

1. App Developer Storyboard	2
1.1 App Developer Device Observation	2
1.2 App Publishing	3
1.3 App Simulation and Testing	5
2. Bosch Admin Storyboard	6
2.1 App Approval	6
2.1.1 App Compatibility Testing	9
2.1.2 App Content Testing	10
2.1.3 App SW Testing	10
2.2 App Management	11

App Developer Storyboard

Navigation

[Home](#)

Search

search key:

App Developer Storyboard

- [App Developer Device Observation](#)
- [App Publishing](#)
- [App Simulation and Testing](#)

App Developer Device Observation

Navigation

[Home](#)

Search

search key:

User Story: Device Observation

User Story ID: AppDeveloper3

Actor: App Developer

Pre-condition: device specifications are provided by manufacturer and can be accessed through WSC.

Scenario

1. App developer logs into his/her account

2. (Alternatives)

- Bosch allows app developers to browse the available devices and features after paid registration
- Bosch allows app developers to browse the available devices and features after basic registration (just name, email address, and password)

3. App developer browses the list of devices and their information as follows:

- device manufacturer

- device capabilities
- device driver specifications
- usage statistics for the device (how many devices already installed, how many applications aims to use the device, how many applications are being used for the device)
- currently available apps for the device and their functionality
- manufacturer restrictions on a device
- Bosch restrictions on a device

UI storyboard

...

Discussions

1. List of devices and their features

- What kind of information we should allow the app developers to see?
- If there is a device specific functionality (if we allow manufacturers to provide such functionality)
 - How do we provide sufficient information to app developers in order them to use all the functionality of the device correctly, effectively and efficiently?

2. Device statistics

- Should we provide the number of installed devices?
 - Option 1 - Yes: Widely used devices will have more apps available, while rarely used devices might not have enough apps to use
 - Option 2 - No: App developers might be discouraged when they provide apps for the devices which is not installed & being used yet or installed & used rarely

3. Available apps statistics

- Should we provide app usage statistics for a device?
- Should we provide what kind of apps are already available and how many of them are purchased and how often they are being used?
 - No existing app store provides these info, providing may cause some types of the apps (which are rarely used) being ignored always

4. Should we allow device manufacturers put some restrictions on a device (this applies if we allow custom functionality)?

- If yes, how do we control these restrictions?

5. Should we put some restrictions for a device?

- e.g. security devices cannot have apps which use public internet access
- if yes, how do we define these restrictions and how do we make sure app developers follow?

App Publishing

Navigation

[Home](#)

User Story: App Publishing and Management

User Story ID: AppDeveloper1

Search

Actor: App Developer

search key:

Pre-condition: App Developer uses Bosch Smart Home SDK and develops an app.

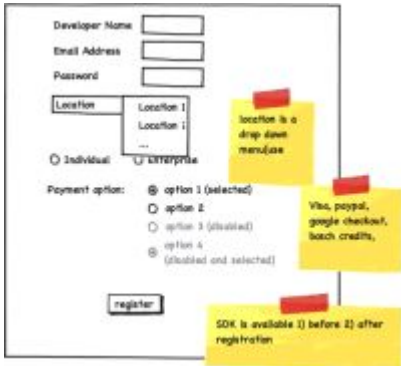
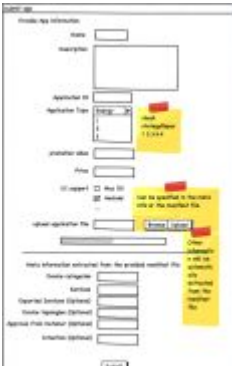
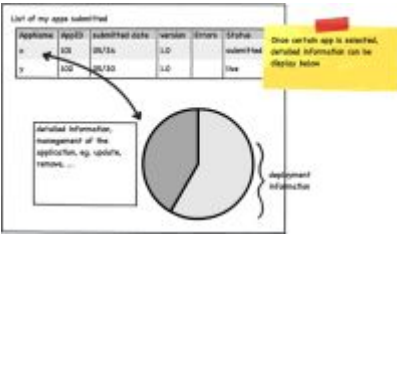
Scenario

1. App developer registers his/her account

- provide user interfaces: developer name, email address, password, location info, etc
 - individual/enterprise selection, payment option (paypal, google checkout, BoschCash, visa ...)
2. (Alternatives)
- Bosch allows app developers to download SDK only after registration
 - Bosch allows app developers to download SDK after basic registration (just name, email address, and password)
3. App developer provides app information including
- app name, description,
 - App ID (generated by Web Solution Center)
 - screenshot (optional)
 - promotion video
 - app category (entertainment, energy, security, comfort...)
 - application software and meta information (smart home specific manifest file is required)
 - Web Solution Center propagates the manifest information automatically
4. App developer submits the app.
5. App developer sees the list of apps including the status of apps (submit, approved, live, pending, ...).
6. App developer manage the app including update the app, see the statistics of download, errors.

UI storyboard

- UI Mock-up: pdf file

1. App registration	2. App Submission	3. App Management															
 <p>Developer Name: <input type="text"/></p> <p>Email Address: <input type="text"/></p> <p>Password: <input type="password"/></p> <p>Location: <input type="text"/> Location 1: <input type="text"/> Location 2: <input type="text"/></p> <p>Individual <input type="radio"/> Enterprise <input type="radio"/></p> <p>Payment option: <input checked="" type="radio"/> option 1 (selected) <input type="radio"/> option 2 <input type="radio"/> option 3 (disabled) <input type="radio"/> option 4 (disabled and selected)</p> <p>register</p> <p>location is a drop down menu/see</p> <p>Visa, paypal, google checkout, Bosch credits,</p> <p>SDK is available 1) before 2) after registration</p>	 <p>App Name: <input type="text"/></p> <p>App ID: <input type="text"/></p> <p>App Version: <input type="text"/></p> <p>App Description: <input type="text"/></p> <p>App Icon: <input type="text"/></p> <p>App Category: <input type="text"/></p> <p>App Status: <input type="text"/></p> <p>App License: <input type="text"/></p> <p>App Privacy Policy: <input type="text"/></p> <p>App Terms of Service: <input type="text"/></p> <p>App Manifest File: <input type="text"/></p> <p>Submit</p>	 <p>List of my apps submitted</p> <table border="1"> <thead> <tr> <th>App ID</th> <th>App Name</th> <th>Submitted Date</th> <th>Version</th> <th>Status</th> </tr> </thead> <tbody> <tr> <td>101</td> <td>App 1</td> <td>20/10/16</td> <td>1.0</td> <td>Submitted</td> </tr> <tr> <td>102</td> <td>App 2</td> <td>20/10/16</td> <td>1.0</td> <td>Live</td> </tr> </tbody> </table> <p>Due to certain app is selected, detailed information can be displayed below</p> <p>Detailed information, management of the application, e.g. update, remove, ...</p> <p>Development Information</p>	App ID	App Name	Submitted Date	Version	Status	101	App 1	20/10/16	1.0	Submitted	102	App 2	20/10/16	1.0	Live
App ID	App Name	Submitted Date	Version	Status													
101	App 1	20/10/16	1.0	Submitted													
102	App 2	20/10/16	1.0	Live													

Discussions

1. Payment option for SDK
- One-time payment (like Google)
 - Annual payment
 - Individual account: 100 devices in development phase before publication (apple)
 - Enterprise account
 - managed by each company
 - allows internal distribution
2. Download SDK
- Do we allow downloading Bosch SDK without registration? (like Google)
3. App sharing/publishing only through appstore or allow users to share apps individually with each other without appstore or share over a third-party appstore(like Amazon market)?

4. Should Bosch allow virtual money such as BoschCash such as Facebook credits e.t.c.?

5. SDK Provisioning procedure

- Certificate for the user: created and downloaded from Web Solution Center
- Application ID generation

6. Meta information for App (Manifest file)

- Device categories (types) of devices required for the App (e.g., security system)
- Services (functionalities) required from the devices (e.g, arm, disarm, alarm)
- Imported APIs from gateway
- Exported APIs from App for further use for UI or other developers
- Device topologies
 - (e.g., energy application may requires three motion sensors deployed to the home with specific spatial information)
 - Topology can be automatically generated from simulation environment and test results
 - Topology can be specified by App developer manually
- Approval (Confirmation) from installer before running an app
- App Category :comfort, protect, save, ..
- Resolve conflicts from different intentions from apps
 - intentions can matched to home profile (comfort, protect, save, ...)
 - Appstore shall provide prediction of conflicts between different apps and intention
 - There should be a policy-based control system using user profile to extract and/or prevent conflicts in the gateway architecture
 - example1: a policy is like "If it's above 80 Fahreneit don't increase temperature when save mode is active
 - example2:windows can be open if save profile (mode) is active but not be open if protect mode is active

Note: Enabling technologies include ontology, virtual smart home simulator running on the cloud, data mining ...

7. Need to decide how we can leverage simulation and test data from cloud for SDK and to provide efficient and reliable approval process

- Test sets
- Use of devices
- Topology
- etc

8. Need to device how we can leverage user experienced errors after deployment, how we create error reporting channel with security support.

- e.g., Android uses strings for error report generation, which introduce a security hole

9. Approval phases (from Amazon Appstore - may discuss which are relevant for Bosch)

- Submitted
- Under review
- Approved
- Live
- Pending
- Rejected
- Suppressed

App Simulation and Testing

Navigation

[Home](#)

User Story: App Publishing and Management

User Story ID: AppDeveloper2

Search

Actor: App Developer

search key:

Pre-condition: App Developer uses Bosch Smart Home SDK and develops an app.

Scenario

1. app developer runs app on the simulator on the bosch simulation cloud
 - app developer creates his/her own virtual devices
 - app developer uses virtual home environments provided by bosch
2. app developer tests application
3. option 1: app developer submits the test results
option 2: app developer does not submit test results, record the bugs for debugging

UI storyboard

UI needs to be discussed in the context of SDK and IDE.

Discussions

1. Before submitting an app by app developer
 - Run app on virtual smart home profiles
 - Security small home
 - Security medium house
 - etc
 - Run app on their own defined profiles
 - Run in real room rented by Bosch
2. SDK should provide a test wizard to app developer
 - It should be optional to use it
3. Do we need to enforce app developers to publish test results to Bosch?
 - Tradeoffs: e.g. liability

Bosch Admin Storyboard

Navigation

[Home](#)

Search

search key:

Bosch Admin Storyboard

- [App Approval](#)
 - [App Compatibility Testing](#)
 - [App Content Testing](#)
 - [App SW Testing](#)
- [App Management](#)

App Approval

Navigation

[Home](#)

User Story: App Approval

User Story ID: BoschAdmin1

{*}Search*

Actor: Bosch Admin


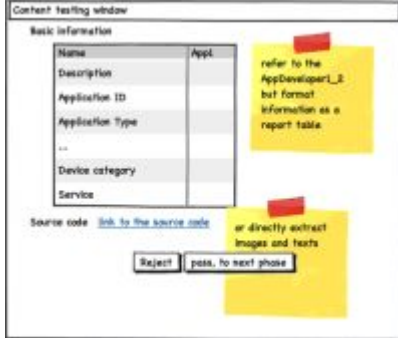

search key:

Pre-condition: App Developer submitted an app through Web Solution Center.

1. Bosch Admin gets new applications.
 - list of submitted applications
2. Bosch Admin tests applications with respect to.
 - contents
 - test cases
 - software bugs
 - compatibility with the real devices
3. Bosch Admin decides
 - the submitted application is approved or not.
 - if app is not approved, notifies the developer and gives the reasons for rejection.

UI Storyboard

- [UI Mockup pdf file:](#)

1. List of apps	2. Content testing window	3. Reason for rejection
		

Discussion

1. Vetting (Approval) process
 - content
 - what is Bosch guideline for contents?
 - SW bug testing
 - Test cases

- Know-how, represented a model e.g. on, off, functionality should be exclusive
 - Model checking
 - How do we create test cases?
 - How do we leverage test cases created by app developers?
- Automated checking for SW quality
 - ***violation of using provided APIs
- Compatibility with devices
 - Some visualization of app working on defined profiles for human testers

2. How do we make sure we provide relevant test cases w/o concrete requirement for the provided app?

- Need more protection mechanism in the gateway
 - E.g. protection against device constraints: check whether app tries to increase temperature more than maximum
 - Options about conflict management:
 - Allow apps to do every legal action without checking conflicts
 - Check by policies on runtime
 - Check rigorously during approval process
 - Even in this case, the gateway architecture will need policies as conflicts are about the state

3. Trade-offs in testing process

- Phases:
 - Content testing
 - Software testing
 - Compatibility of hardware testing
- Pipeline testing phases in order to make Bosch's tester's time consumption efficient
- Don't pipeline and test from every aspect in parallel in order to provide rejection information in more detail to app developer
 - This makes more sense when testing process is more automatic
- States of application in testing lifecycle
 - Under content testing
 - Under software testing
 - Under compatibility of hardware testing
 - Accepted
 - Rejected

3. In App Approval scenarios

*In content window show to the content checker

- We can just give the source code/app to the tester
 - System can extract the content (images, writings etc) and show in content checking window
- What are the rejection criteria and rejection causes that will appear in reasons for rejection dialog box?
 - The rejection causes can change according to the phase app rejected
 - For content phase:
 - Used copyrighted material
 - Using inappropriate material: Racist e.t.c.
 - For SW phase:
 - Failing some tests:
 - like test home test, medium home test e.t.c.
 - like security test, privacy test, energy test, e.t.c.
 - Unchecked edge cases (extreme cases)
 - Using unauthorized libraries or APIs
 - Unintended behaviour
 - illegal functionality
 - For compatibility phase:
 - (driver problems: this will be taken care at the testing for drivers from manufacturers)
 - unauthorized usage of devices (e.g. trying to dim a turn on/off lamp or getting a heater with 110V)

For the software test window

- Should we just give the source code for tester to make tests somewhere else?
 - In this case should we give him a list of tests he should run for convenience, reminding?
 - Tester can (should) run optional tests too.
 - The default tests list may differ according to the type of the application, services e.t.c.

- Should we make tester to make tests over Web solution center?
 - Automated tests should also be reported
 - Automated tests can start just after content test is over
 - Automated tests can start with software tester confirmation
- Test order:
 - Tests should be performed with the order one by one by the order listed
 - Tests can be performed in any order
- When software fails a test
 - Tester stops testing and reject
 - In this case, the report returned to the application developer can include information as
 - performed 7 tests failed 1 nonperformed 8(or on rejection window instead of just checkbox r
 - just about failed test case
- - Tester continue testing and when rejecting return all the fail reasons
 - Instead of one check box representing a test performed two check boxes fail succes can be shosen e: meanse passed X means failed)

SW test window can have

- a general progress bar
 - just for automated tests
- What happens if auto tests take longer than manual tests?
 - Tester can be notified
 - by a message
 - on main screen
 - Tester should return himself later

For compatibility test window

- We can show software test results for compatibility tester
- We should show compatibility test window before software window because software test is tightly dependent over con
- Do we need to provide application or source code for compatibility tester? Will he need to run it?

App Compatibility Testing

Navigation

[Home](#)

Search

search key:

User Story: App Compatibility Testing

User Story ID: BoschAdmin1_3

Actor: Bosch Admin

User Story

Pre-condition: Bosch Admin Lists approval pending apps (ready for compatibility evaluation)

1. bosch admin clicks one of the pending apps

- alternative 1: app info and source download link provided
- alternative 2: app info and compatibility testing tool provided

2. bosch admin downloads the source code and tests against compatibility
 - alternative: bosch admin tests against compatibility using the tool provided by Web Solution Center
3. bosch admin investigates automated test results on Web Solution Center
4. bosch admin decides
 - the submitted application is compatible, so publishes the app
 - the app failed the test and rejected, so provides reason for rejection

App Content Testing

Navigation

[Home](#)

Search

search key:

User Story: App Content Testing

User Story ID: BoschAdmin1_1

Actor: Bosch Admin

User Story

Pre-condition: : Bosch Admin Lists approval pending apps (ready for content evaluation)

1. bosch admin clicks one of the pending apps
 - alternative 1: app info and source download link provided
 - alternative 2: app info and extracted content is provided
2. bosch admin downloads the source code and investigates the content
 - alternative: bosch admin investigates the content provided by the wsc
3. bosch admin decides
 - the submitted application is passed content evaluation, so lists for sw testing
 - the app failed the test and rejected, so provides reason for rejection

App SW Testing

Navigation

[Home](#)

Search

search key:

User Story: App SW Testing

User Story ID: BoschAdmin1_2

Actor: Bosch Admin

User Story

Pre-condition: Bosch Admin Lists approval pending apps (ready for compatibility evaluation)

1. bosch admin clicks one of the pending apps
 - alternative 1: app info and source download link provided
 - alternative 2: app info and sw testing tool provided
2. bosch admin downloads the source code and tests the sw
 - alternative: bosch admin tests the sw using the tool provided by the wsc
3. bosch admin investigates automated test results on wsc
4. bosch admin decides
 - the submitted application is passed sw evaluation, so lists for compatibility testing
 - *the app failed the test and rejected, so provides reason for rejection

App Management

Navigation

[Home](#)

Search

search key:

User Story: App Management

User Story ID: BoschAdmin2

Actor: Bosch Admin

Pre-condition: List of available apps and some filtering options are available through WSC.

1. Bosch Admin logs into his/her account.
 - list of available applications
2. Bosch Admin browses app usage history and user feedback.
 - user provides reviews and rating - no action needed(?)
 - user submits an appeal - admin decides an action
 - rejects the appeal
 - notifies the user with the reason for rejection
 - approve the appeal
 - notifies the app developer
 - if serious problem found (reported)
 - option 1: kills the application temporarily

- option 2: updates the compatible device list

3. Bosch Admin decides and updates the compatible device list according to the reviews and notifies the app developer

UI Storyboard

...

Discussion

1. How do we kill applications (make no longer available to purchase and make non-functional on the installed home gateways)?
2. Should we allow app developers to submit new apps if they publish malicious apps?
3. Should we kill all apps of a malicious app developer?