# Telemetry Summery

## 1. The database structure of Dual SIM Card

The structure related to Dual SIM Card are network, icc and deviceinfo.MEID, and they are changed from object to array, the element of array is the same as the former object.

The former:

"network": {

"mcc": "216",

"mnc": "70",

"operator": "vodafone HU vodafone"

},

"icc": {

"spn": "vodafone",

"mcc": "216",

"mnc": "70"

},

"deviceinfo.MEID": 123456789012345

After the change:

"network": [

{

"mcc": "216",

"mnc": "70",

"operator": "vodafone HU vodafone"

},

{

"mcc": "216",

"mnc": "70",

"operator": "vodafone HU vodafone"

}

],

"icc": [

{

"spn": "vodafone",

"mcc": "216",

"mnc": "70"

},

{

"spn": "vodafone",

"mcc": "216",

"mnc": "70"

}

],

"deviceinfo.MEID": [123456789012345, 123456789012345]

## 2. The rule of sending message: 1) Retry 2) Monthly

### 2.1 The work flow of FtuPing

The entry point of FtuPing is ensurePing, after promise chain of self.initSettings(), self.initPreinstalledApps(), self.initImeis(), self.startPing() is called. An interval of this.\_pingTimer is set through the code: this.\_pingTimer = setInterval(this.tryPing.bind(this), this.\_tryInterval);

As a result this.tryPing() is call by interval of this.\_tryInterval.

Notice: Even this.\_networkFailCount >= this.\_maxNetworkFails and Max voice network failures reached, pinging anyway!

This.ping() is called in function tryPing(), it duty is send data collected by the promise chain above to server with a call of global object TelemetryRequest which is exported in file shared/telemetry/telemetry.js.

this.\_pingEnabled is used to indicate ping or not, it is set to true by default. After ajax callback function of pingSuccess is called, this.\_pingEnabled is set to false, and ping action will stop at funciton startPing(), so the ping action is not executed at all. Once FtuPing is success, this.\_pingEnabled is set to false, ping action will never excuted unless you set \_pingEnabled to true in function reset.

As a summery, retry of action ping is set by setInterval in function startPing, as the value of this.\_tryInterval is set to 60\*60\*1000 by default, ping is activated per hour. This.\_networkFailCount is used to count the fail times of checkMobileNetwork , even it is bigger than this.\_maxNetworkFails, ping action will continue until success, once success, the ping action will stop. You can set this.\_pingEnabled to true by function reset call, then ping action will consume. The mechanism of Monthly call does note exits in current code.

### 2.2 The work flow of function call



## 3. Which service enable Telemetry(The start of Telemetry)

### 3.1 introduce to FtuLauncher

FTU is known as First Time Usage, it is a part of System Application, all actions used by FTU is located in file js/ftu\_ping.js. FtuLauncher is a service registered in file js/ftu\_launcher.js, it is one of the services defined in System App. The entry point of FTU is started by FtuLauncher service.

System is a base Application, and all the related files are located in directory gaia/apps/system/. Some start0up javascript files are launched by index.html, including js/applications.js, shared/lazy\_loader/lazy\_loader.js, js/service.js, js/base\_module.js, js/app.js, js/bootstrap.js, and they are very important for System App startup. Other js files needed are loaded via lazy\_loader.js.

### 3.2 Concept of BaseModule

Window.BaseModule which is export in js/base\_module.js, and it is a very import object for module management in System App.

BaseModule is a class skeleton which helps you to build a module with

\* Centralized event handler

\* Centralized settings observation

\* Sub modules management including loading and starting

\* Import preload files

\* DOM rendering

\* Consistent logging function with System boot time and module name

\* Common publishing interface

BaseModule is a very important concept used to warp all services and modules used in System Application, Launcher, FtuLauncher, and Core are all managed by BaseModule.

Important properties of BaseModule are create, instantiate, and SUB\_MODULES.

BaseModule.create is used to create a module based on base module and give properties. A constructor will be returned and placed in AVAILABLE\_MODULES if you specify an unique name in the prototype.

BaseModule.instantiate is used to instantiate an object by name through the constructor stored in AVAILABLE\_MODULES.

BaseModule.SUB\_MODULES is an array store the sub modules belong to this module. BaseModule will load and then start these sub modules automatically.

### 3.3 How FtuPing is started?

Concept of SubModule

Functions of System App is managed in the form of module, such as KeyEventDispatcher, LayoutManager, AudioChannelService, AppWindowManager, AppInstallManager, and FtuLaucher is one of them. Each module has a important property SUB\_MODULES. All submodules must start before the current module, and the module FtuLauncher is started by the submodule chain.

All the files related to FtuPing are list below:

js/bootstrap.js

js/app.js

Core(js/core.js)

AppCore(js/app\_core.js)

AppWindowManager(js/app\_window\_manager.js)

FtuLauncher(js/ftu\_launcher.js)

Js/bootstrap.js is the entry point of whole System App. A global object App exported in js/app.js is started in bootstrap.js. The first module Core is instantiate and start in in app.start, and all its submodule is started one after another, FtuLauncher is a sub module of AppWindowManager, AppWindowManager is a sub module of AppCore, AppCore is a sub module of Core.

## 4. what is the process when the data can not be gotten

Some asynchronous data are got through window.asyncStorage, the following example showing how data is got through window.ayncStorage by itemKeys.

function getAsyncStorageItems(itemKeys, callback) {

var itemsLeft = itemKeys.length;

var items = {};

itemKeys.forEach(function(key) {

window.asyncStorage.getItem(key, function(value) {

itemsLeft--;

items[key] = value;

if (itemsLeft === 0 && callback) {

callback(items);

}

});

});

}

Some data about Setting is get through navigator.mozSettings, in the form of below.

function self\_getSettings(settingKeys, callback) {

var settingsLeft = settingKeys.length;

var settings = {};

var lock = window.navigator.mozSettings.createLock();

settingKeys.forEach(function(key) {

var request = lock.get(key);

request.onsuccess = function(evt) {

var value = request.result[key];

settingsLeft--;

settings[key] = value;

if (settingsLeft === 0 && callback) {

callback(settings);

}

};

});

}

So the case of failing to get a data is handled by window.ayncStorage or navigator.mozSettings, and has not direct relation with FtuPing.

## 5. Change FMD and Kill-Switch to Anti-Theft

https://jira.acadine.com/browse/CORE-1415

## 6. Confirm the content of URL and json

### 6.1 The json string send by FtuPing.

The json string send to server by TelemetryRequest is listed below:

{

"activationTime": 1398300000000,

"devicePixelRatio": 1,

"deviceinfo.MEID": [123456789012345, 123456789012345],

"deviceinfo.firmware\_revision": "",

"deviceinfo.hardware": "qcom",

"deviceinfo.last\_updated": "",

"deviceinfo.os": "2.0.0.0-prerelease",

"deviceinfo.platform\_build\_id": "20150817143800",

"deviceinfo.platform\_version": "37.0",

"deviceinfo.product\_model": "go\_flip",

"deviceinfo.software": "Boot2Gecko 2.2.0.0-prerelease",

"deviceinfo.update\_channel": "default",

"findmydevice.enabled": "",

"icc": [

{

"spn": "vodafone",

"mcc": "216",

"mnc": "70"

},

{

"spn": "vodafone",

"mcc": "216",

"mnc": "70"

}

],

"network": [

{

"mcc": "216",

"mnc": "70",

"operator": "vodafone HU vodafone"

},

{

"mcc": "216",

"mnc": "70",

"operator": "vodafone HU vodafone"

}

],

"locale": "hu",

"pingID": "8f1bbb30-a1f0-4403-938f-88acf19df5ad",

"preinstalled": {

app://deviceinfo.h5os/manifest.webapp:"DeviceInfo",

http://mochi.test:8888/manifest.webapp:"Mochitest"

}

"info": {

"appUpdateChannel": "default",

"reason": "ftu",

"appVersion": "31.0a1",

"geoCountry": "HU",

"appName": "FirefoxOS",

"appBuildID": "20140423185429"

},

"pingTime": 1398300000000,

"screenWidth": 320,

"screenHeight": 480

}

### 6.2 the format of URL

The url used for request includes two parts: the constant part and custom part, the latter is used to indicate a unique device and request.

Constant part is: https://fxos.telemetry.mozilla.org/submit/telemetry/

Custom part include six parts:

deviceID, the pingID generated by function uuid()

Reason: constant string ‘ftu’

appName: constant string ‘H5OS’

appVersion: the value of deviceinfo.platform\_version

appUpdateChannel: the value of app.update.channel

appBuildID: the value of deviceinfo.platform\_build\_id

One of example of URL used for request is list as follow:

https://fxos.telemetry.mozilla.org/submit/telemetry/e426da9f-2a29-4e09-895b-c883903956cb/ftu/FirefoxOS/31.0a1/default/20140325104133