Discussions on the next generation OS

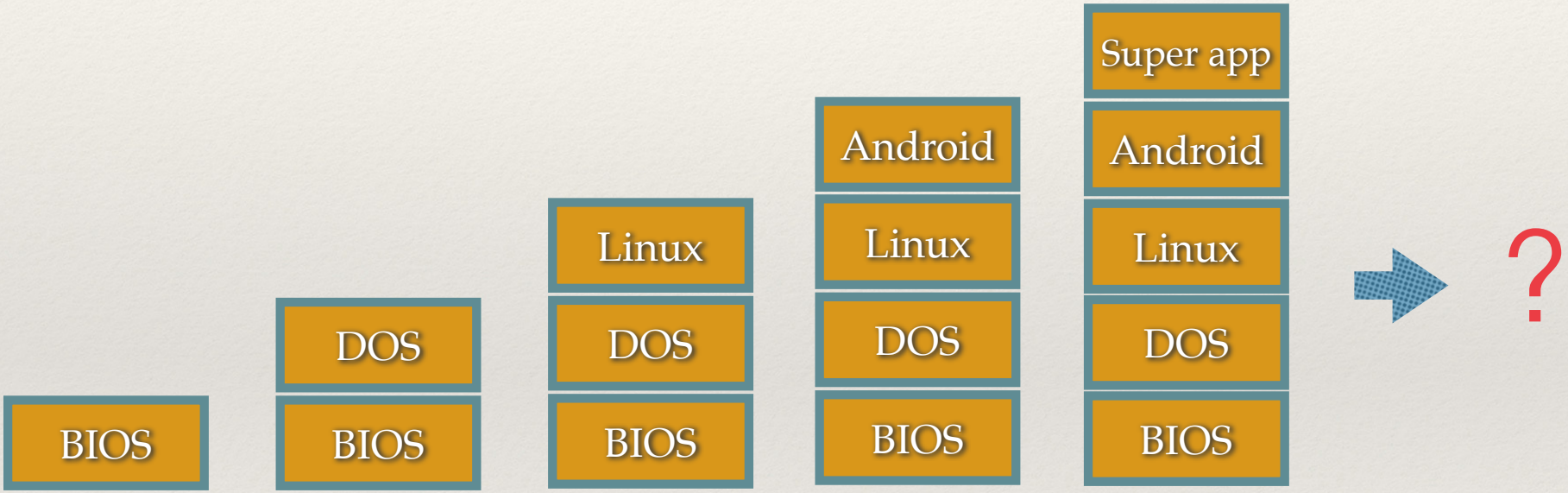
OS Design for the Future of a Connected World

Xiao-Feng Li
xiaofeng.li@gmail.com
January 2020

Agenda

- OS from individual to group
- OS from application to service
- OS from I/O to sensing
- Survey of the industry

OS design is evolving



- Trend: from hardware enabling towards user services.

OS components in the highest level

- A consumer OS is comprised of three components
  - Resource management (for vendors)
  - APIs (for app developers)
  - Interfaces (for users)

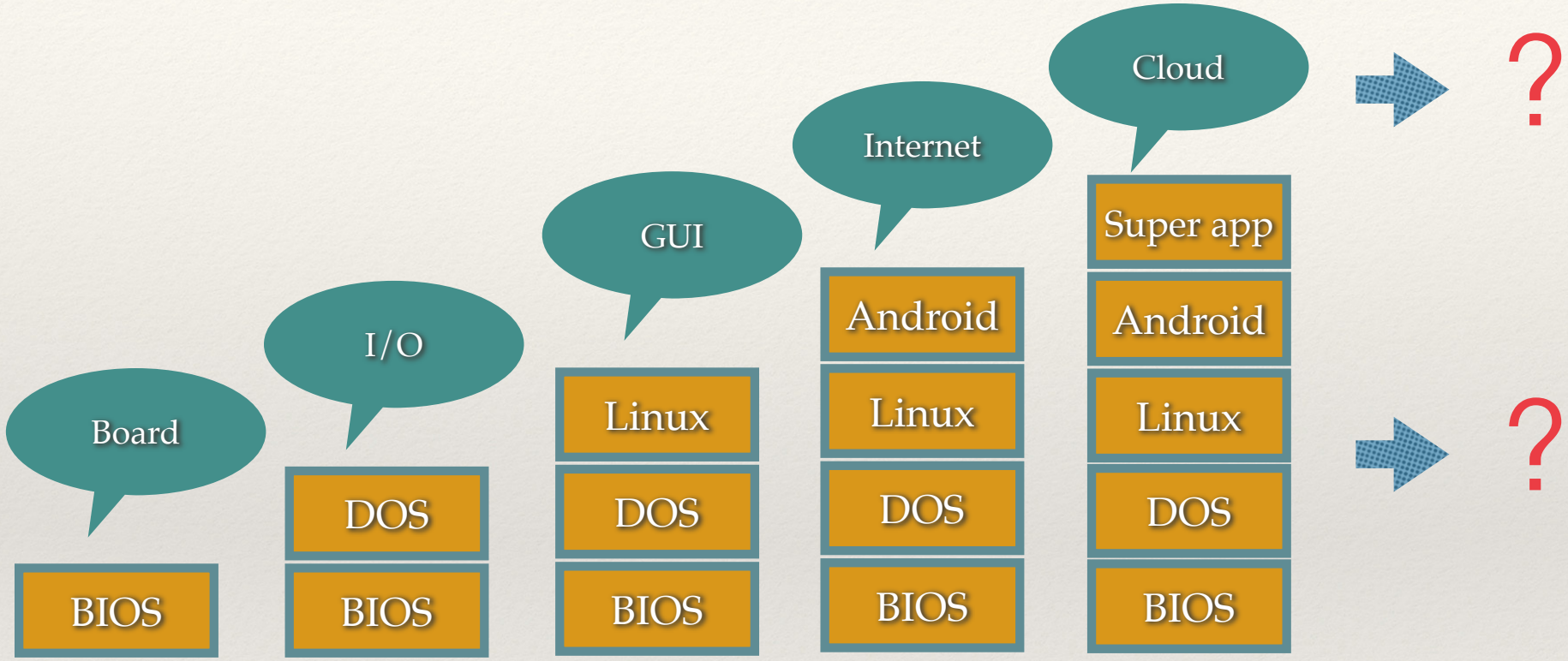
- Trend: most innovations in recent years are in app development. Methodology of hardware enabling and user interactions changes slowly.

What decides a consumer OS?

Android stack

- Why we call it Android
  - Not u-boot, little kernel, Linux, super app, although all the rest are OSes too
  - Because the primary APIs are provided by Android
    - Most apps rely on the services provided by Android framework
  - OS is decided by the primary APIs

OS service is changing



- Trend: from resource management towards daily assistant

Levels of service APIs

*Only examples, not a full list.

	Data source	Functionalities	Protocol
Process	File system	Hardware abstraction	System call
Inter-process	Data server	App features	IPC
Local	Device	P2P / AV / Sensor	WiFi / BT / NFC
Remote	Internet	Cloud services	HTTP

- Diff services use diff protocols at diff levels, because of the design legacy from ground up accumulatively

One protocol for all services

*Android status as an example.

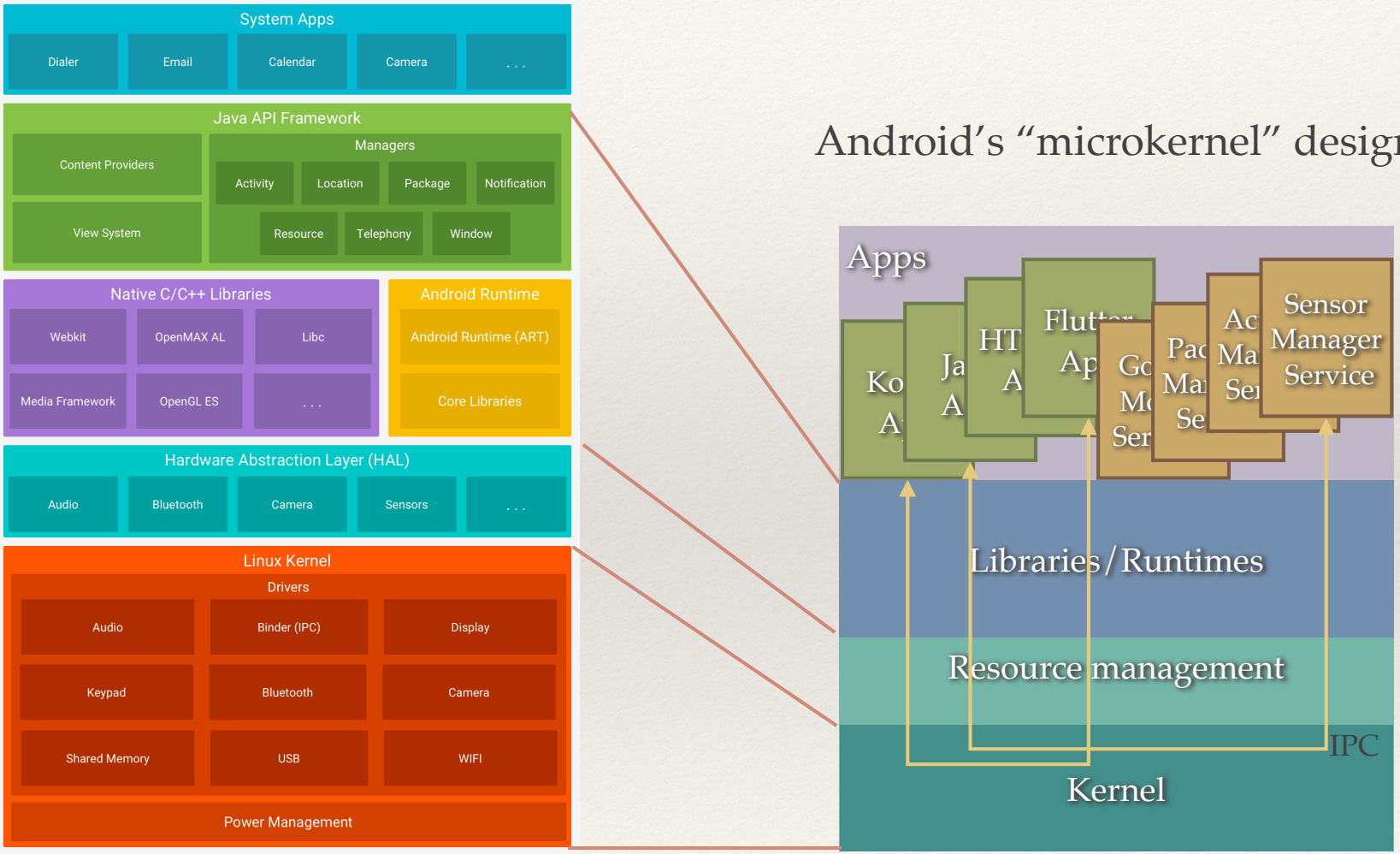
	Data source	Functionalities	Protocol
Process	File system	Hardware abstraction	HIDL
Inter-process	Data server	App features	AIDL
Local	Device	P2P / AV / Sensor	?
Remote	Internet	Cloud services	Google APIs

- Android is evolving close to a consistent protocol through Binder, except for local services

Local area protocols

- Google has Nearby / Wear / Auto / Cast protocols, specifically for different devices
- They are not general enough to various devices. A unified solution is,
  - Extend Binder to cross-devices services
  - With multi-channel Audio / Video streaming
  - With Nearby for discovery and connection
  - With multi-display for other scenarios

Another view of OS architecture



Agenda

- OS from individual to group
- OS from application to service
- OS from I/O to sensing
- Survey of the industry

Service-oriented OS

- Modern OS is around services, instead of hardware
  - Service protocol: binder IPC
  - Registration/discovery: package manager
  - Lifetime/session: activity manager
  - User interface: GUI+touch
  - External events: notification manager
  - Data service: content provider

OS: From individual to group

*Android status as an example.

OS	Individual	Group
Service protocol	IPC	Wear / Cast / Auto / …
Service discovery	package manager	Nearby / WiFi
Service session	activity manager	? (device-specific)
Service data	content provider	? (device-specific)
External events	notification manager	? (device-specific)
Service interface	GUI / touch	? (device-specific)

- Mostly other devices act as accessories to phone. Very little about p2p services or group services.

Build a group OS

OS	Individual	Group
Service protocol	IPC	RPC
Service discovery	package manager	Nearby manager
Service session	activity manager	session manager
Service data	content provider	distributed data
External events	notification manager	distributed events
Service interface	GUI/touch	voice/gesture
Meta API	Tasker/Shortcuts	DSL

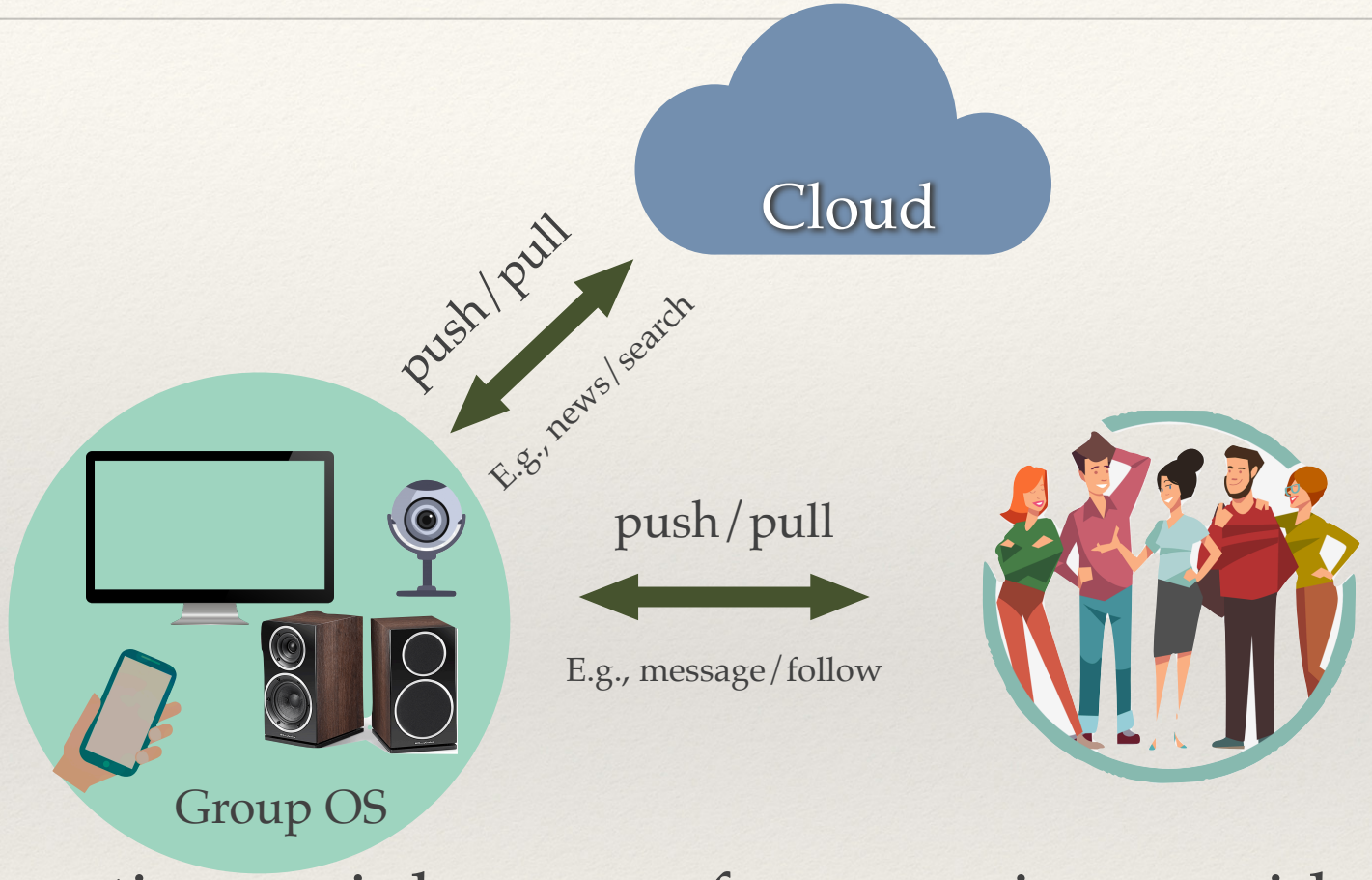
- Android can be extended for a group OS

Data are services too

- Services today only provide end-to-end integrated data
- Users should regain the control of their own data

Data source	Examples	Exposed as services
External service	apps, movies, books	DRM/blockchain
Self produced	photos, credit cards, passwords, app state	Distributed data
Runtime data	video stream, sensor data	Exposed with APIs

Services: from local to external



- Information mainly comes from service provider and social network

External service entry points

- Three main entry points
  - Pull: Search (Google, Baidu, …)
  - Push: News (Yahoo, Toutiao, …)
  - Bidirectional: Social (Facebook, Wechat, …)
- (Notes: Once dominating one entry, anxious to dominate others. Other services, e.g., maps, etc. are natural extensions.)
- OS for next generation should support the models natively

Apps readiness to external services

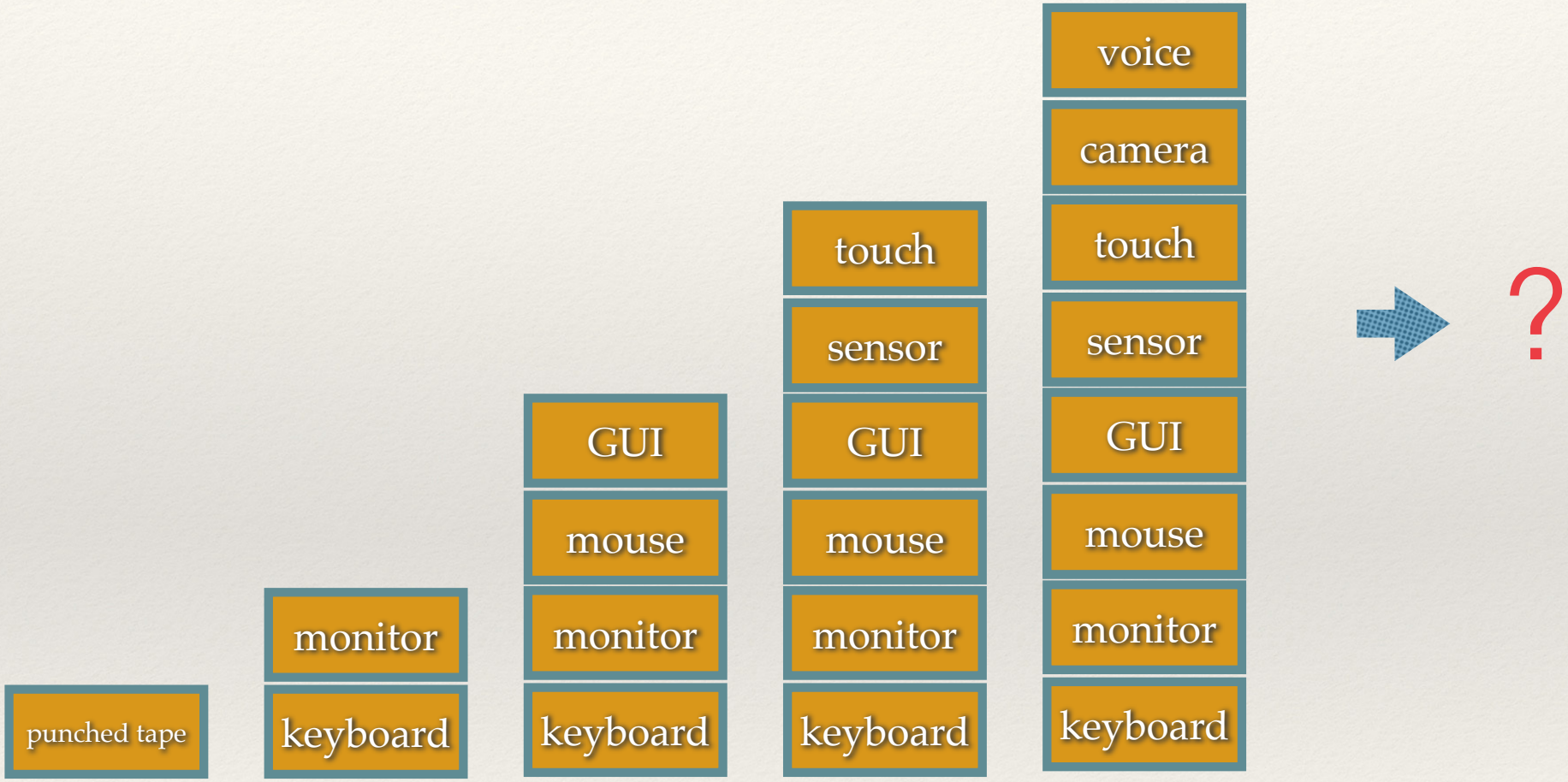
- Android app is barely external service ready.
  - H5 Applet: app ≈ doc ≈ info ≈ social ≉ cloud

	PWA	H5 Applet	Android
Page searchable	Internet	In super app	Slice in phone App in store
State recoverable	Partially	Partially	Yes
Optional install	Yes	Yes	Instant app
Cross-platform	Yes	Yes	Partially
Native access	Partially	Partially	Yes
Composable	No	No	Bundle/Binder
Distributable	Yes	Yes	Partially

Agenda

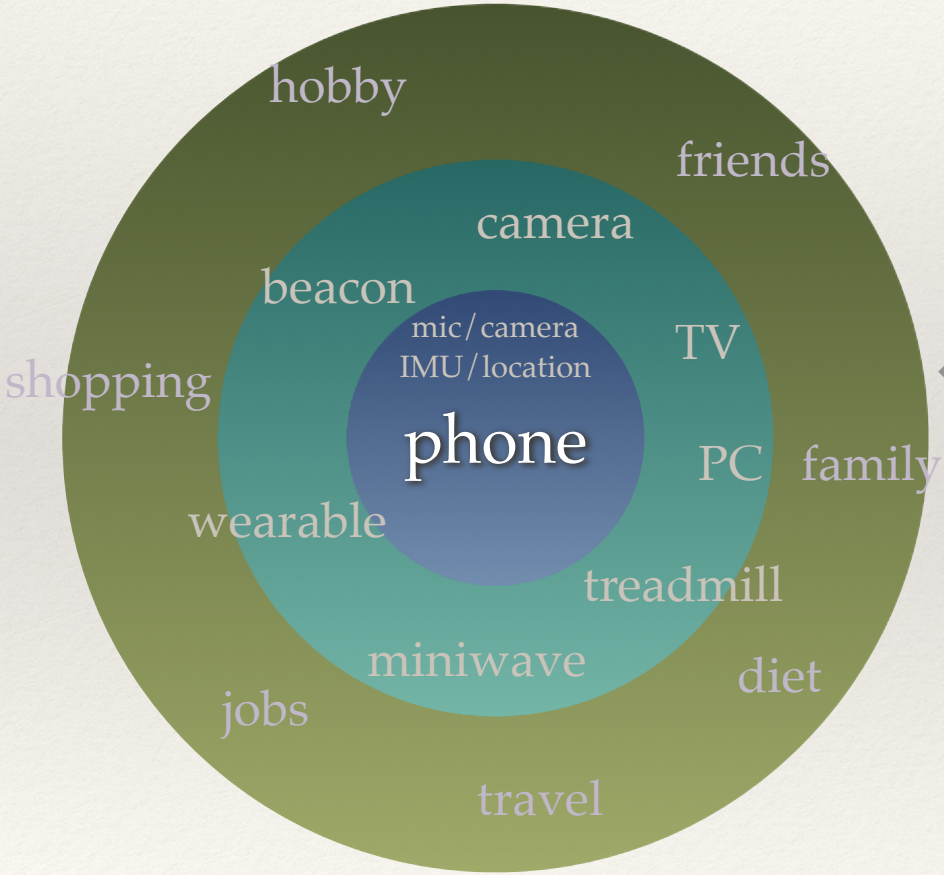
- OS from individual to group
- OS from application to service
- OS from I/O to sensing
- Survey of the industry

HCI is from I/O to sensing



- User interface becomes more and more natural

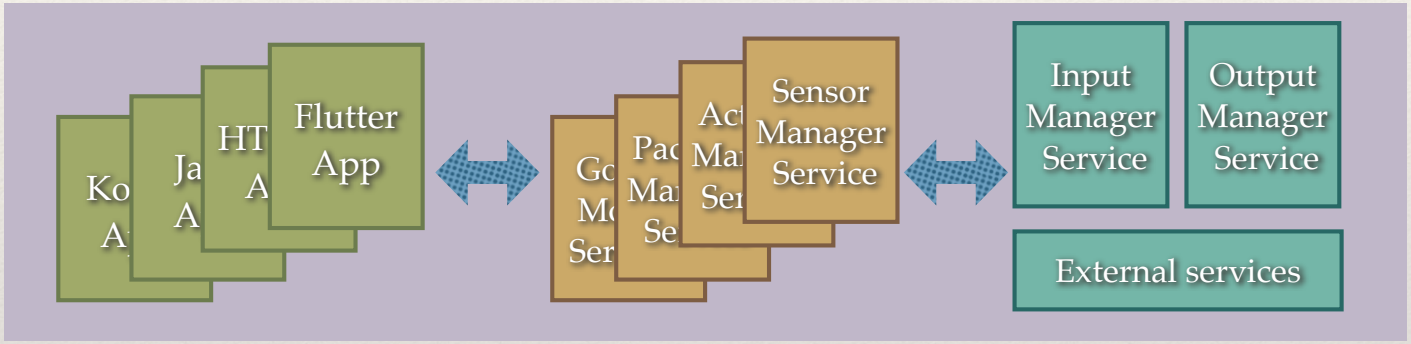
Context-aware OS



- OS has to receive inputs from devices around it, and present outputs to suitable devices
- Inputs are not necessarily immediate info, can be long term profile

Inputs/outputs are services

- Input - computing - output should be decoupled and distributed



- Application development still follows MVC-class models, and here V is essentially based on I/O services
  - OS APIs provide seamlessly access to the decoupled M-V-C service providers

Remote View

- Each has their respective use cases:
  - (Virtual) display mirror
  - Remote frame-buffer / desktop
  - Remote rendering
  - UI client distribution
- Align the inputs, synchronize the outputs
  - Critical for AV experiences

Service semantics

- Services are mostly binary interfaces
  - Not human readable
  - Not web searchable
  - Not directly map to human activities
- Services should be defined with semantics
  - Describe services with human readable tokens
  - Services can be connected according to the tokens

DSL as high level APIs

- DSL program can be developed or generated
  - Programmable like IFTTT
  - Generated from human multi-modal instructions
    - Human instructions ➡ Sequence of semantic tokens ➡ DSL program ➡ OS service operations
    - AI engine is mandatory
- DSL program can be converted to common app
  - Managed by app store and is distributable

Abstraction and modularity

- Context-aware OS involves lots of dynamic resources and states
  - Device loads components according to context
  - Different devices loads different components
  - Devices can use different OSes but agree on protocols
- Considerations
  - System update, app migration, API stability, real time, account, security, etc.

Agenda

- OS from individual to group
- OS from application to service
- OS from I/O to sensing
- Survey of the industry

Summary

- Design the next generation OS based on existing one, by enhancing the capabilities of,
  - Distributed and connected
  - Service-oriented APIs
  - Natural interactions