ALLSEEN ALLIANCE

Technical Steering Meeting

August 03, 2015

Antitrust Compliance Notice

- AllSeen Alliance meetings involve participation by industry competitors, and it is the intention of AllSeen Alliance to conduct all of its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of and not participate in any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.

- Examples of types of actions that are prohibited at AllSeen Alliance meetings and in connection with AllSeen Alliance activities are described in the AllSeen Alliance Antitrust Policy. If you have questions about these matters, please contact your company counsel, or if you are a member of AllSeen Alliance, feel free to contact Lee Gesmer or Andrew Updegrove, of the firm of Gesmer Updegrove LLP, which provides legal counsel to AllSeen Alliance.

Reminder:

This call is being recorded

1. Approve minutes from previous meeting
2. AllJoyn Ambassador Program
3. AJATS and Release Testing RFPs
4. qcc::String to std::string migration
5. Provisioning project proposal
   – Introduction
   – Scope
   – Dependencies
   – Project Working
   – Timeline

AllJoyn Ambassador Program

AllJoyn Ambassadors Program

- Overview
  - The goal of the Ambassadors program is to sustain and grow the worldwide AllJoyn community.
  - Ambassadors are enthusiastic experts who know and use the code and who are willing to share that knowledge with others around the world, with the project providing support and resources to enable that.
- Ambassador Profile
  - Someone who is passionate about AllJoyn and recognized for their expertise and willingness to help others learn about the framework and our community. Usually hands-on practitioners. Someone who has the characteristics of being helpful, hopeful and humble. People like bloggers, influencers, evangelists who are already engaged with the project in some way. Contributing to forums, online groups, community, etc.
  - Ambassadors will selecting by the TSC from those who apply with a small number of AllJoyn Ambassadors selected to be the primary point of contact in a particular region.
    - Their role will be to:
      - Be a go-to resource for people interested in AllJoyn
      - Help local users learn more about AllJoyn
        - Organize and host local AllJoyn Users Group meetups (with resources from the AllSeen Alliance) (See the OLD User Groups HERE)
      - Represent the community publicly

Interested in representing the AllSeen Alliance as an AllJoyn Ambassador?

Apply now!

info@allseenalliance.org

Applicant: AllJoyn Ambassador Program

1. Name: Jun Zhang

2. Email: zhangjun@haierubic.com

3. Member Company: Haier

4. Where do you live? (There are certain countries where we're seeing a lot of interest so it helps to know where you're located): China

5. Why are you applying to be an AllJoyn Ambassador?

I heard from some China members about the barriers from language to technical support to AllSeen engagement. There are also non-members interested in AllJoyn and want to learn more about AllJoyn. I would like to help match those members and non-members with support they need.

6. How have you participated in the AllJoyn community to-date?

I am representing Haier in TSC and act as committer of the Home Controller Project.

7. What ideas do you have for your community that you wish you had time or resources to implement?

The initial ideas are to help members connect with the information they need, promote AllJoyn to non-members, coordinate meet-ups in China to share AllJoyn development information.

8. How will you work with others to achieve your goals?

I will collaborate with the community to provide AllJoyn progress and technical resources to local group, introduce AllJoyn to potential members, help new members get started, and work with the local group to coordinate meet-ups.

9. LinkedIn profile page (if available): none

10. Twitter handle (if available): none

AJATS and Release Testing RFPs

AJATS and Release Testing RFPs

- AllJoyn Automated Test System RFP
  - Vendor will reproduce system currently operated by QCE
  - System will integrate with Jenkins build system
  - Results will be available to all AllSeen Alliance members
  - Primary task is to host the system and provide maintenance and diagnostic support
  - AllSeen Alliance members will participate in developing new test cases and expanding or modifying the design
- AllJoyn Release Testing RFP
  - Vendor will execute release testing of AllJoyn Core and Base Services
  - Test cases are being migrated to AllSeen Alliance TestLink
  - Test applications are available in AllSeen Alliance repositories
  - Vendor will be trained by AllSeen Alliance
  - Expect vendor to be ready to conduct testing for AllJoyn v16.04

AJATS and Release Testing RFPs

- Proposed schedule
  - July 31: RFP sent to TSC members mail list for review
  - Aug 3: The RFP is presented to the TSC at the Monday night call for review
  - Aug 17: The RFP is voted on by the TSC
    - If approved, the RFP is locked and published
      - All responses should be sent to rfp@allseenalliance.org
    - The 2 week clock for responding begins
  - Sept 01: TSC updated on process
  - Sept 14: TSC updated on process
    - Time to respond is closed
      - All submitted proposals are sent to the special mail list for review
        - » A special email list will be created for TSC voting members that did not submit a proposal
  - Sept 22: Two-hour TSC meeting to review and discuss proposals
  - Sept 28: Vote is called and vendor chosen

qcc::String to std::string migration

qcc::String to std::string migration- Summary

- For 16.04:
  - Deprecate qcc::String::secure_clear()
  - Remove the internal usage of this function and develop a suitable alternative that external users can switch to. – Modify qcc::String to be a wrapper around std::string
  - This would include introducing functionality to support mixing qcc::String and std::string. Hopefully a type conversion operator to convert qcc::String to std::string will solve most of the mixing problems.
  - The deprecated version of qcc::String::secure_clear() will need to use const_cast<> to cast away const from the point obtained from std::string::data() so that it can wipe the memory before calling string::clear(). Hopefully, compilers won't warn about casting const away.
  - Document that std::string can be trivially converted to qcc::String and vice versa without an underlying strcpy().
- For 18.04 (4 releases after 16.04):
  - Remove qcc::String::secure_clear()
  - Change qcc::String to just be a typedef of std::string
  - Change APIs that use qcc::String to take std::string instead

qcc::String to std::string migration- differences in interfaces and behavior (1/2)

- secure_clear() - no equivalent in std::string.
- Many qcc::String constructors take a size hint that is not part of std::string constructors.
- qcc::String provided a version of append() that could take a single char as its only parameter. std::string does not provide a version of append() that takes only a char. Calls to this method will need to be changed to use append(size_t count, char c) and pass in 1 for the count parameter.
- qcc::String iterators were just typedefs of char*. Found one place where qcc::String::iterator was used in pointer arithmetic.
- find_last_of() behaves differently between qcc::String and std::string. Given the string "abcdefgdijk", find_last_of('d', 7) will return 3 for qcc::String and 7 for std::string. (Note that position 7 is the second "d" in the string.) This is believed to be a bug in the qcc::String implementation. Need to examine the places where this function is used to make sure nothing breaks.

qcc::String to std::string migration- differences in interfaces and behavior (2/2)

- Calling erase() with the start position greater than the size will cause an out of range exception for std::string but will be silently ignored for qcc::String.
- Calling substr() with an out of bounds index will result in an empty string from qcc::String but throw an exception for std::string.
- qcc::String can be instantiated with a NULL const char*. std::string will throw an exception in that case.
- This was I found building only the C++ binding for Linux. Other platforms and bindings will need to be built/tested as well, but there should not be too many more differences.
- Approvals
  - Core WG approved on 7/29/15
  - Call for TSC vote

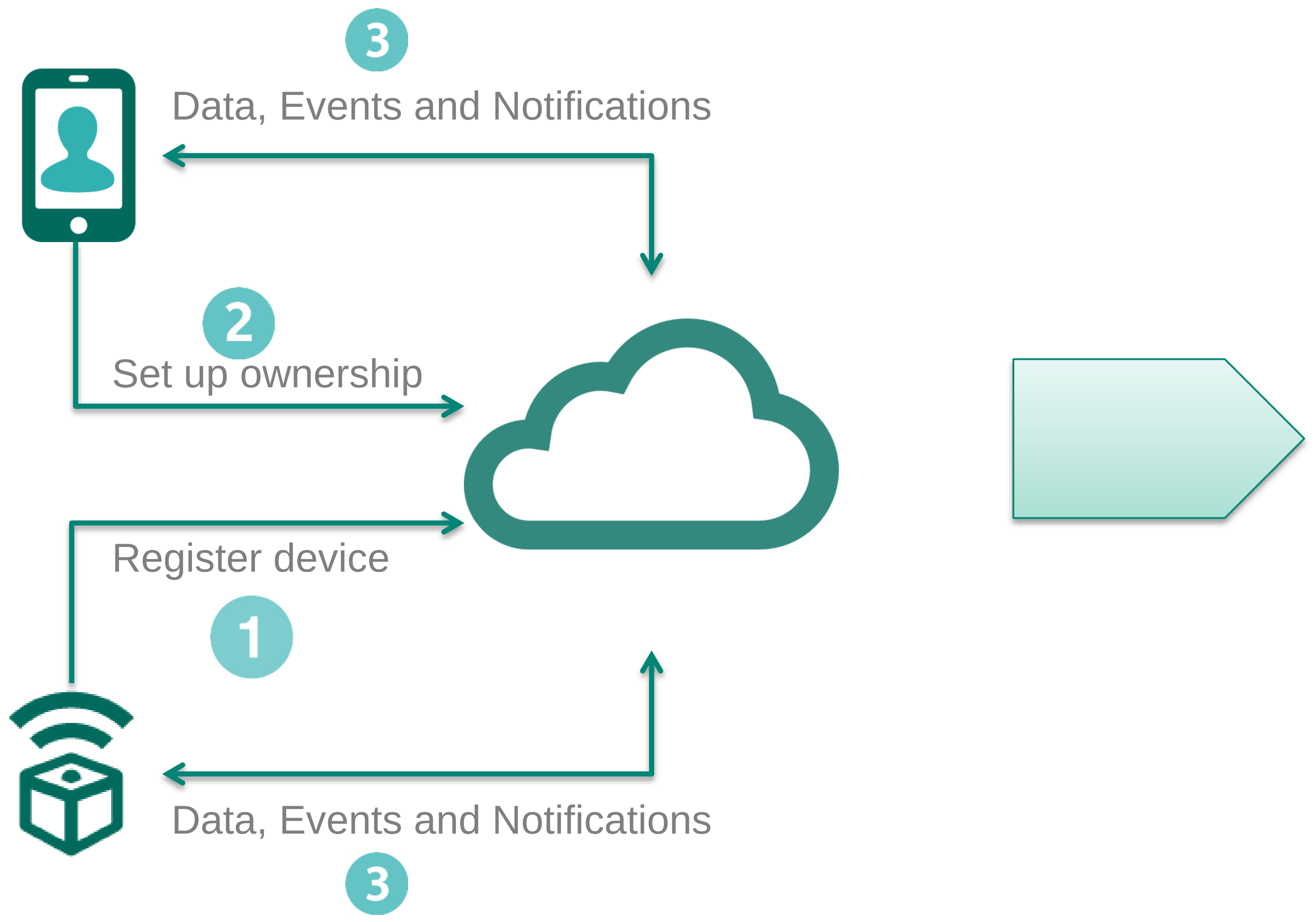
Provisioning Project Proposal

Phani Pandrangi

Kii

Provisioning
Introduction

Typical Working – Things & Cloud


3
Data, Events and Notifications
2
Set up ownership
Register device
1
Data, Events and Notifications
3

Telemetry Data Storage
Store transactional and sensory data, events

Analytics
Analyze data and events for gaining insights or other purposes

Upgrades
Push firmware and configuration upgrades to devices

Warranties
Enforce warranties and other such services

Diagnostics
Remotely diagnose problems and improve products and services

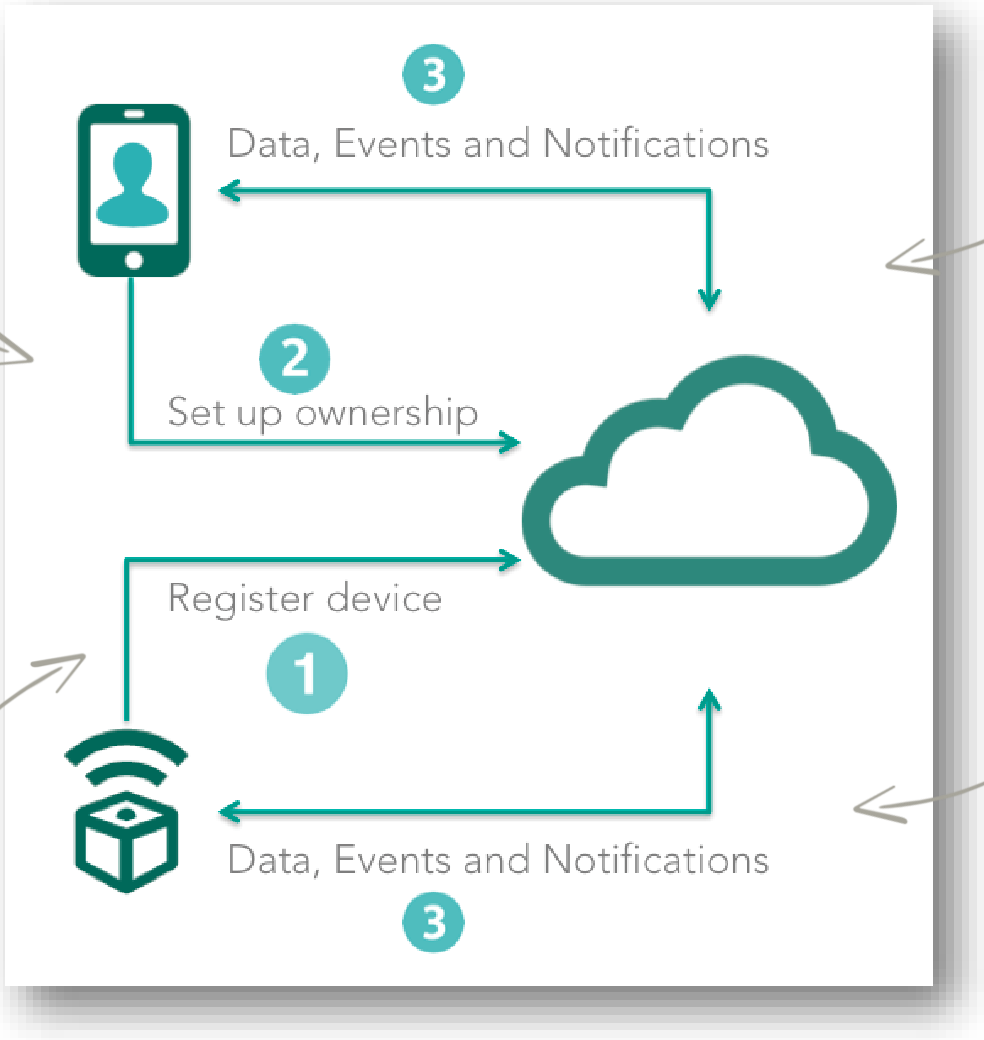
Control & Interoperate
Control devices from cloud and integrate with other services

AllJoyn Interfaces In This Process

AllJoyn Security
User claims device thru security and establishes cloud identity

AllJoyn About Feature
AboutData can be used when registering device in the cloud

AllJoyn Onboarding Service
When the signal indicating connection completed is received, start cloud onboarding


3
Data, Events and Notifications
2
Set up ownership
Register device
1
Data, Events and Notifications
3

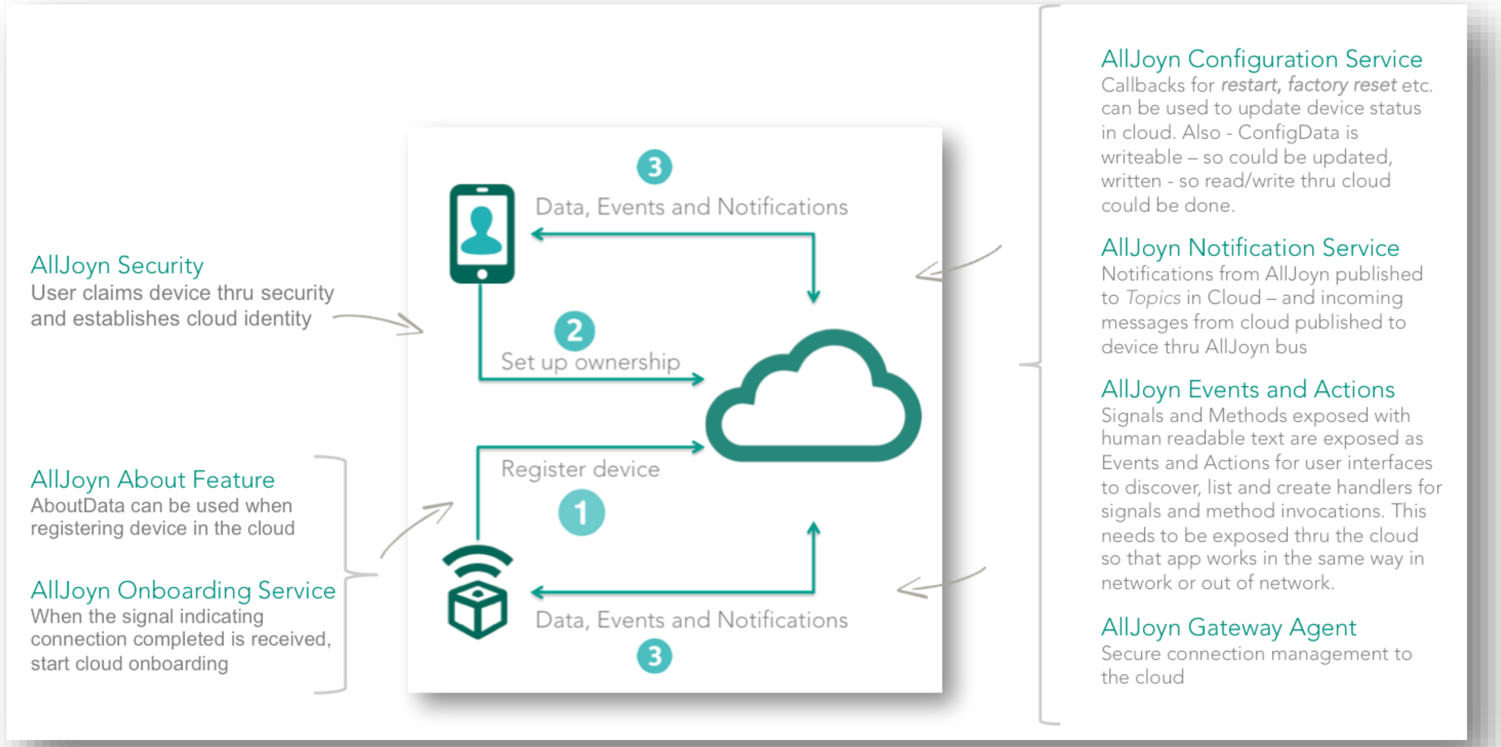
AllJoyn Configuration Service
Callbacks for ***restart***, ***factory reset*** etc. can be used to update device status in cloud. Also - ConfigData is writeable – so could be updated, written - so read/write thru cloud could be done.

AllJoyn Notification Service
Notifications from AllJoyn published to *Topics* in Cloud – and incoming messages from cloud published to device thru AllJoyn bus

AllJoyn Events and Actions
Signals and Methods exposed with human readable text are exposed as Events and Actions for user interfaces to discover, list and create handlers for signals and method invocations. This needs to be exposed thru the cloud so that app works in the same way in network or out of network.

AllJoyn Gateway Agent
Secure connection management to the cloud

But What Is Missing Is A Standard Way To


3
Data, Events and Notifications
2
Set up ownership
Register device
1
Data, Events and Notifications
3

AllJoyn Security
User claims device thru security and establishes cloud identity

AllJoyn About Feature
AboutData can be used when registering device in the cloud

AllJoyn Onboarding Service
When the signal indicating connection completed is received, start cloud onboarding

AllJoyn Configuration Service
Callbacks for *restart, factory reset* etc. can be used to update device status in cloud. Also - ConfigData is writeable – so could be updated, written - so read/write thru cloud could be done.

AllJoyn Notification Service
Notifications from AllJoyn published to *Topics* in Cloud – and incoming messages from cloud published to device thru AllJoyn bus

AllJoyn Events and Actions
Signals and Methods exposed with human readable text are exposed as Events and Actions for user interfaces to discover, list and create handlers for signals and method invocations. This needs to be exposed thru the cloud so that app works in the same way in network or out of network.

AllJoyn Gateway Agent
Secure connection management to the cloud

- register a device in the cloud

Register an AllJoyn installation first and then various devices within the installation as needed

- send telemetry data to the cloud

Send data from various hooks (as described before) to the cloud the device is registered with

Note that receive from cloud doesn’t need any special treatment.

Provisioning
Scope

What will the provisioning working group work on?

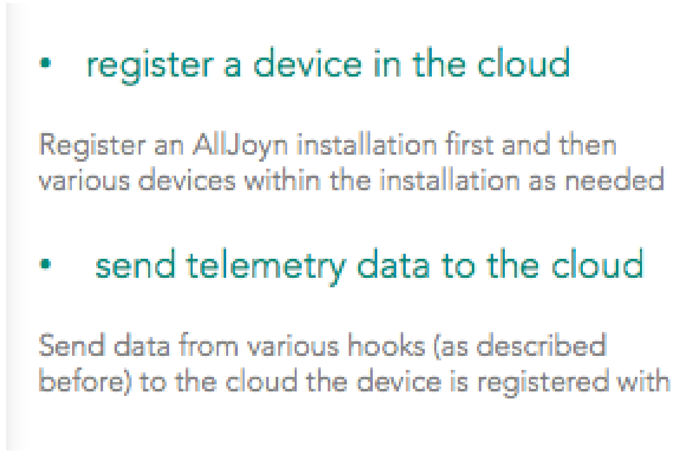
Thing activation API

Thing telemetry API

Integration with Base Services

Integration with Core Framework

This will be the focus of the working group



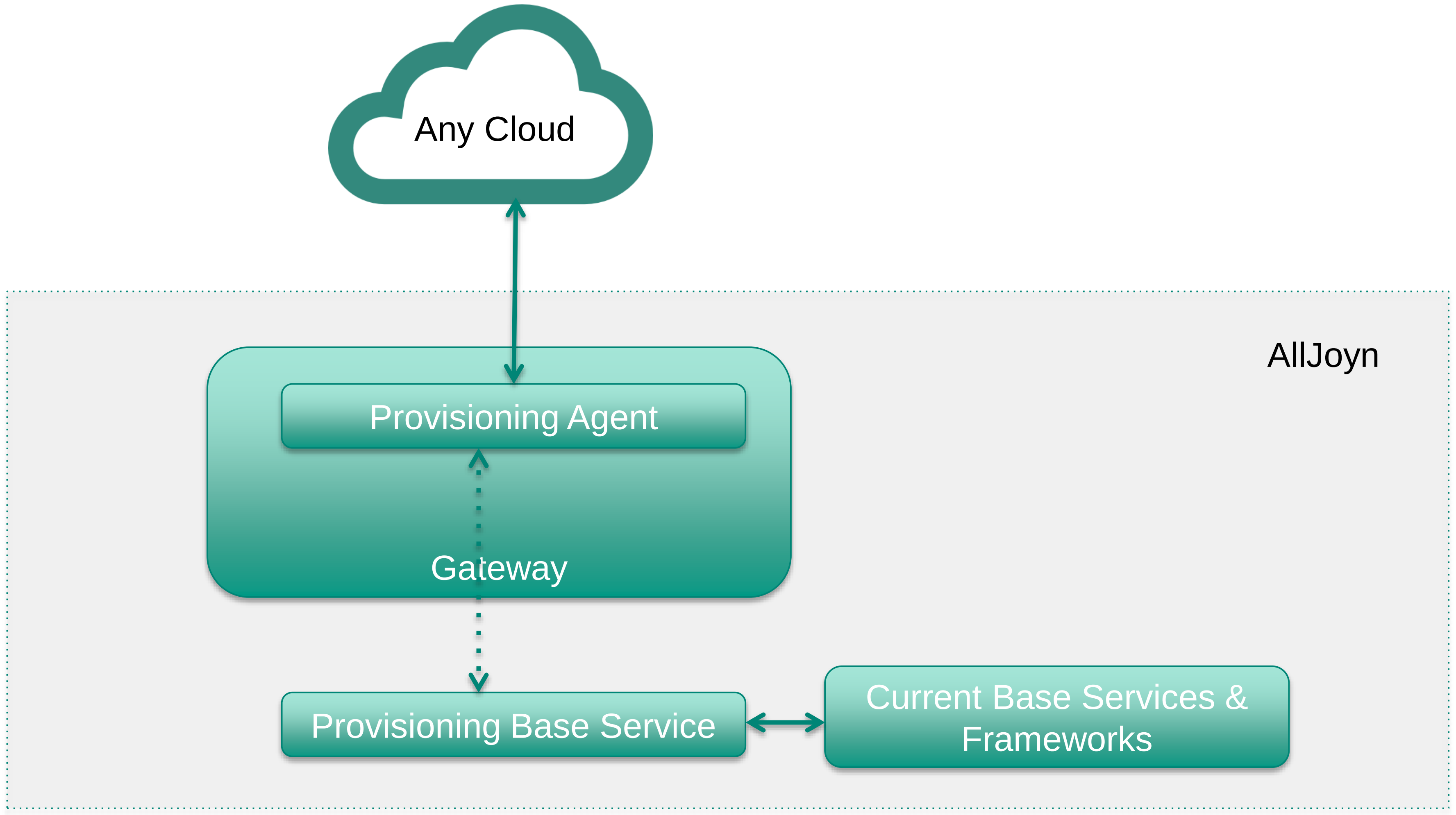
- register a device in the cloud

Register an AllJoyn installation first and then various devices within the installation as needed

- send telemetry data to the cloud

Send data from various hooks (as described before) to the cloud the device is registered with

How This Could Work



Provisioning Dependencies

Dependencies

- This spec depends on and integrates with the following core framework and base services
  - About Feature
  - Actions and Events
  - Onboarding Service
  - Notification Service
  - Gateway Agent
  - Security
  - Configuration Service

Provisioning Project Team

Project Team

- Committers & Maintainers
  - Phani Pandrangi (Kii)
    - phani@kii.com
  - Chris Beauchamp (Kii)
    - chris.beauchamp@kii.com
- Contributors
  - Phani Pandrangi (Kii)
  - Chris Beauchamp (Kii)
  - Noah Harlan (twobulls)
    - noah@twobulls.com
- Other Committers & Maintainers – Open/TBD

Provisioning Timeline

Timeline

- Propose to create the first release in 1-2 months after the TSC/approval of this proposal

Thank You

Follow Us On

- For more information on AllSeen Alliance, visit us at: allseenalliance.org & allseenalliance.org/news/blogs