Research for Mbaas iOS design

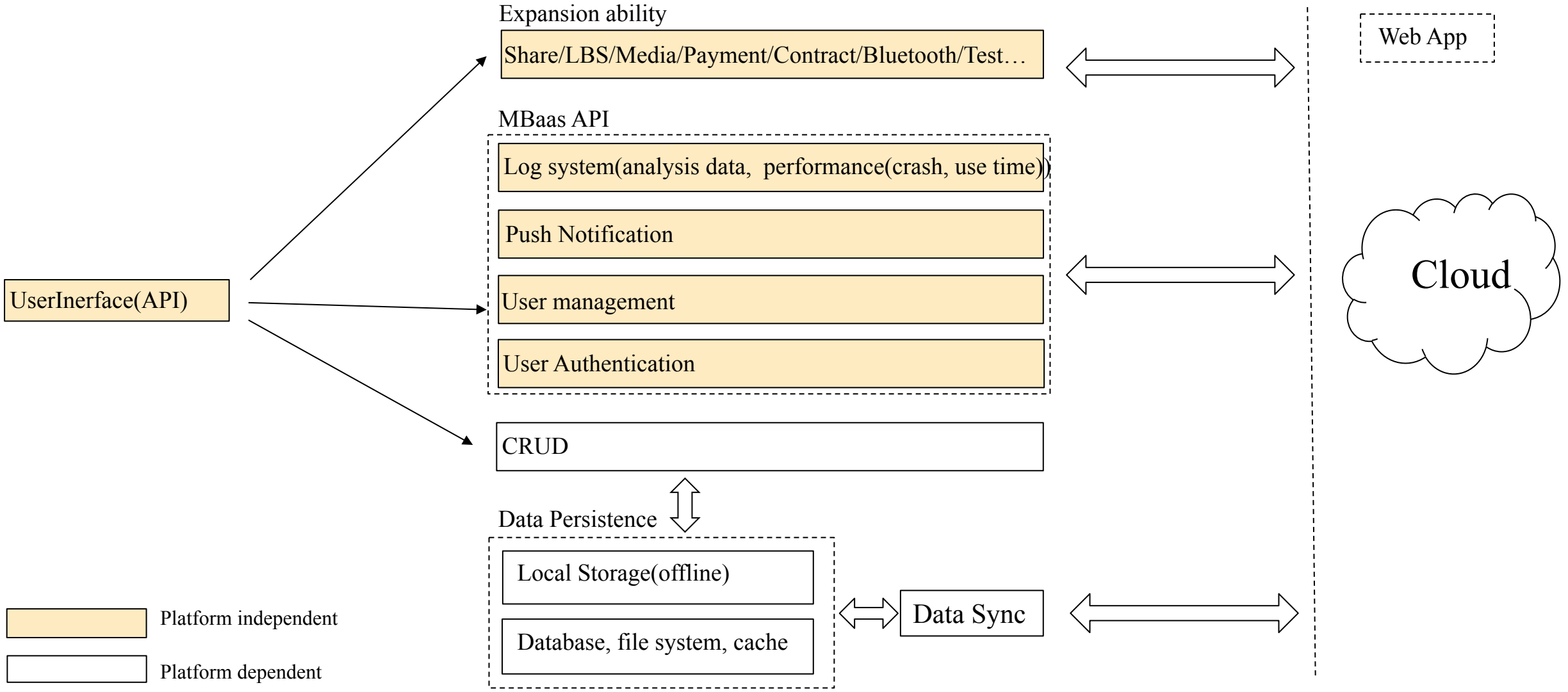
Introduction

- Object

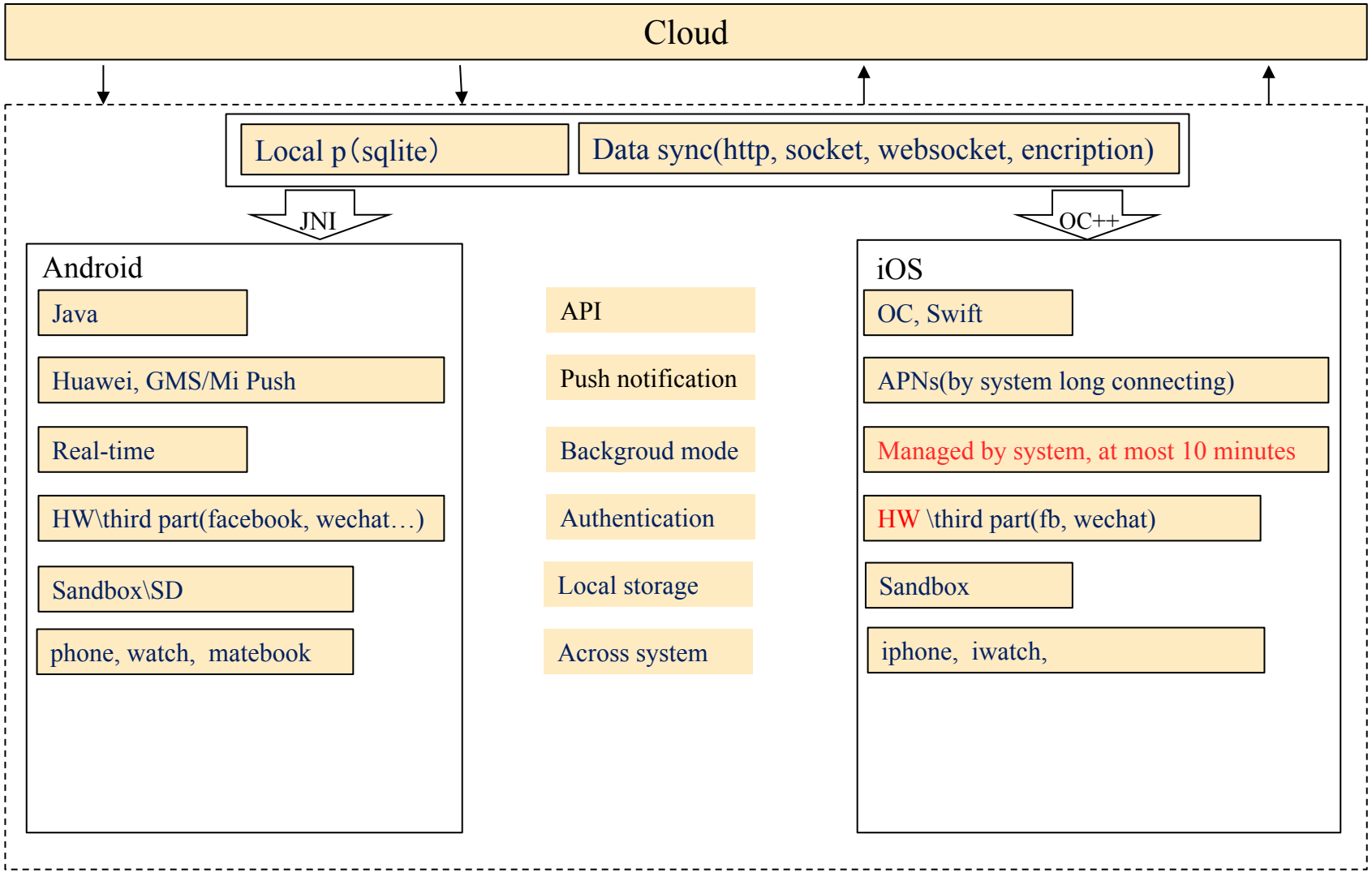
  Do some researchers for the Mbaas iOS client side design based on our whole architecture design, analysis the difference of the platforms and the productions at the market for client side, produce some ideas for our own design, still in continuous update, for today's sharing, mainly for get some feedback, go on to research and plan the next steps.

- Topics:
  - ✓ Techniques at client side for Mbaas
  - ✓ Different platform dependences
  - ✓ Push notification for iOS
  - ✓ Some other thoughts
  - ✓ iOS design for competitive products

Client side architecture



Techniques at client side



Techniques to research under iOS context

- **Technique points**

  1.network layer: network framework, data parse protocol(pb, …), (encryption ?ex:AES128 + ECB + No Padding, compression)

  2.handdle different network status change, how about in Android?

  3.Threads pool and database(sqlite, …) part(also need handle background and frontground),data sync logic,

  4.all API should be wraped by swift/objective-c/java, first stage may use swift for iOS

  5.is there any others platform dependences? local storage: the file path, …, reference another page

  6. Performance analyze:

  - ✓ battery, CPU, launch time should lower than average level, otherwise may will be killed by system(especially for background task and gps)
  - ✓ package size (depends functions, code lines)

- **Need be implemented at different platforms:**

  1.Push Notification

  2.Authrity, user management (the way to get the device id, user id may different, for the sdk produce token for the application)

  3.All third part lib, or the lib from other team should support iOS

  4.Activity time depends system, such as the behaviors at background mode

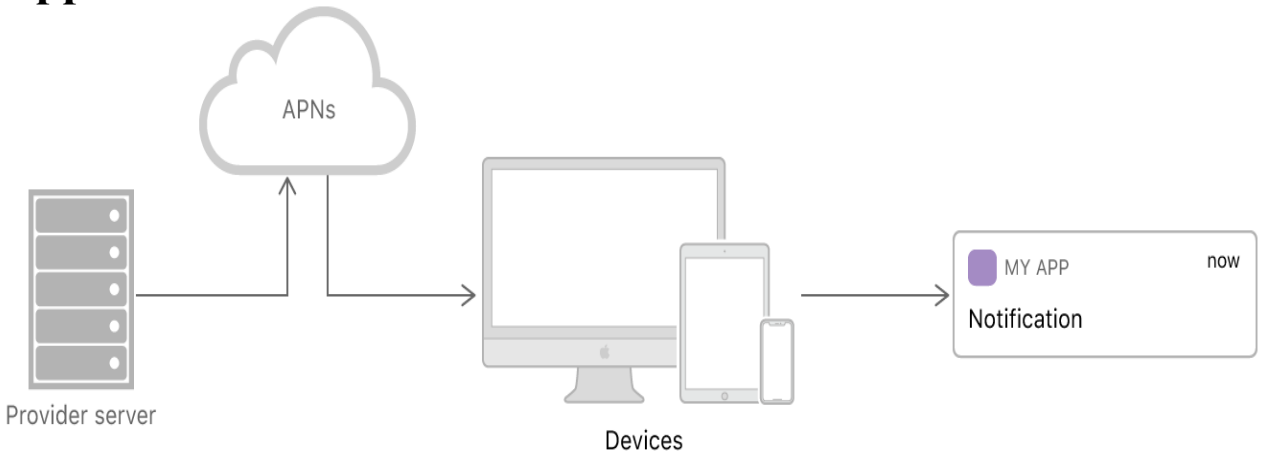
- **Some other Potential challenge**

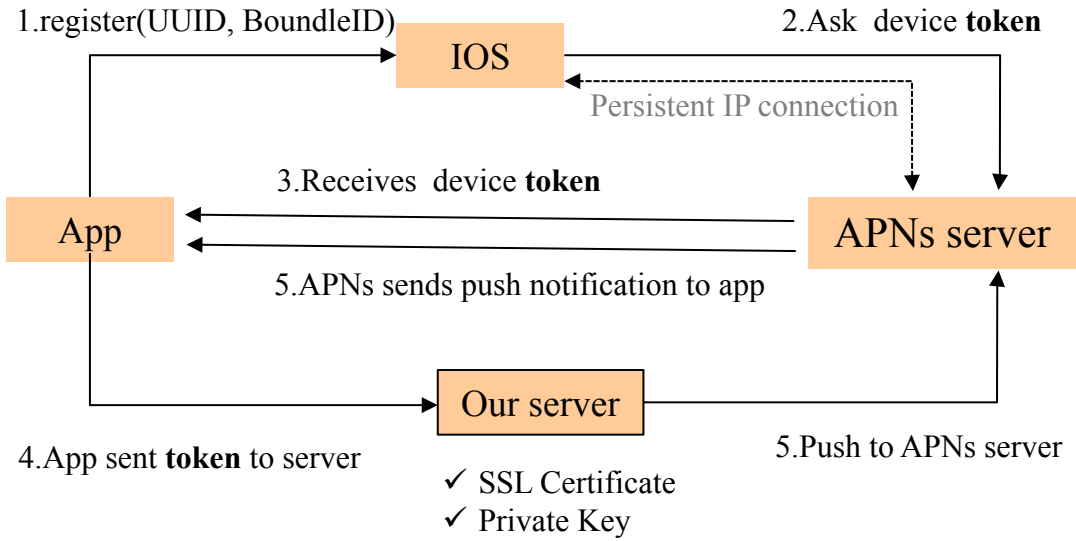
Push Notification at iOS

- **Applications in Mbaas for push notification**

Feathers	Cloudkit	Firebase
Subscription	1.Query for record create, update, deletion 2.Database, CKDatabaseSubscription(not allow for public database) 3.Zone, CKRecordZoneSubscription(not allow for private database) 4.Fetch all subscriptions	
Remote notification	1.APNs 2.Work for mode: active, inactive(screen lock), background, suspend, not running 3.authentication by certification, device token,	1. APNs

- **Apple Push Notification Service**





Push Notification at iOS

- **Apple Push Notification Service**
  - ✓ For iOS, 1 process per app, provides process for the push service, no matter with app status mode.
  - ✓ Allow finite-length task execute at background, around
  - ✓ Some task with specific permission be allow run in background(audio, location, VoIP), need be configured in the project
  - ✓ Background download by NSURLSession (further research)

- **Backgroud mode**
  - ✓ Normally for the background mode limited in 10 minutes .
  - ✓ For GPS, voip, video, not for data download, can be in background without time limit

- **For our situation**
  - ✓ APNS, depends on the iOS system for the tcp long connection(arrive rate)
  - ✓ Long Connection, self implement, but can not be used in not funning and suspended mode, also limited for background mode
  - ✓ Long Connection+APNS, in the front use long connection, at background switch to APNs
  - ✓ For my understand, our core part is data sync, much care about the user experience at frontground

- **To do**
  - ✓ the file payload limit(4KB? 2KB)
  - ✓ Is there any limitation at our system…

Some other thinkings

Steps:

- Phase 0, Research for most features
- Phase 1, Demo for core part at iOS platform, push notification
- Phase 2, Produce API design for iOS
- Phase 3, Produce implement for key-value, objects implement

Client sdk:

- Plan 1, Compile common layer as sdk-> multiple platforms(Android .so, iOS .a) by CMake
- Plan 2, source code compile at different platforms

OC .v.s Swift:

- For stability purpose, most products use OC
- OC is also easy be bridged to swift

Cloudkit v.s. Firebase v.s. Realm iOS sdk design

Features	Common	Cloudkit	Firebase	Realm
Language	OC, swift			
Authentication				
Push notification	APNs			
Architecture		Only online data sync	Different modules for sync, Authentication	
System support			iOS 8+	
Package size				Realm.framework() 83.7 MB binary: 80MB RealmSwift.framework 15.7 MB binary: 9MB

Thank You.

Different between iOS and android

Features	Common	iOS	Android
Local storage	Application sandbox sqlite	Application sandbox(Documents, Library, tmp) NSKeyedArchiver Core Data(core part is sqlite)	SD card, sandbox path(/data/data/xxxx/), File storage, SD , Shared Preferences Content Provider
Backgroud Mode		Process suspend, limit in 10 minutes, system control	Background process, download, upload
Priority of the instructions		UI highest, should be main thread	Data is highest
Multiple threads	Main thread for UI, support c/c++ thread API	NSThread GCD(task and queue) NSOperation/NSOperationQueue RunLoop	Sunable/Thread ExecuteorServie IntentService AsyncTask Service Looper
Push Notification		APNS, Arrived when app isn't active	GCM, by services, only arrived when services running
Shareing		Developer implement, use mid page share to another app	Share Bundle
Process communication		URL Schema, UIPasteboard, CFMessagePort, CFNotificationCenter	Bind(ContentProvider, Activity, boardcast, AIDL)
GC	Reference count	ARC	Dalvik/ART
Others(platforms dependent)	GPS, Camera, TouchID, Bluetooth		Root, NFC

Different between iOS and android

Features	Common	iOS	Android
language		OC\swift, dynamic	Java, static on java vm
UI Design		Apple design	Material
Core		Apple Darwin(XNU, XNIX shell)	Linux

Swift v.s. OC

- **Swift->LLVM->Objective-c->C->compilation->machine code**
- **Same project can support 4 language compile at the same time: swift, object-c, c, c++**
- **Swift depends on the ARC, GCD in objective-c**
- **Compile objective-c in swift project by bridge provide by Xcode**

- **Pros of :**

  Some new grammar much easier to use, closure, Operator overloading : var, let , DSL, Touple

  very good memory management, don't need user to take care,

- **Cons**
  - Most project core layer still need use c/c++,but swift hasn't mix well with c and c++, especially not with c++
  - Some libs are still implement by objective-c, need use bridge