ALLSEEN ALLIANCE

Core Working Group

October 03, 2014

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

Agenda

1. AllJoyn Build and SDK improvements proposal
2. Overview of QCE process for validating Core releases
3. Continue discussion regarding Core processes
   - New feature requests
   - Test

AllJoyn Build and SDK Improvements Proposal

Summary

- AllJoyn Build and SDKs have become inconsistent and cumbersome over time.
- Proposal is to improve build and SDK.
- Some changes will go into 14.12, others afterwards
- Seeking feedback and contributors to help implement
- Hope is this becomes defacto for all AllSeen WGs

Proposal Summary

1. One source tarball per WG
2. One SDK per platform per WG
3. Consistent tarball and SDK names
4. Release “release” SDKs only
5. Single build products directory
6. Build from any directory
7. Make it easy to run build products
8. Restructure dist/sdk directory
9. Revisit iOS packages/libs

1. One source tarball per WG

- Problem: we currently don’t release source tarballs for all source.
- 1 per WG keeps projects cleanly separated, allows WG to rev independently

2. One SDK per platform per WG

- One SDK per “platform” per source tarball
- “Platforms” are:
  - Android SDK
  - Android NDK
  - Mac OSX
  - iOS
  - Windows (merge 32 bit and 64 bit into one SDK)

3. Consistent tarball and SDK names

- alljoyn-<component>-<ver>-<type>-<platform>.<ext>
- Examples:
  - alljoyn-core-14.12-src.tar.gz
  - alljoyn-core-14.12-sdk-ios.zip
  - alljoyn-base-services-14.12-src.tar.gz
  - alljoyn-lighting-14.12-src.tar.gz
  - alljoyn-lighting-14.12-ndk-android.zip

- Source Tarball format: .tar.gz
  - More flexible if we need symlinks
- SDKs format: .zip
  - Better supported natively on all platforms

4. Release “release” SDKs only

- Proposal:
  - Only release “release” variant of SDK
  - Update “release” variant with debug logging.
    - Disable logging by default
    - Can turn logging on at run time

- Why
  - debug logging with run-time control is most important to most developers.
  - most developers don’t need debug symbols
  - those who need debug symbols are contributors, who would likely be building from source anyhow
  - single SDK variant easier to manage
  - debug strings adds only small amount of code to library.
  - those needing the smallest lib can rebuilt without debug logging
- Some concerns over performance; analysis underway

5. Single build products directory

- single build products directory regardless of where build is executed from
- Will be located from top level, e.g.
  - alljoyn/
    - core/
    - services/
    - build/

6. Build from any directory

- same build/out dir regardless of where build started
- implementation may be related to larger scons cleanup

7. Make it easy to run build products

- Easy means:
  - no explicit typing LD_LIBRARY_PATH
  - no need to cd into bin dir or explicitly add path

- Potential implementations
  - wrapper scripts
  - scons install to install into system/specified dirs
  - perhaps automatically add path to dist libs and bin to shell env

8. Restructure dist/sdk directory

- Needs reorg now that more code has been added
- Optimize structure for common case

8. Restructure dist/sdk directory - Linux

Notes: applications will #include just core.h, config.h, etc; headers placed in core/ and config/ subdirs for c headers (as opposed to c++), #include c variant instead.

```
dist/
    bin/
        alljoyn-daemon
    include/
        alljoyn/
            core.h
            core_c.h
            config.h
            config_c.h
            core/
                <additional core headers>
            core_c/
                <c binding header files>
            config/
                <additional config headers>
            config_c/
                <config header files>
    lib/
        liballjoyn.a
        liballjoyn.so
        liballjoyn_c.so
        liballjoyn_c.a
        libajrouter.a
        alljoyn/
            BundledDaemon.o
        liballjoyn_about.a
        liballjoyn_about.so
        liballjoyn_config.a
        liballjoyn_config.so
        <etc...>
    mozilla/
        libnpalljoyn.so
    java/
        liballjoyn_java.so
        alljoyn.jar
        about.jar
        config.jar
    docs/
        index.html
        cpp/
            index.html
            core/
                index.html (core cpp docs)
            config/
                index.html (config cpp docs)
        c/
            index.html
            core/
                index.html
            config/
                index.html
        java/

    samples/
        cpp/
            core/
                basic/
                    <buildable cpp source>
                chat/
            config/
                <buildable cpp source)
        c/
        java/
            core/
                basic/
                    <buildable java source>
            config/
```

8. Restructure dist/sdk directory - Android

Two SDK zips: SDK and NDK

```
dist/
  sdk/
    apk/
        <pre-built sample apks>
    libs/
        armeabi/
            liballjoyn_java.so
        alljoyn.jar
        alljoyn_about.jar
        alljoyn_config.jar
        <etc>
    docs/
        index.html # links to all java docs
        core/
            index.html # java docs
        config/
            index.hml # java docs
    samples/
        core/
            basic/
                <buildable android source>
                <has libs/jars in here>
            chat/
        config/
```

```
ndk/
    apk
       <pre-built sample apks>
    lib
       liballjoyn.so
       alljoyn_about.so
       ...
    include/
        alljoyn/
            <mirror linux include dir>
    docs/
        index.html # links to all cpp docs
        core/
            index.html (core cpp docs)
        config/
            index.html (config cpp docs)
        <etc...>
    samples/
        core/
            basic/
                jni/ (cpp)
                src/ (java)
            chat/
```

8. Restructure dist/sdk directory – iOS and OSX

Separate iOS and OSX SDKs

```
dist/
  ios/
    docs/
        cpp/
            core/
            config/
        objc/
            core/
            config/
    samples/
        cpp/
        objc/
    cpp/
        inc/
            <match linux headers dir>
        lib/
            <match linux lib dir>
    objc/
        inc/
            <TBD>
        lib/
            <tbd>
```

```
  osx/
    docs/
        cpp/
            core/
            config/
        objc/
            core/
            config/
    samples/
        cpp/
        objc/
    cpp/
        inc/
            <match linux headers dir>
        lib/
            <match linux lib dir>
    objc/
        inc/
            <TBD>
        lib/
            <tbd>
```

9. Revisit iOS packages/libs

- Currently, adding AllJoyn to iOS requires 20 some steps.
- Need to investigate simpler solutions to add AllJoyn in 1 minute.

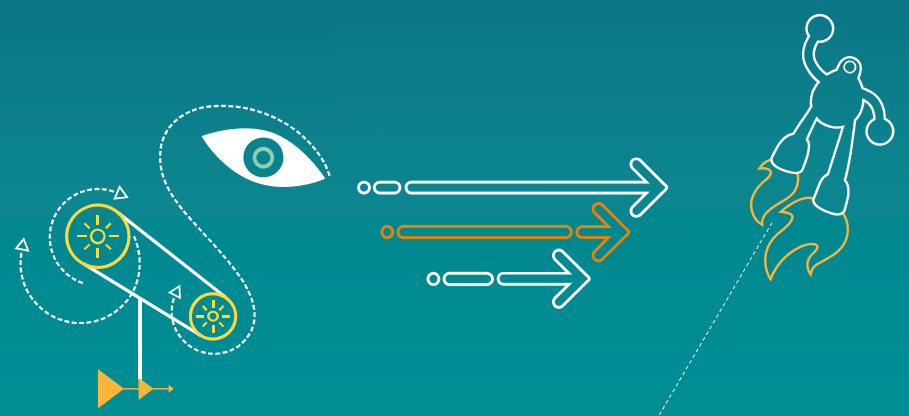
Other suggestions?

Link to QA and test overview slide deck

David McBride
Engineer, Sr. Staff/Manager
Qualcomm Connected Experiences, Inc.

AllJoyn™ QA and Test Overview

QUALCOMM®



Continue discussion regarding Core processes

Core Process Discussion

- Triage meetings
  - Held on Wednesdays at 3pm PT
  - Considering using JIRA Voting mechanism
    - Marcello/Chris to look into how to use this feature and send email to Core WG list
- Release content discussion
  - Propose leveraging JIRA tickets
    - Consider creating a “new feature request” and proposed features are tracked through JIRA
      - Consider updating required fields for this request to include documentation and testing requirements
    - Consider a Wiki page for releases that point to the JIRA tickets
    - Leveraging JIRA tickets for new features will be voted on the next Core call
  - Documentation
    - Members agreed to discuss with their teams and will update
    - This includes
      - Identifying the relevant documents affected
      - Making appropriate updates within the time required to meet the desired release

Core Process Discussion

- Test
  - Members agreed to discuss with their teams and will update
  - Testing includes
    - Feature
    - C&C
    - Regression
- Test coordination
  - Members agreed to discuss with their teams and will update
    - Open question: What does test coordination mean to each of the members?
  - Effort to publish current test cases and test plans for Core releases underway
  - Still requires discussion:
    - Scaling test across contributors

Thank You

Follow Us On 

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