|  |
| --- |
|  |
| **Architectural Drivers Specification**   * IoT ecosystem -   https://lh5.googleusercontent.com/d9waw423S9h55LnkeNBrtDPCF3Ws1hTpg2zJW9GYIB_fb86X1xlam67OmKmnzmTksWQTaNhug5vNJwqIbghgs5KdpP95p0I9a6sOI7GCFwHfpuiIuPwtnrFt_GTix17jXJft3ZcC |
| Eagle Five (Team number 1) |

**REVISION HISTORY**

|  |  |  |
| --- | --- | --- |
| Ver. | Date | Description |
| 0.1 | 05/11, 2015 | Create initial ADS.  Add Context, Use cases, Quality attributes, Constraints. |
| 0.2 | 05/14, 2015 | Mentor’s comments |
| 0.3 | 05/18, 2015 | Add detail Quality attribute scenarios |
| 0.4 | 05/21, 2015 | Add functional requirements.  Refine Use cases and Quality attribute scenarios. |
| 0.5 | 05/26, 2015 | Shared on Google docs. |
| 0.6 | 05/27, 2015 | Refine QA’s and set priority. |
| 0.7 | 05/31, 2015 | Add FR10 “Give SA node access permission”.  Add UC18 “Sharing SA node to other user”.  Add UC19 “Transfer register/sharing authority”.  Add QA02 “SA node hijack” scenario.  Add QA04 “Internet cut off” scenario.  Modify QA05, this QA is not about ‘Performance’ but ‘Scailability’. |
|  |  |  |
|  |  |  |

**Table of Contents**

[**1.Introduction**](#h.30j0zll)

[1.1. Project Goals](#h.1fob9te)

[1.2. Stakeholders](#h.3znysh7)

[**2.High Level Functional Requirements**](#h.tyjcwt)

[2.1. Functional Requirements](#h.3dy6vkm)

[2.2. Use Case Scenario](#h.1t3h5sf)

[**3.** **Quality Attribute**](#h.4d34og8)

[3.1. Summary](#h.2s8eyo1)

[3.2. Quality Attribute Scenarios](#h.17dp8vu)

[**4.** **Constraints**](#h.26in1rg)

# **1. Introduction**

The aim of this document is to describe the architectural drivers of the IoT project: high level functionality in the form of use cases, quality attributes scenarios and constraints.

## Project Goals

* Create an IoT infrastructure to support accessing sensors and actuators installed in the home or business.
* Create an infrastructure to provide an ecosystem to develop cost competitive home or business IoT products for value-added-resellers and other 3rd party hardware and software application developers, service providers, and installers and maintainers.
* Build a basic data centric infrastructure to provide IoT data sets for developers to create future data mining, analytic operations, and services.

## Stakeholders

* Consumers.
* Sensor/Actuator producers.
* Home builders.
* Third-party service providers.
* IoT application developers.
* Anthony J. Latanze & his development team.

# **2.High Level Functional Requirements**

## Functional Requirements

|  |  |
| --- | --- |
| **Access secured services** | FR01 |
| **Description**:  User accesses the system in secured environment. User must login to the system for services. Unauthorized persons are not allowed to control sensors installed in home, register SA Nodes, or access any data gathered in the system. | |

|  |  |
| --- | --- |
| **Discover SA Nodes** | FR02 |
| **Description**:  User queries home to find out how many nodes are installed and what sensors/actuators are installed on each node. | |

|  |  |
| --- | --- |
| **Determine sensors and control actuators** | FR03 |
| **Description**:  User can determine the temperature/humidity, turn on and off lights, open and close the door, turn on the alarm, and determine if anyone is home. However, user must set the alarm off prior to opening the door. | |

|  |  |
| --- | --- |
| **Log user commands and sensor values** | FR04 |
| **Description**:  User commands and sensor values are stored in IoT infrastructure for some period of time. This data set can be utilized by developers to create future data mining, analytic operations, and services. | |

|  |  |
| --- | --- |
| **Send emergency message** | FR05 |
| **Description**:  An emergency message is sent to the user when door is opened manually or the house is suddenly occupied while alarm is set. | |

|  |  |
| --- | --- |
| **Lock house automatically** | FR06 |
| **Description**:  User is informed upon the vacancy of house and asked to lock the house. If the home is vacancy for 30 seconds and alarm is not set. SA node notifies user and if the user failed to respond to the message within 5 minutes, the door is closed, and the alarm is set automatically. | |

|  |  |
| --- | --- |
| **Turn off light automatically** | FR07 |
| **Description**:  When no one is home for 10 minutes, the light is turned off automatically. | |

|  |  |
| --- | --- |
| **Register SA node** | FR08 |
| **Description**:  Authorized user adds nodes to the system. Equipped sensors and actuators are recognized. | |

|  |  |
| --- | --- |
| **Unregister SA node** | FR09 |
| **Description**:  Authorized user removes nodes from the system. | |

|  |  |
| --- | --- |
| **Grant SA node access permission** | FR10 |
| **Description:**  User who registered a SA node is the owner of the node. He/she grants an access permission of the node to others. However, only owner of the node can unregister the node. | |

|  |  |
| --- | --- |
| **Transfer SA node ownership** | FR11 |
| **Description:**  User who registered a SA node is the owner of the node. He/she can give up the ownership of the node, and transfer the ownership to other user. The user who gives up the ownership still has the access permission of the node. | |

## Use Case Scenario

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Add new user account | | **Use Case ID**:  01 |
| **General use case description**:  This use case describes how user adds a new account to the system. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**: | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to add a new user account. | |
| 2. | E02 prompts E01 to enter registration information including id and password. | |
| 3. | E01 elects to submit the registration information. | |
| 4. | E02 requests a new account registration with information to E03. | |
| 5. | E03 validates information, adds the temporal account, and sends the account activation mail to E01. | |
| 6. | E03 sends the confirmation to E02. | |
| 7. | E02 presents the guide to activate the account to E01. | |
| 8. | E01 activates the account by following the instruction described in the account activation mail. | |
| 9. | E03 activates the account. | |
| **Alternative Flows:**  **\* Invalid account registration information**  5.1. If the registration information is invalid, the request is rejected. E03 sends error message to E02. Proceed to step 7 of the primary use case.  **\* Account activation timeout**  8.1 If E03 doesn’t receive the activation request for a specific amount of time, the account registration is canceled. | | |
| **Postconditions**:  A new user account was regitered. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Login | | **Use Case ID**:  02 |
| **General use case description**:  This use case describes how user logs into the system. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  User account is registered to the system | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to login to the system. | |
| 2. | E02 prompts E01 for id and password. | |
| 3. | E01 enters id and password. | |
| 4. | E02 requests the login with id and password to E03. | |
| 5. | E03 validates the entered id and password and logs E02 into the system. | |
| 6. | E03 sends the confirmation to E02. | |
| 7. | E02 presents the results to E01. | |
| **Alternative Flows:**  **\* Invalid id and/or password**  5.1. If id and/or password is invalid E03 rejects E02. E03 sends an error message to E02. Proceed to step 7 of the primary use case. | | |
| **Postconditions**:  User was logged into the system. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Register new SA node | | **Use Case ID**:  03 |
| **General use case description**:  This use case describes how SA node is registered to the system. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to register E04. | |
| 2. | E02 prompts E01 for serial number of the node. | |
| 3. | E01 enters the serial number. | |
| 4. | E02 requests a new SA node registration with the serial number to E03. | |
| 5. | E03 validates the serial number, and waits for the connection request of E04. | |
| 6. | E03 sends confirmation of the temporal registration to E02. | |
| 7. | E02 guides E01 to manipulate E04 to establish a connection with E03. | |
| 8. | E01 manipulates E04 for the connection between E04 and E03. | |
| 9. | Proceed to step 1 of primary use case of UC05. | |
| 10. | E03 adds E04 to the user account. | |
| 11. | E03 sends the confirmation to E02. | |
| 12. | E02 presents the result to E01. | |
| **Alternative Flows:**  **\* Invalid serial number**  6.1. If the serial number is already registered, temporally registered, or invalid, E03 sends error message to E02. E02 presents the result to E01. The use case ends.  **\* Connection request timeout**  8.1. If E03 doesn’t receive the connection request from E04 for a specifed amount of time, E03 revokes the temporal registration, and sends error message to E02. E01 presents the result to E01. The use case ends. | | |
| **Postconditions**:  A new node is registered to the system | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Unregister SA node | | **Use Case ID**:  04 |
| **General use case description**:  This use case describes how SA node is unregistered to the system. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to unregister E04. | |
| 2. | E02 prompts E01 to select a node to unregister. | |
| 3. | E01 selects the node. | |
| 4. | E02 requests the unregistration of the node to E03. | |
| 5. | E03 unregisters the node, and disconnects it if connected. | |
| 6. | E03 sends the confirmation to E02. | |
| 7. | E02 presents the result to E01. | |
| **Postconditions**:  SA node was unregstered. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Request SA node connection | | **Use Case ID**:  05 |
| **General use case description**:  This use case describes how SA node establishes connection to the system. | | |
| **Entities involved**:  E03 – System  E04 – SA Node | | |
| **Preconditions**: | | |
| **Primary use case flow of events**: | | |
| 1. | E04 attemps to connect to E03. | |
| 2. | E03 grants the connection request | |
| **Alternative Flows:**  **\* Invalid connection request**  2.1. If E04 is not registered to E03, E03 rejects the request. The use case ends. | | |
| **Postconditions**:  The node was connected to the system. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Discover SA Nodes | | **Use Case ID**:  06 |
| **General use case description**:  This use case describes how user discovers installed SA nodes, and equipped sensors and actuators. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  Nodes are registered to the system.  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to discover SA nodes. | |
| 2. | E02 requests SA node information to E03. | |
| 3. | E03 sends the nodes information including the list of equipped sensors/actuators to E02. | |
| 4. | E02 presents the information to E01. | |
| **Postconditions**:  Node information were presented. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Determine presence/proximity | | **Use Case ID**:  07 |
| **General use case description**:  This use case describes how user determines presence/proximity sensor. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to determine presence/proximity. | |
| 2. | E02 requests presence/proximity sensor value to E03. | |
| 3. | E03 requests the value to E04. | |
| 4. | E04 reads the sensor, and sends the value to E03. | |
| 5. | E03 sends the value to E02. | |
| 6. | E02 presents the value to E01. | |
| **Postconditions**:  The value of the presence/proximity sensor was presented. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Determine temperature/humidity | | **Use Case ID**:  08 |
| **General use case description**:  This use case describes how user determines temperature/humidity. | | |
| **Entities involved**:  E01 – User  E02 - End user scenario  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to determine temperature/humidity. | |
| 2. | E02 requests temperature/humidity sensor value to E03. | |
| 3. | E03 requests the value to E04. | |
| 4. | E04 reads the sensor, and sends the value to E03. | |
| 5. | E03 sends the value to E02. | |
| 6. | E02 presents the value to E01. | |
| **Postconditions**:  The value of the temperature/humidity sensor was presented. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Control door light | | **Use Case ID**:  09 |
| **General use case description**:  This use case describes how user turns on or off the light. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to control the light. | |
| 2. | E02 requests light control to E03. | |
| 3. | E03 requests to E04. | |
| 4. | E04 controls the light, and sends the status of light to E03. | |
| 5. | E03 sends the status to E02. | |
| 6. | E02 presents the status of light to E01. | |
| **Post conditions**:  The light was turned on or off as user requested. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Control door | | **Use Case ID**:  10 |
| **General use case description**:  This use case describes how user opens or closes the door. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to control the door. | |
| 2. | E02 requests door control to E03. | |
| 3. | E03 requests to E04. | |
| 4. | E04 controls the door. | |
| 5. | E04 sends the status of door to E03. | |
| 6. | E03 sends the status to E02. | |
| 7. | E02 presents the status of ddor to E01. | |
| **Alternative Flows**  **Alarmed**  4-1. If the alarm is set, step 4 is skipped and proceed to step 5 of the “Primary use case flow”. | | |
| **Postconditions**:  The door was opened or closed as user requested if the alarm was not set. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Control alarm | | **Use Case ID**:  11 |
| **General use case description**:  This use case describes how user sets the alarm on or off. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to set the alarm. | |
| 2. | E02 requests alarm setting to E03. | |
| 3. | E03 requests to E04. | |
| 4. | E04 sets the alarm, and sends the status of the alarm to E03. | |
| 5. | E03 sends the status to E02. | |
| 6. | E02 presents the status of the alarm to E01. | |
| **Postconditions**:  The alarm was set or unset as user requested. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Log user commands. | | **Use Case ID**:  12 |
| **General use case description**:  This use case describes how user commands are stored in IoT infrastructure | | |
| **Entities involved**:  E02 – End users application  E03 – System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E02 sends command to E03. | |
| 2. | E03 stores the command. | |
| **Postconditions**:  User command logs were stored. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Log sensor values | | **Use Case ID**:  13 |
| **General use case description**:  This use case describes how sensor values are stored in IoT infrastructure | | |
| **Entities involved**:  E03 – System  E04 – SA Node | | |
| **Preconditions**:  SA Node is registered to the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E04 reports sensor value to E03 when changed | |
| 2. | E03 stores reported sensor value. | |
| **Postconditions**:  Sensor values were stored. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Configure log time window | | **Use Case ID**:  14 |
| **General use case description**:  This use case describes how log time window is set. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to set log time window. | |
| 2. | E02 requests E03 for the current configuration. | |
| 3. | E03 sends the configurations to E02. | |
| 4. | E02 presents the configuration to E01. | |
| 5. | E01 enters the new configuration, and elects to update the configuration. | |
| 6. | E02 requests configuration updates to E03. | |
| 7. | E03 applies configurations. | |
| 8. | Repeat 2 ~ 7 as necessary | |
| **Post conditions**:  New configurations were applied. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Review logs | | **Use Case ID**:  15 |
| **General use case description**:  This use case describes how user reviews sensor state and user command history. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to review the logs. | |
| 2. | E02 requests logs to E03. | |
| 3. | E03 sends logs of specified time window to E02. | |
| 4. | E02 presents logs to E01. | |
| **Post conditions**:  User command and sensor state logs were presented to the user. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Send emergency message | | **Use Case ID**:  16 |
| **General use case description**:  This use case describes how an emergency message is sent to user when the door is opened manually or the house is occupied suddenly while the alarm is set. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  The alarm is set. | | |
| **Primary use case flow of events**: | | |
| 1. | E04 detects door opening or detects the presence of moving object(s). | |
| 2. | E04 sends the emergency message to E03. | |
| 3. | E03 sends the message to E02. | |
| 4. | E02 presents the emergency message to E01. | |
| **Postconditions**:  Emergency message was presented to the user. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Lock house automatically | | **Use Case ID**:  17 |
| **General use case description**:  This use case describes how the house is locked automatically. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System  E04 – SA Node | | |
| **Preconditions**:  The alarm is not set. | | |
| **Primary use case flow of events**: | | |
| 1. | E04 detects the vacancy of the house for 5 minutes. | |
| 2. | E04 sends inform message to E03. | |
| 3. | E03 sends the message to E02. | |
| 4. | E02 presents the message to E01. | |
| 5. | E01 elects to lock the house. | |
| 6. | E02 requests E03 to lock the house. | |
| 7. | E03 requests to E04. | |
| 8. | E04 closes door and sets the alarm. | |
| **Alternative Flows:**  **\* Not to lock the house**  5.1. If E01 elects not to lock the house, E02 requests E03, and E03 requests E04 not to lock the house. The use case ends.  **\* Response timeout**  5.1. If E04 does not receive the response from E01, proceed to step 8 of the “Primary use case flow”. | | |
| **Postconditions**:  The door was closed and the alarm is set. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Turn off light automatically | | **Use Case ID**:  18 |
| **General use case description**:  This use case describes how the light is turned off automatically. | | |
| **Entities involved**:  E04 – SA Node | | |
| **Preconditions**:  Light is on. | | |
| **Primary use case flow of events**: | | |
| 1. | E04 detects the vacancy of the house, and starts timer for 10 minutes. | |
| 2. | E04 turns off lights when timer is expired | |
| **Alternative Flows:**  **\* Not to lock the house**  1.1. If E04 detects the light being turn off, or presents of people, the timer is canceled. The user case ends. | | |
| **Postconditions**:  Lights were turned off | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Grant SA node access permission | | **Use Case ID**:  19 |
| **General use case description**:  This use case describes how the owner of the SA node grants a node access permission to other user account. | | |
| **Entities involved**:  E01 – User  E02 – End user application  E03 - System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | E01 elects to grant a node access permission to other user. | |
| 2. | E02 requests a list of owning nodes to E03. | |
| 3. | E03 sends the list. | |
| 4. | E02 requests a list of “friend” account to E03. | |
| 5. | E03 sends a list of “friend” account to E02. | |
| 6. | E02 presents the list of nodes and account. | |
| 7. | E01 selects a node and an account. | |
| 8. | E02 requests a node access permission assignment to E03. | |
| 9. | E03 updates the account, and sends a confirmation to E02. | |
| **Postconditions**:  An access permission of the seleted node was assigned to the selected account. | | |

|  |  |  |
| --- | --- | --- |
| **Use case title**:  Transfer SA node ownership | | **Use Case ID**:  20 |
| **General use case description**:  This use case describes how the owner of the SA node transfer the ownership to other user account. | | |
| **Entities involved**:  E01 – User  E02 - End user application  E03 – System | | |
| **Preconditions**:  User is logged into the system. | | |
| **Primary use case flow of events**: | | |
| 1. | User elects to transfer the node ownership to other user account. | |
| 2. | E02 requests a list of owning nodes to E03. | |
| 3. | E03 sends the list. | |
| 4. | E02 requests a list of “friend” account to E03. | |
| 5. | E03 sends a list of “friend” account to E02. | |
| 6. | E02 presents the list of nodes and account. | |
| 7. | E01 selects a node and an account. | |
| 8. | E02 requests the ownership transfer to E03. | |
| 9. | E03 updates accounts, and sends a confirmation to E02. | |
| **Postconditions**:  An ownership of the seleted node was assigned to the selected account.  User lost the ownership of the seleted node, but kept an access permission of the node. | | |

# **3.** **Quality Attribute**

## 3.1. Summary

|  |  |  |  |
| --- | --- | --- | --- |
| **Total** | **High Priority** | **Medium Priority** | **Low Priority** |
| 9 | 5 | 4 | 0 |

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Priority** | **Quality Attribute** | **Descriptions** |
| QA01 | H | Security | Hackers or ill minded people try to break into the system. When unauthorized user attempts to login to the system, the system maintains the audit trail. If the attempt is repeated more than 5 times, the account is locked, and the source of tempering is identified. |
| QA02 | H | Security | Hackers or ill minded people try to register the SA node that is not owned by them. When unauthorized user attempts to register the SA node that he/she doesn’t own, the system maintains the audit trail, and cancel the registration in 10 minutes. |
| QA03 | H | Availability | SA node can crash, hang, or be disconnected from the network for various reasons. If SA node is inoperable or out of reach, the system should be aware of such events, and notify user within 2 minutes. |
| QA04 | M | Availability | SA node can be disconnected from the network for various reasons. If SA node is not able to reach the system due to network failure, it should store recent logs at least for one day. When the network is restored, SA node should send the logs to the system. |
| QA05 | M | Scalability | The number of SA node user can be more than one. The system should be able to serve 10 user controls to the same SA node. (Concurrent access and control is not considerred in this scenario) |
| QA06 | M | Scalability | More than one SA node can be installed at home. The system should be able to support at least 100 nodes concurrently. |
| QA07 | H | Modifiability | The system should make it easy to add emerging protocols (eg. Bluetooth 802.15, ZigBee 802.15.4) to the system. Average skilled developers should be able to implement it within two months. |
| QA08 | M | Usability | The system should make it easy for users to register or unregister SA nodes. Ordinary user should be able to register or unregister the node within 5 minutes by following the provided mannual. |
| QA09 | H | Extensibility | The system should make it easy for application developers (private persons, VARs, or other 3rd parties) to build custom apps, services, and/or make mashups from existing available services. Average skilled developers should be able to build the application in six months. |

## 3.2. Quality Attribute Scenarios

|  |  |  |
| --- | --- | --- |
| **Security** | | **ID:** QA01 |
| **Quality Attribute:**  Hackers or ill minded people try to break into the system. When unauthorized user attempts to login to the system, the system maintains the audit trail. If the attempt is repeated more than 5 times, the account is locked, and the source of tempering is identified. | | |
| Stimulus | Login attempt with an Incorrect id or a password | |
| Source(s) of the stimulus | Human or machine | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System | |
| System response | The system bans any further access, and logs all access attempts. | |
| Response measure(s) | Five repeated attempts locks the account, and the source of tempering is identified. | |

## 

|  |  |  |
| --- | --- | --- |
| **Security** | | **ID:** QA02 |
| **Quality Attribute:** Hackers or ill minded people try to register the SA node that is not owned by them. When unauthorized user attempts to register the SA node that he/she doesn’t own, the system maintains the audit trail, and cancel the registration in 10 minutes. | | |
| Stimulus | Unauthorized SA node registration | |
| Source(s) of the stimulus | Human or machine | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System | |
| System response | The system prevents the registration of SA node of which onership is not identified. | |
| Response measure(s) | Unidenfied SA node registration is canceled in 10 minutes. | |

|  |  |  |
| --- | --- | --- |
| **Availability** | | **ID:** QA03 |
| **Quality Attribute:** SA node can crash, hang, or be disconnected from the network for various reasons. If SA node is inoperable or out of reach, the system should be aware of such events, and notify user within 2 minutes. | | |
| Stimulus | SA node failure | |
| Source(s) of the stimulus | SA node | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System, SA node | |
| System response | The system monitors and detects the failure of SA node. | |
| Response measure(s) | System notifies failure to user within 2 minutes. | |

|  |  |  |
| --- | --- | --- |
| **Availability** | | **ID:** QA04 |
| **Quality Attribute:** SA node can be disconnected from the network for various reasons. If SA node is not able to reach the system due to network failure, it should store recent logs at least for one day. When the network is restored, SA node should send the logs to the system. | | |
| Stimulus | Network failure between SA node and the system | |
| Source(s) of the stimulus | Network environment | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System, SA node | |
| System response | SA node stores recent logs and send them to the system whenever the system connection is available. | |
| Response measure(s) | SA node should be available to store logs at least for one day. | |

|  |  |  |
| --- | --- | --- |
| **Scalability** | | **ID:** QA05 |
| **Quality Attribute:** The number of SA node user can be more than one. The system should be able to serve 10 user controls to the same SA node. (Concurrent access and control is not considerred in this scenario) | | |
| Stimulus | Multiple user access | |
| Source(s) of the stimulus | Human | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System | |
| System response | The system services the user controls to the same SA node. | |
| Response measure(s) | The system should allow at least 10 user controls. | |

|  |  |  |
| --- | --- | --- |
| **Scalability** | | **ID:** QA06 |
| **Quality Attribute:** More than one SA node can be installed at home. The system should be able to support at least 100 nodes concurrently. | | |
| Stimulus | Concurrent inbound or outbound SA node messages | |
| Source(s) of the stimulus | User, SA node | |
| Relevant environmental conditions | Normal Operation | |
| Architectural elements | System | |
| System response | The system supports services for multiple SA nodes concurrently. | |
| Response measure(s) | At least 100 nodes should be able be served at the same time. | |

|  |  |  |
| --- | --- | --- |
| **Modifiability** | | **ID:** QA07 |
| **Quality Attribute:** The system should make it easy to add emerging SA node protocols (eg. Bluetooth 802.15, ZigBee 802.15.4) to the system. Average skilled developers should be able to implement it within two months. | | |
| Stimulus | New protocols for SA node | |
| Source(s) of the stimulus | Developer | |
| Relevant environmental conditions | After release | |
| Architectural elements | System, SA node | |
| System response | New protocols is supported by the system and SA node. | |
| Response measure(s) | Average skilled developers can implement it within two months. | |

|  |  |  |
| --- | --- | --- |
| **Usability** | | **ID:** QA08 |
| **Quality Attribute:** The system should make it easy for users to register or unregister SA nodes. Ordinary user should be able to register or unregister the node within 5 minutes by following the provided mannual. | | |
| Stimulus | Add/Remove a new node to the system | |
| Source(s) of the stimulus | User | |
| Relevant environmental conditions | Normal operation | |
| Architectural elements | System, SA node | |
| System response | The system registers or unregisters SA nodes, and the  system nor the other SA nodes are not rebooted. | |
| Response measure(s) | User should be able to register or unregister the node within 5 minutes by following the provided mannual. | |

|  |  |  |
| --- | --- | --- |
| **Extensibility** | | **ID:** QA09 |
| **Quality Attribute:** The system should make it easy for application developers (private persons, VARs, or other 3rd parties) to build custom apps, services, and/or make mashups from existing available services. Average skilled developers should be able to build the application in six months. | | |
| Stimulus | New application or service | |
| Source(s) of the stimulus | Developers (including 3rd party), VARs | |
| Relevant environmental conditions | After release | |
| Architectural elements | System | |
| System response | The system suppots the new application/service. | |
| Response measure(s) | Average skilled developers should be able to build the application in six months. | |

# **4.** **Constraints**

|  |  |  |
| --- | --- | --- |
| **ID** | **Type** | **Description** |
| TC01 | Technical | JAVA compiler, Arduino 1.0.6 is preferred. |
| TC02 | Technical | Permissible languages for this system (excluding the SA Nodes) include JAVA and Python. |
| TC03 | Technical | 802.11 is only supported in the system. |
| TB01 | Business | Development period: 5 weeks (3 hours/day) |
| TB02 | Business | Development team: 5 developers. |