BlueMountain: An Architecture to Customize Data Management on Mobile Systems*

Sharath Chandrashekhara, Taeyeon Ki, Kyungho Jeon, Karthik Dantu, and Steven Y. Ko

(*To appear in MobiCom 2017, Utah, USA, Oct 16th – 20th)

Mobile Apps Today

➢ Complex and powerful apps
➢ Rich data-centric interactions
➢ Use various clouds, local storage, and personal devices to sync user and app data
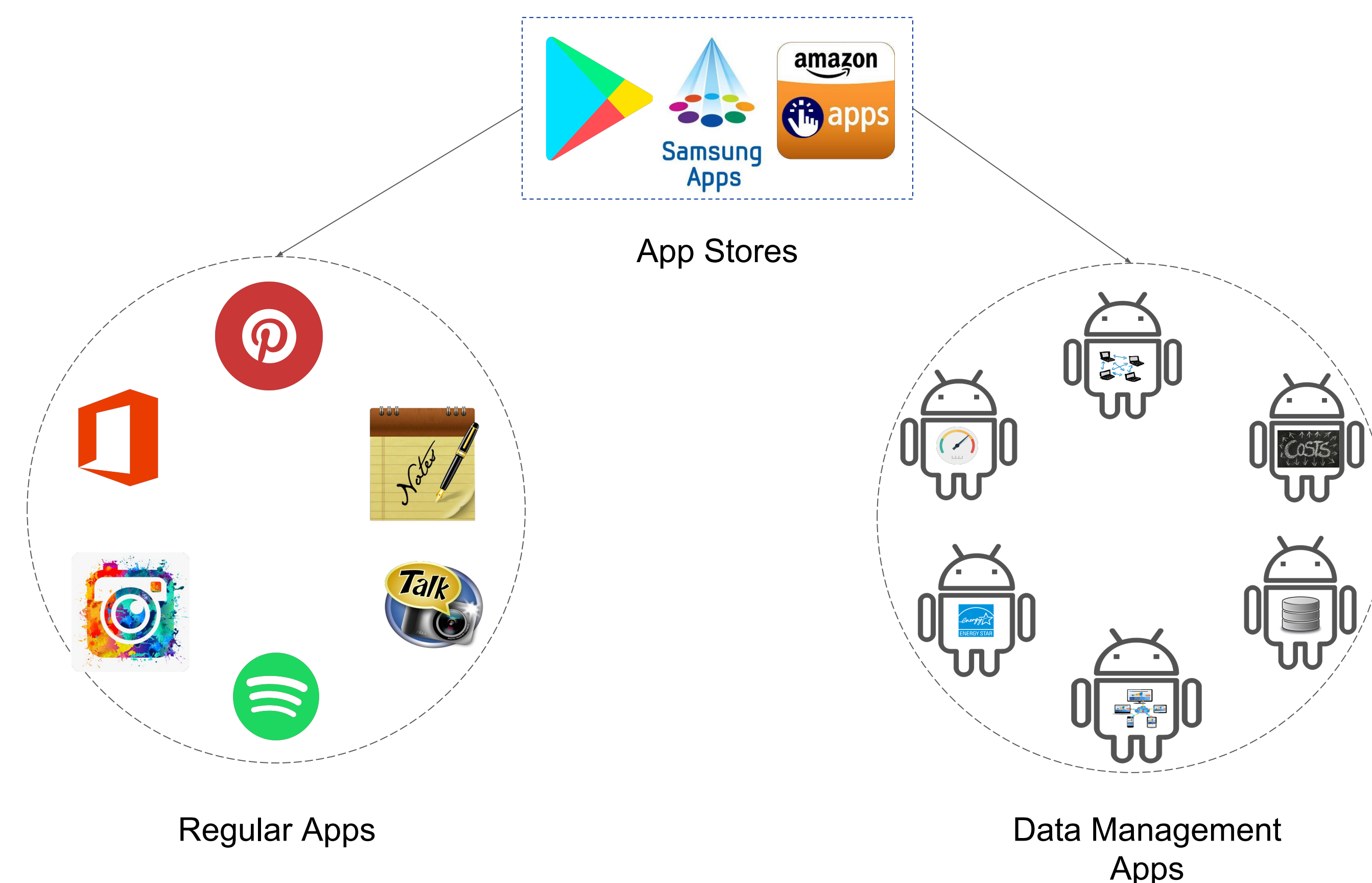
Problems

➢ Developers manually craft the data management logic into each app!
➢ App logic and data management logic are intertwined.
➢ Users don't have control over their data.

Repetitive and difficult for developers; inflexible for users; hard to deploy new data management solutions!
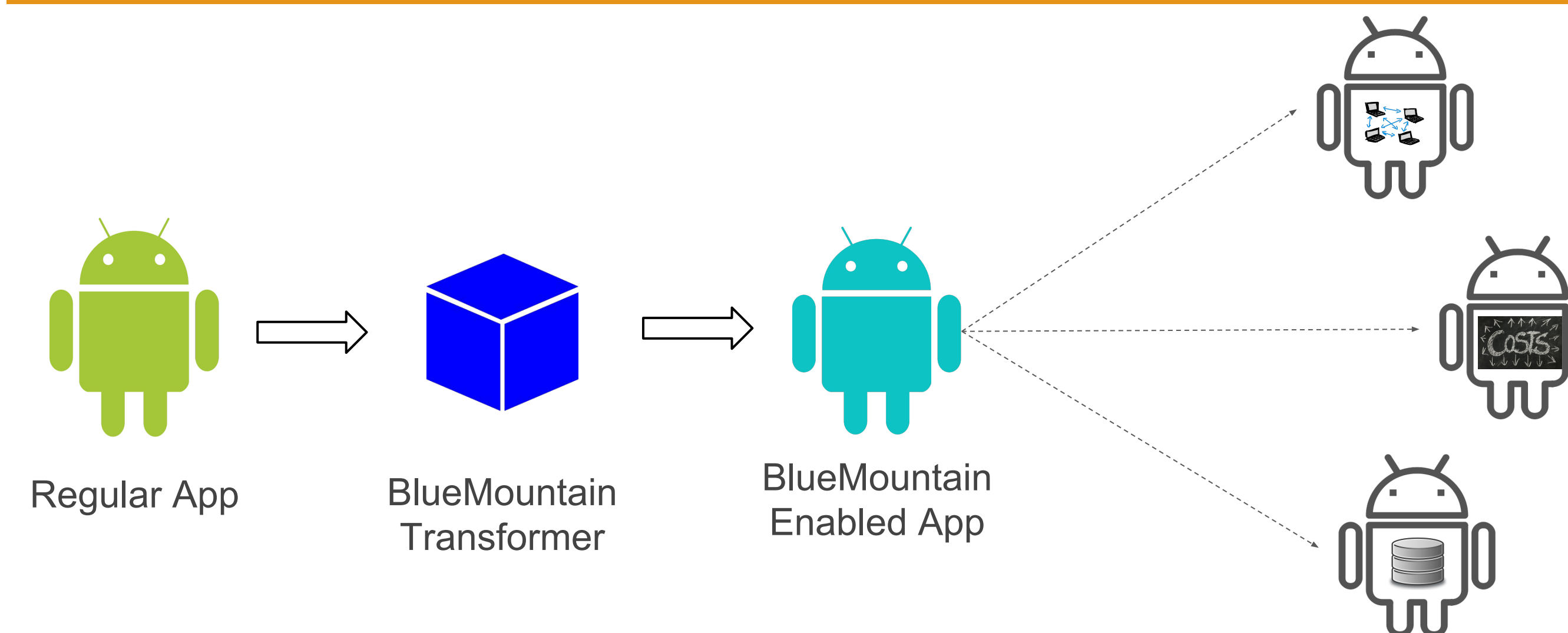
BlueMountain Vision

App Stores

Regular Apps

Data Management Apps

➢ Two kinds of user installable apps
➢ Regular app implements only core-app logic
➢ *Data management app* manages the data and data policies
➢ Users can link any *data management app* with any regular app!

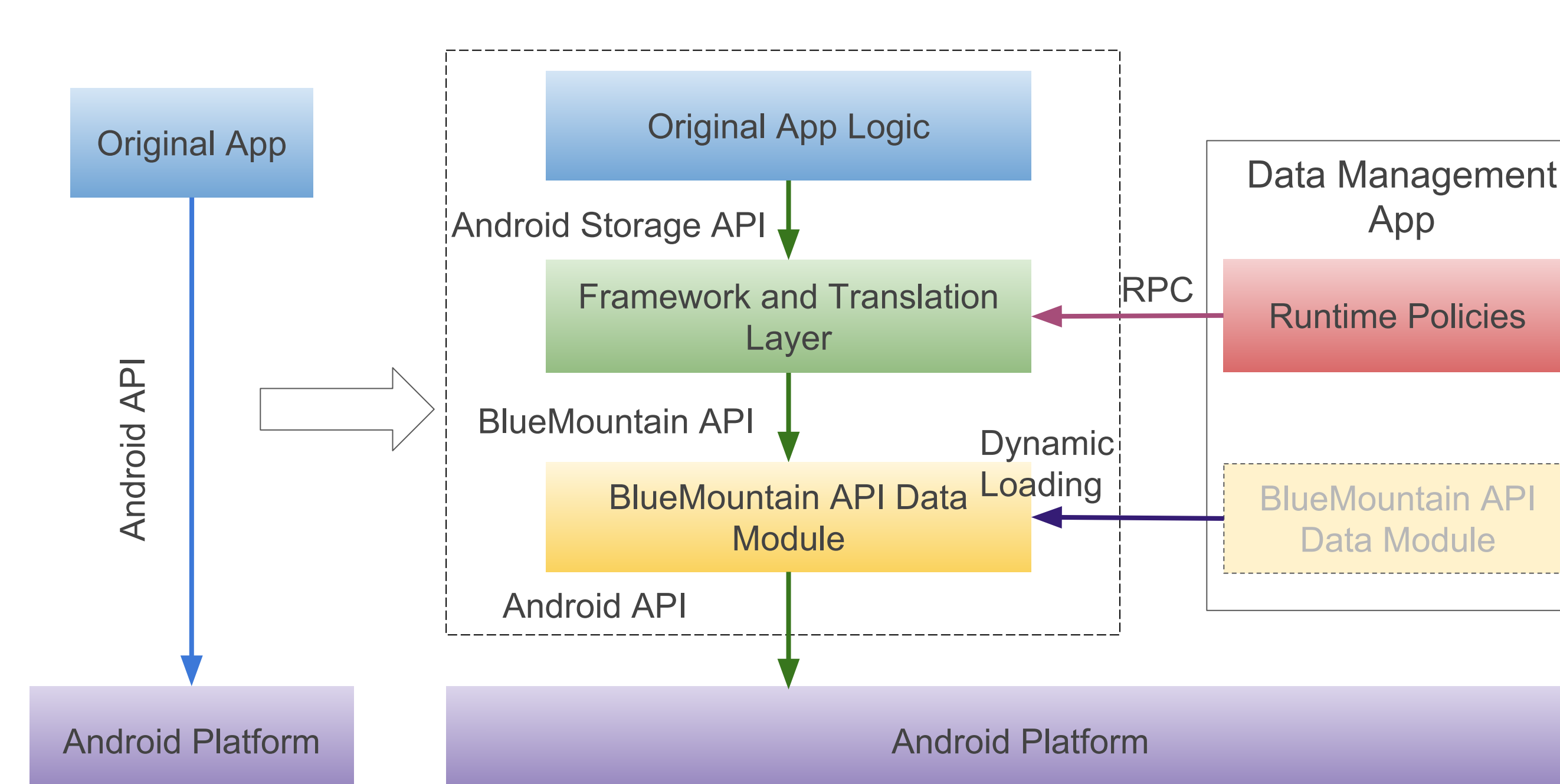*Easy to develop apps; users have more control; easy to deploy new data management solutions!*

Workflow

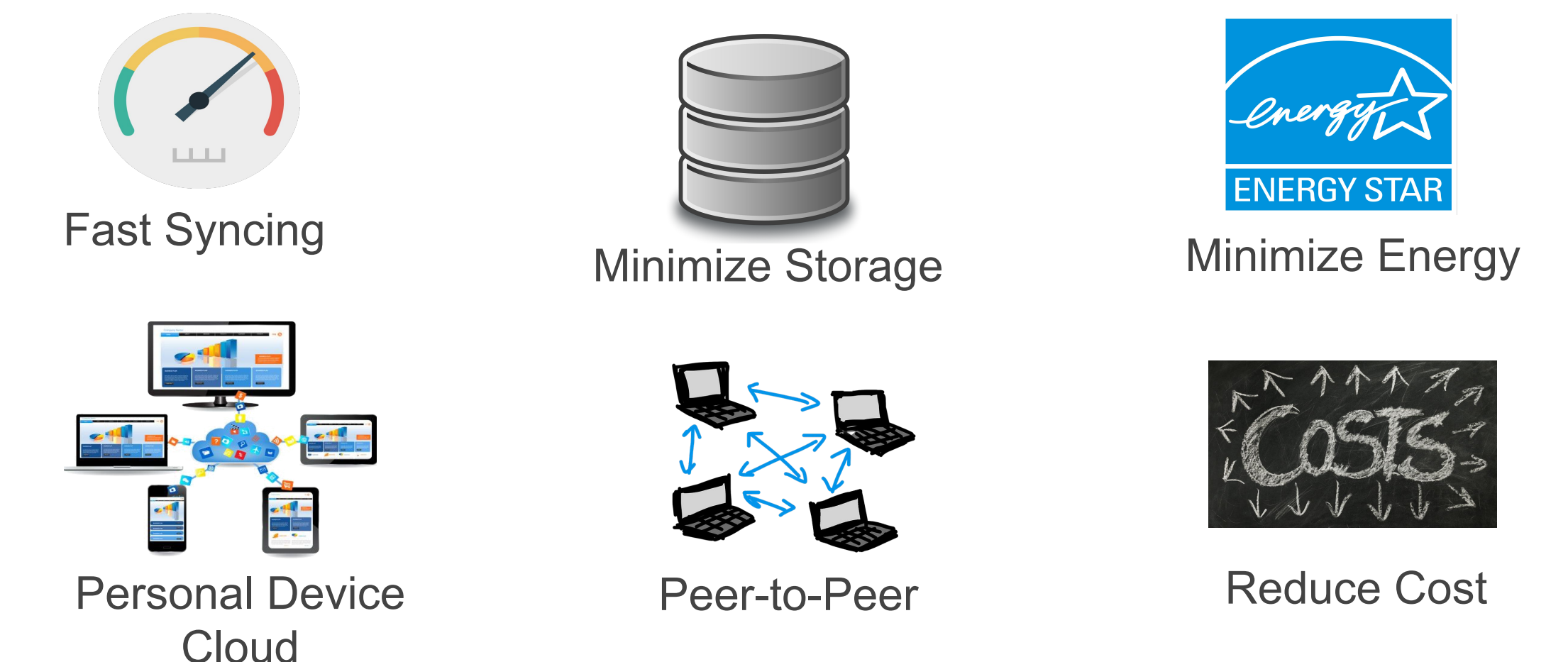Regular App → BlueMountain Transformer → BlueMountain Enabled App

➢ BlueMountain transformer, based on *Reptor* [1], can transform any app into a '*BlueMountain enabled app*' by injecting our framework into the app
➢ '*BlueMountain enabled apps*' can be linked with any *data management app* by the users based on their needs.

Architecture

Original App · Original App Logic · Android Storage API · Framework and Translation Layer · BlueMountain API · BlueMountain API Data Module · Android API · Android Platform · RPC · Dynamic Loading · Data Management App · Runtime Policies · BlueMountain API Data Module · Android Platform · Android API

➢ **BlueMountain API:** All *data management apps* implement this
  ○ Files: Object based API, suitable for cloud
  ○ Database: SQL statement based API
  ○ Key/Value pair: Developed on top of files
➢ **Translation Layer:** Translates Android APIs to BlueMountain APIs
➢ **Dynamic Loading:** Data management modules are loading into an app at runtime and works in the same address space.
➢ **Runtime policies:** Can be set by IPC (Android Binder framework)

Use Cases

Fast Syncing · Minimize Storage · Minimize Energy

Personal Device Cloud · Peer-to-Peer · Reduce Cost

➢ Custom data management apps based on previous work can be implemented.
➢ We have implemented five *data management apps* for BlueMountain
  ○ File sync systems based on PSCloud [2] & QuickSync [3]
  ○ Database sync systems based on *Operation logs, File sync & Multiple-databases.*

Evaluation

Use Case Testing

➢ *Photo Talks* app, a popular photo editing app on Android, is transformed to a BlueMountain enabled app.
➢ Can be linked with any of our test *data management apps.*
➢ Photos are redirected from local storage to desktop and Dropbox!
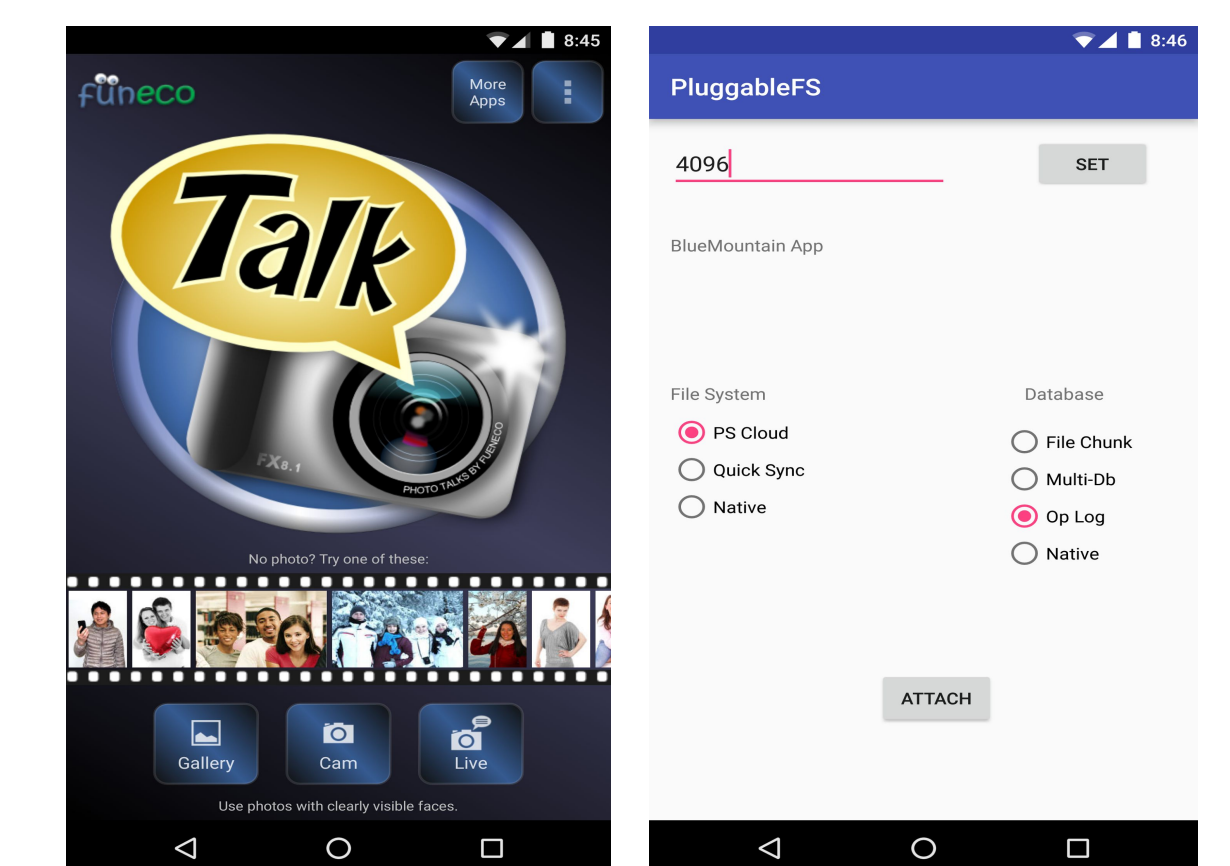
Tests and Instrumentation

➢ Tested with 400 apps downloaded from Google Play.
➢ All apps were instrumented by injecting BlueMountain framework.
➢ Overhead due to translation, framework and dynamic loading indicate a reasonable overhead of 10-30%.

References

1. Ki, T., et al. "Reptor: Enabling api virtualization on android for platform openness." In MobiSys 2017
2. Bazarbayev, S., et al. "Pscloud: a durable context-aware personal storage cloud." In HotDep 2013
3. Cui, Y., et al. "QuickSync: Improving Synchronization Efficiency for Mobile Cloud Storage Services." In MobiComm, 2015
4. Chandrashekhara, S., et al. "Enabling automated, rich, and versatile data management for android apps with Bluemountain." In HotStorage 2015