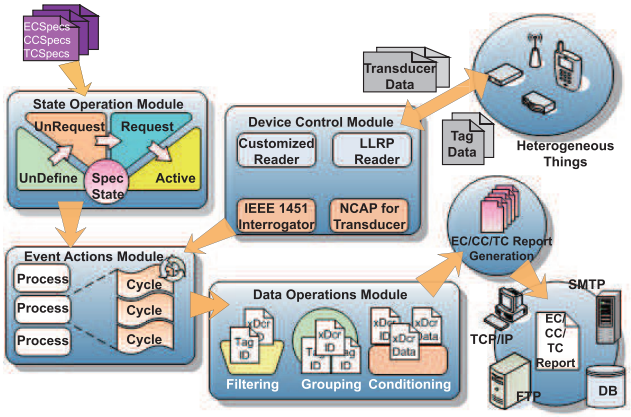

Fig. 9 Interactions of ALE modules extension

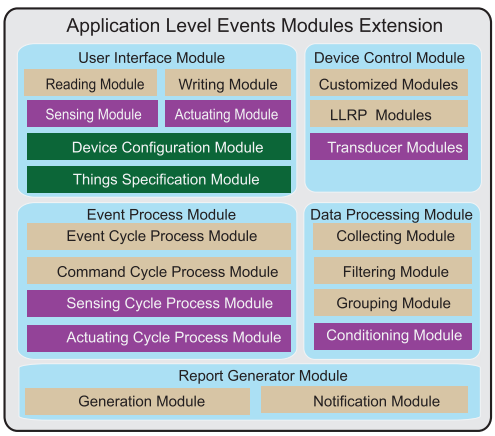

Fig. 10 ALE modules extension

1451 interrogator driver in Device Control Module.

As to the streaming raw data of transducers, it is necessary to reduce the volume for effective usage. Conditioning Module is appended to reduce the volume of streaming in the Data Process Module. The functions such as LastN and MaxValue are designed to get more precise and useful data. We modify Collecting, Grouping, and Filtering Modules by adding transducer capability on Analysis XML block, and IFiltering, and IGrouping interface.

For generating SCReport and ACReport, we must modify the Analysis and Report blocks in Report Generator Module. After that, reports will be generated and sent to EPCIS repository.

4. Implementation and Experimental Results

An IoT framework must have the identification and transducing ability to handle situations of the real world. This paper puts emphasis on adding the transducer capability to EPCglobal ALE extension. Most of the deployment of IoT applications can be treated and summarized as shown in Fig. 11. These layers can be mapped to EPCglobal architecture framework intuitively.

In the above architecture, the interrogator reads the raw

data of tags and transducers at interrogator zone. The original raw data contains identification data of RFID tags and sensor data of transducers. The transducer ID is sent to middleware by SGXIN format. The other TEDS data of transducer is sent to the upper layer by User Bank of Tag Memory and physical memories of interrogator.

Transducer Module in Device Control Module can get raw data of sensors and actuators. IEEE 1451 interrogator driver or ALE middleware can analyze the TEDS and communicate with internet by NCAP. Figure 12 shows the system architecture of ALE modules implementation for extension. We use Apache Tomcat as a server platform and develop a web service oriented middleware for ALE extension.

4.1 Use Case for Experimental Simulation

The SmartHome application is used to demonstrate ALE extension. This application can reduce the manual operations of home management effectively and support the existing home management system. We expect that it is even further extended to other areas of application such as eco-park and health care systems.

The transducer capability of SmartHome application is divided into sensing and triggering. The temperature sensors are located in rooms. The smart actuator has the capability to monitor the room temperature and can be triggered to set the room temperature by specific criteria.

As we have mentioned in Sect. 2.1, the uniform EPC scheme of transducers such as temperature sensors and trigger actuators is listed by SGXIN format as follows. The transducers will be resolved by the same mechanism at interrogator or ALE middleware like tags.

- Room1TEMPSensor

  urn:epc:id:sgxin.1.94994.1.492181.0.2015.226038.10

- Room3AirConActuator

  urn:epc:id:sgxin.1.94994.1.492181.0.2015.226038.30

We use Spec Design Tool in Fig. 13 to specify sensor detection and actuator behaviors. We demonstrate how to design SCSpecs and ACSpecs and generate reports after specifications is executed.

4.2 Walkthrough of SCSpec and SCReport

There are many temperature sensors located in two rooms. They are used to monitor the temperature for controlling air conditioners. We can view them as logical sensor Room1TEMPSensor and Room2TEMPSensor. The duration of Sensor Cycle is set to 10 seconds and repeat period is 15 seconds. We want to get the maximum five temperature values between 20 and 25 °C.

4.2.1 SCSpec

According to the scenario, SCSpecs can be specified by Spec Design Tool in Fig. 13. We use the XML format to represent the specification of SCSpecs in Fig. 14, boundarySpec is listed in Fig. 15, and reportSpec is listed in Fig. 16.

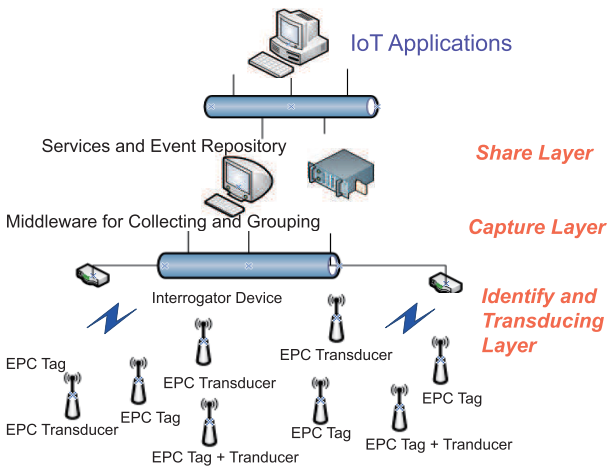

Fig. 11 Deployment of IoT applications

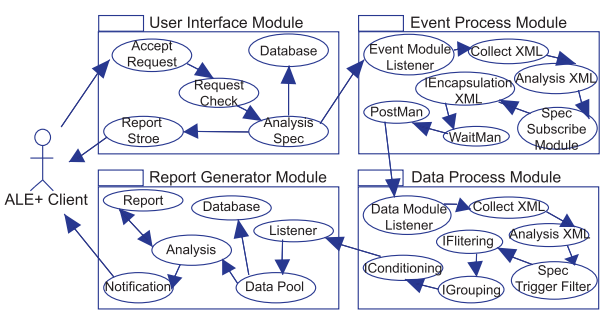

Fig. 12 Implementation modules of ALE extension

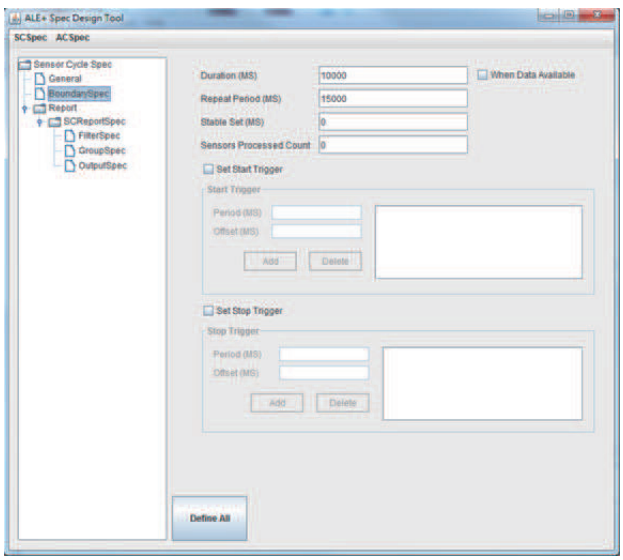

Fig. 13 Spec design tool

```
<ns2:SCSpec xmlns:ns2="urn:rfidlab:aleplus:xsd"
includeSpecInReports="true" schemaVersion="10"
creationDate="2014-09-03T23:30:28.873+08:00">
        <logicalSensors>
                <logicalSensor>Room1TEMPSensor</logicalSensor>
                <logicalSensor>Room2TEMPSensor</logicalSensor>
        </logicalSensors>
        <boundarySpec>…</boundarySpec>
        <reportSpecs>…</reportSpecs>
        <includeSpecInReports> true </includeSpecInReports>
        <extension/>
</ns2:SCSpec>
```

Fig. 14 SCSpecs design

```
<boundarySpec>
    <startTriggerList/>
    <repeatPeriod unit="MS">15000</repeatPeriod>
    <stopTriggerList/>
    <duration unit="MS">10000</duration>
    <stableSetInterval unit="MS">0</stableSetInterval>
    <whenDataAvailable>true</whenDataAvailable>
    <sensorProcessedCount>0</sensorProcessedCount>
    <extension/>
</boundarySpec>
```

Fig. 15 boundarySpec of SCSpecs

```
<reportSpecs>
  <reportSpec reportName="SCReportSpec_140503_1"
   reportIfEmpty="true" reportOnlyOnChange="false">
  <reportSet>ACTIVE</reportSet>
  <filterSpec>
    <filterList>  <filter>
       <sampling> <counter>5</counter> <dataset>MAXIMUM</dataset> </sampling>
    <fieldSpec>
       <fieldname>data</fieldname> <datatype>float</datatype>
      <format>raw-decimal</format>
    </fieldSpec>
    <condList>
       <condition> <interval>GE</interval> <value>20</value> </condition>
       <condition> <interval>LE</interval> <value>25</value> </condition>
    </condList>
    </filter> </filterList>
    <extension/>
  </filterSpec>
  <groupSpec> ..... <pattern>urn:epc:pat:sgxin.*.*.*.*</pattern> ..... </groupSpec>
  <output includeHex="false"includeDecimal="true"includeBinary="false"> .... </output>
  <statProfileNames>Timestamps</statProfileNames>
  <extension/>
  </reportSpec>
</reportSpecs>
```

Fig. 16 reportSpec of SCSpecs

4.2.2 SCReport

SCSpecs is configured at ALE middleware client. After execution, SCReport is generated as in Fig. 17.

4.3 Walkthrough of ACSpec and ACReport

There is a smart actuator located in a room. The smart actuator has the capability to monitor the room temperature and can be triggered to set the room temperature by specific criteria. It is named as Room3AirConActuator. The duration of Actuator Cycle is set to 10 seconds and repeat period is 5 seconds. For the purpose of energy conservation, room temperature will be set between 26 and 28 °C. If the minimum value of room temperature is less than 26 °C, smart actuator will be triggered and adjust temperature to 27°C.

```
<SCReports xmlns:ns2 = " urn:rfidlab:aleplus:xsd "
    ALEID = "RFIDLAB-ALEplus-1" date = "2014-09-06 03:34:00.292+08:00"
    initiationCondition="REQUESTED" initiationTrigger=""
    schemaVersion="10" specName="SCSpec"
    terminationCondition="DURATION" terminationTrigger=""
    totalMilliseconds="10014">
    <reports> <report reportName="SCReportSpec_140503_1">
     <group groupName="urn:epc:pat:sgxin.*.*.*.*">
       <groupList> <member>
          <TRType>Sensor</TRType>
          <WorkType>TEMPERATURE</WorkType>
          <Data> <Data_Dec>24.0</Data_Dec> </Data>
          <Accuracy>0.2</Accuracy>
          <Stat statType="SCTimestampStat">
            <FirstSenseTime>2014-09-06 03:33:55.879+08:00</FirstSenseTime>
          </Stat>
       </member> </groupList>
     </group>
  </report> </reports>
</SCReports>
```

Fig. 17 SCReport XML format

```
<ns2:ACSpec xmlns:ns2="urn:rfidlab:aleplus:xsd"
    includeSpecInReports="true" schemaVersion="10"
    creationDate="2014-09-03T23:30:28.873+08:00">
    <logicalActuators>
       <logicalAtuator>Room3AirConActuator</logicalActuator>
    </logicalActuators>
    <boundarySpec>…</boundarySpec>
    <cmdSpecs>…</cmdSpecs>
    <reportSpec> ... </reportSpec>
    <inclueSpecInReports> false </inclueSpecInReports>
    <extension/>
</ns2:ACSpec>
```

Fig. 18 ACSpecs design

```
<boundarySpec>
    <startTriggerList/>
    <repeatPeriod unit="MS">15000</repeatPeriod>
    <stopTriggerList/>
    <duration unit="MS">10000</duration>
    <actuatorProcessedCount>0</actuatorProcessedCount>
    <afterError>true</afterError>
    <extension/>
</boundarySpec>
```

Fig. 19 boundarySpec of ACSpecs

4.3.1 ACSpec

According to scenario, ACSpecs can be specified using Spec Design Tool. We use the XML format to represent specified ACSpecs in Fig. 18. The boundarySpec and cmdSpec are listed in Fig. 19 and 20.

4.3.2 ACReport

ACSpecs is configured at ALE middleware client. After execution, ACReport is generated in Fig. 21.

4.4 SCReport and ACReport XML for Applications

XML reports are used to be parsed and stored at EPCIS repository. EPCIS repository is located between the capture layer and the share layer of EPCglobal architecture framework. It plays the role of data storage center, provides information sharing capabilities, uses web service technology, and allows other applications or trading partners to query

```
<cmdSpecs>
 <cmdSpec name="ACCommandSpec_140505" reportIfEmpty="true">
  <filterSpec>
    <filterList> <filter>
      <sampling> <counter>1</counter> <dataset>MAXIMUM</dataset> </sampling>
      <fieldSpec> ..... </fieldSpec>
      <condList>
        <condition> <interval>LE</interval> <value>26</value> </condition>
    </filter> </filterList>
    <extension/>
  </filterSpec>
  <opSpecs>
    <opSpec> <opType>ADJUST</opType>
    <fieldspec> ..... </fieldspec>
    <dataSpec> <specType>LITERAL</specType> <data>SET:26</data> </dataSpec>
    <opName>OP_AirCon</opName>
    </opSpec>
    <statProfileNames>Timestamps</statProfileNames>
  </opSpecs>
 </cmdSpec>
</cmdSpecs>
```

Fig. 20 cmdSpec of ACSpecs

```
<ACReports xmlns:ns2=" urn:rfidlab:aleplus:xsd "
  ALEID="RFIDLAB-ALEplus-1"  date="2014-09-06 01:27:48.264+08:00"
  initiationCondition="REQUESTED" initiationTrigger="" schemaVersion="10"
  specName="ACSpec" terminationCondition="DURATION"
  terminationTrigger="" totalMilliseconds="13034">
  <cmdReports> <cmdReport name="ACCommandSpec_140505">
    <sensorReports> <sensorReport>
      <TRType>Sensor</TRType>
      <WorkType>TEMPERATURE</WorkType>
      <Data>25.8</Data>
      <Accuracy>0</Accuracy>
      <Stat statType="SCTimestampStat">
        <FirstSenseTime>2014-09-06 01:27:40.827+08:00</FirstSenseTime>
      </Stat>
    </sensorReport> </sensorReports>
    <actuatorReports> <actuatorReport>
      <TRType>Actuator</TRType>
      <WorkType>TEMPERATURE</WorkType>
      <Data>SET_SUCESS</Data>
      <Accuracy>3</Accuracy>
      <Stat statType="SCTimestampStat">
        <FirstActuateTime>2014-09-06 01:27:40.827+08:00</FirstActuateTime>
      </Stat>
    </actuatorReport> </actuatorReports>
  </cmdReport> </cmdReports>
</ACReports>
```

Fig. 21 ACReport XML format

and update data through standard interface.

EPCIS is in charge of receiving events that are sent from capture application. A key feature of EPCIS is its ability to be extended to adapt to particular business situations. There are four event types on the newest version of EPCIS to form business transactions. These events are stored at EPCIS repository.

Object Event is simply about tag IDs read by a RFID reader and the basic components of an object event is described as (id, time, location). In addition to original events, two types of events are captured in extended EPCIS, sensor events and actuator events. These events will be handled by query service of extended EPCIS. It will help us to build IoT applications like versatile RFID applications in original EPCglobal architecture framework.

5. Conclusions and Future Works

GS1 EPCglobal defines a collection of standard specifications utilizing RFID technology to support identification applications. IEEE 1451 is a series of smart transducer standards. Based on the extension of EPCglobal architecture framework, an IEEE 1451 smart transducer is integrated at identify layer. For adding transducer capability, the design and implementation of EPCglobal ALE extension is discussed in this paper. By cooperating with the other layers of EPCglobal architecture framework, an IoT architecture based on international standards is built. With the identification and transducer capability, the IoT framework provides more possibility of IoT applications.

How to realize the IoT framework from a feasibility perspective will be a focus of ongoing study. We will further continue to enhance the proposed mechanisms.

References

[1] K. Ashton, “That ‘Internet of Things’ Thing,” RFID Journal, 2009. http://www.rfidjournal.com/article/view/4986/
[2] National Intelligence Council, “Disruptive e Civil Technologies: Six Technologies with Potential Impacts on US Interests out to 2025,” 2008.
[3] E. Fleisch, “What is the Internet of Things? - An Economic Perspective,” Auto-ID Labs White Paper WP-BIZAPP-053, 2010.
[4] L. Atzori, A. Iera, G. Morabito, “The Internet of Things: A survey,” Comput. Netw., vol.54, 2010, pp.2787–2805.
[5] J.A. Stankovic, “Research Directions for the Internet of Things,” IEEE Internet Things J., vol.1, no.1, pp3-9, 2014.
[6] T.S. López, D.C. Ranasinghe, M. Harrison, D. McFarlane, “Adding sense to the Internet of Things” An architecture framework for Smart Object systems,” Personal and Ubiquitous Computing, vol.16, 2012, pp.291–308.
[7] G. Kortuem, F. Kawsar, V. Sundramoorthy, and D. Fitton, “Smart objects as building blocks for the internet of things,” IEEE Internet Comput., vol.14, no.1, pp.44–51, 2010.
[8] J. Mitsugi, T. Inaba, B. Patkai, L. Theodorou, J. Sung, T.S. López, D. Kim, D. McFarlane, H. Hada, Y. Kawakita, K. Osaka, and O. Nakamura, “Architecture Development for Sensor Integration in the EPCglobal Network,” Auto-ID Labs White Paper WP-SWNET-018, 2007.
[9] H. Hada and J. Mitsugi, “EPC based internet of things architecture,” IEEE International Conference on RFID Technologies and Applications, 2011, pp.527–532.
[10] J. Sung, T.S. López, D. Kim, “The EPC Sensor Network for RFID and WSN Integration Infrastructure,” Proc. Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops(PerComW’07), 2007.
[11] T.S. López, “RFID and sensor integration standards: State and future prospects,” Computer Standards and Interfaces, 2011, pp.207–213.
[12] C. Chen and S. Helal, “Sifting through the jungle of sensor standards,” Pervasive Comput., 2008, pp.84–88.
[13] K. Traub, F. Armenio, H. Barthel, P. Dietrich, J. Duker, C. Floerkemeier, J. Garrett, M. Harrison, B. Hogan, J. Mitsugi, J. Preishuber-Pfluegl, O. Ryaboy, S. Sarma, K. Suen, and J. Williams, The EPCglobal Architecture Framework, final version 1.6. GS1, EPCglobal, 2014.
[14] EPCglobal, EPC Radio-Frequency Identity Protocols Generation-2 UHF RFID specification for RFID air interface protocol for communications at 860 MHz–960 MHz, version 2.0.0, GS1, EPCglobal, 2013.
[15] EPCglobal, EPCglobal tag data standards, version 1.9, GS1, EPCglobal, 2014.
[16] EPCglobal. GS1 EPCglobal Tag Data Translation (TDT) 1.6, Ratified standard, GS1, EPCglobal, 2011.
[17] EPCglobal, The application level events (ALE) specification, version 1.1.1, part I: Core specification, GS1, EPCglobal, 2009.
[18] EPCglobal, EPC information services (EPCIS), version 1.1, GS1, Apr, 2014.
[19] EPCglobal, Core Business Vocabulary Standard 1.1, GS1, May,

2014.
[20] E.Y. Song and K. Lee, "Understanding IEEE 1451–networked smart transducer interface standard – what is a smart transducer?" IEEE Instrumentation and Measurement Magazine, vol.11, no.2, pp.11–17, 2008.
[21] IEEE Std 1451.1 - IEEE Standard for a Smart Transducer Interface for Sensors and Actuators- Network Capable Application Processor (NCAP) Information Model, IEEE Instrumentation and Measurement Society, 1999.
[22] IEEE Std 1451.0 - IEEE Standard for a Smart Transducer Interface for Sensors and Actuators - Common Functions, Communication Protocols, and Transducer Electronic Data Sheet (TEDS) Formats, IEEE Instrumentation and Measurement Society, 2007.
[23] IEEE Std 1451.7 - IEEE Standard for Smart Transducer Interface for Sensors and Actuators - Transducers to Radio Frequency Identification (RFID) Systems Communication Protocols and Transducer Electronic Data Sheet Formats, IEEE Instrumentation and Measurement Society, 2010.
[24] C.-W. Tseng and C.-H. Huang, "Toward a Consistent Expression of Things on EPCglobal Architecture Framework," Proc. International Conference on Information Science, Electronics and Electrical Engineering, April 2014, pp.1619–1623.
[25] C.-W. Tseng and C.-H. Huang, "A Uniform EPC Scheme Design of IEEE 1451 Transducers for IoT Applications," Proc. Sixth International Conference on Ubiquitous and Future Networks, July 2014, pp.255–260.
[26] Y.-C. Chen, L.-C. Chung, Y.-U. Yang, H.-R. Chen, Y.-H. Chen, and C.-H. Huang, "Design and implementation of an ALE middleware," Proc. 2010 Conference on Information Technology and Industrial Application, 2010.

Chao-Wen Tseng received the B.S. degree in engineering science from National Cheng Kung University, Tainan, Taiwan, in 1989, and the M.S. degree in information engineering and computer science from National Cheng Kung University, Tainan, Taiwan, in 1991. Tseng is currently pursuing the Ph.D. degree. His research interests include software engineering, object-oriented technology. internet of things and industry 4.0.

Yu-Chang Chen received the B.S. degree in computer science from Feng Chia University, Taichung, Taiwan, in 1980 and the M.S. degree in computer science from Feng Chia University, Taichung, Taiwan, in 1997. Taiwan. Chen is currently pursuing the Ph.D. degree. His research interests include software engineering, RFID and internet of things.

Chua-Huang Huang received the B.S. degree in mathematics from Fu-Jen University, Taipei, Taiwan, in 1974, the M.S. degree in computer science from University of Oregon, Eugene, Oregon, in 1979, and the Ph.D. degree in computer science from the University of Texas at Austin, Austin, Texas, in 1987. He was an assistant professor in the Department of Computer and Information Science, the Ohio State University, from 1987 to 1993 and an associate professor from 1993 to 1997. From 1997 to 2000, Dr. Huang was a professor at the Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan. Since 2000, Dr. Huang has been a professor at the Department of Information and Engineering, Feng Chia University, Taichung, Taiwan. Dr. Huang's research interests include software methodology, RFID, and internet of things.