An Empirical Study of API Stability and Adoption in the Android Ecosystem

Tyler McDonnell, Baishakhi Ray and Miryung Kim
The University of Texas at Austin

Motivation

- Despite the benefit of new or updated APIs, developers are often slow to adopt new APIs.
- API evolution and its associated ripple effect throughout software ecosystems are still under-studied.

Study Findings

- We study the **co-evolution** of Android APIs and applications using the github data
  - Android is evolving fast at a rate of 115 API updates per month.
  - 28% of API references in client apps are outdated with a median lagging time of 16 months.
  - API usage adaptation code is **defect prone** than other code.

Outline

- Motivation & Related Work
- Study Approach
- Research Questions and Results
- Limitations
- Conclusions

Related Work

- Many techniques have been proposed to ease API update and version incompatibilities
- API evolution and its associated ripple effect through ecosystems are under-studied
  - Robbes et al. study how API deprecation affects client applications in Smalltalk.
- Kim et al. study the relationship between API refactoring and bugs in libraries.

Study Approach

❶ Android API Version History

❷ Mobile Apps in Github

❸ Correlate changes in mobile apps with changes in Android OS

API Version: 14

Release date: October 19, 2011

Class: android.widget.RemoteViews

void setRemoteAdapter(int, Intent)

Android API Version History

Client Code : Remote.java

Commit Date: January 26, 2012

import android.widget.RemoteViews;

```
int viewID = settings.getViewID();
Intent I = new Intent(this,
ActivityTwo.class);
```

setRemoteAdapter(viewID, I);

Client Source Code

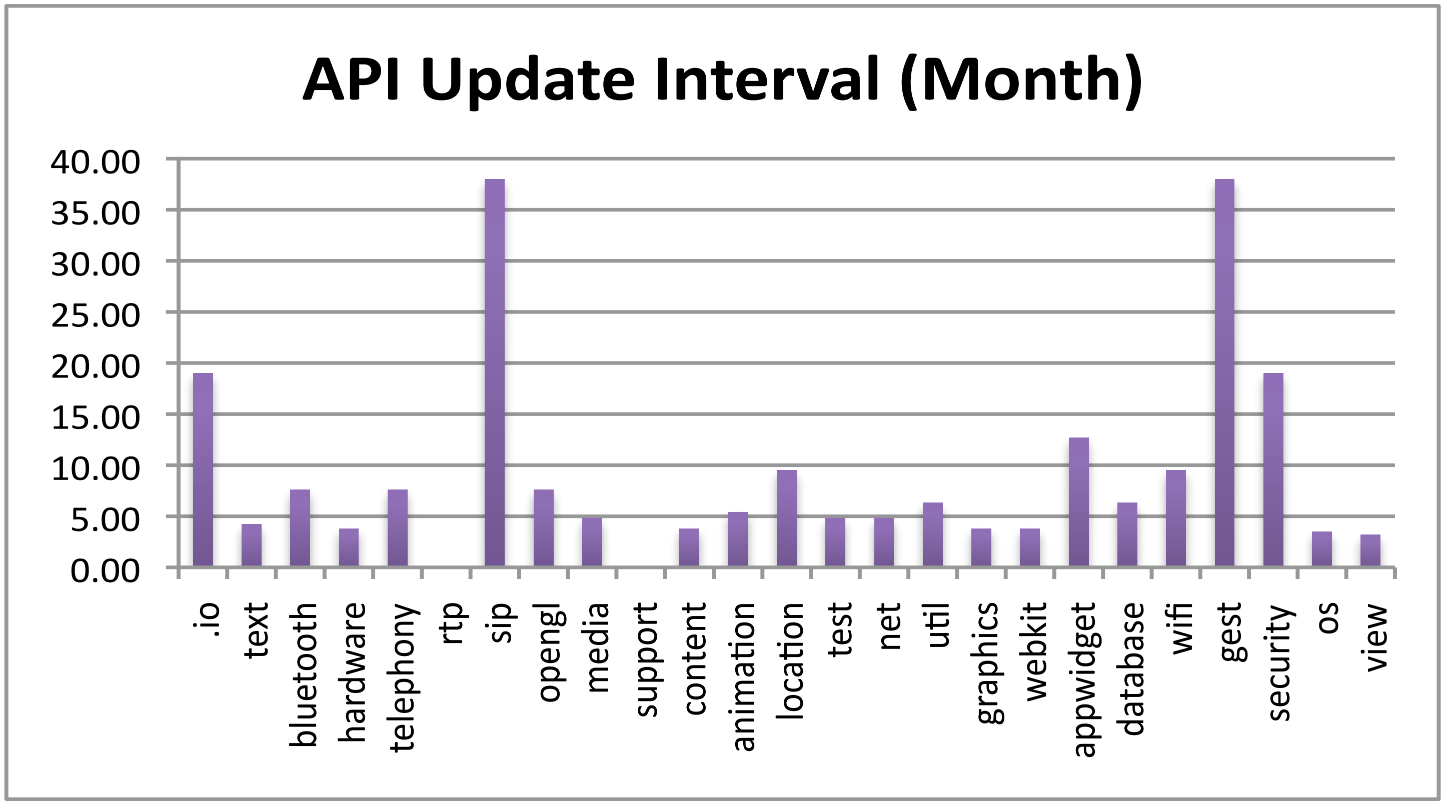
Android OS API Evolution Characteristics

- API Version 3 to 15

	Class	Method			Fields		
	Δ	Δ	+	-	Δ	+	-
Min	37	0	0	0	7	0	0
Max	269	416	98	9	619	205	0
Avg	149	158	37	2	179	32	0
Rate	42	44	11	<1	51	9	0

Android OS is evolving fast at the rate of 115 API updates per month.

Android API Evolution Characteristics



Hardware, user interface and web support are evolving fast.

Data Sets : Mobile Apps

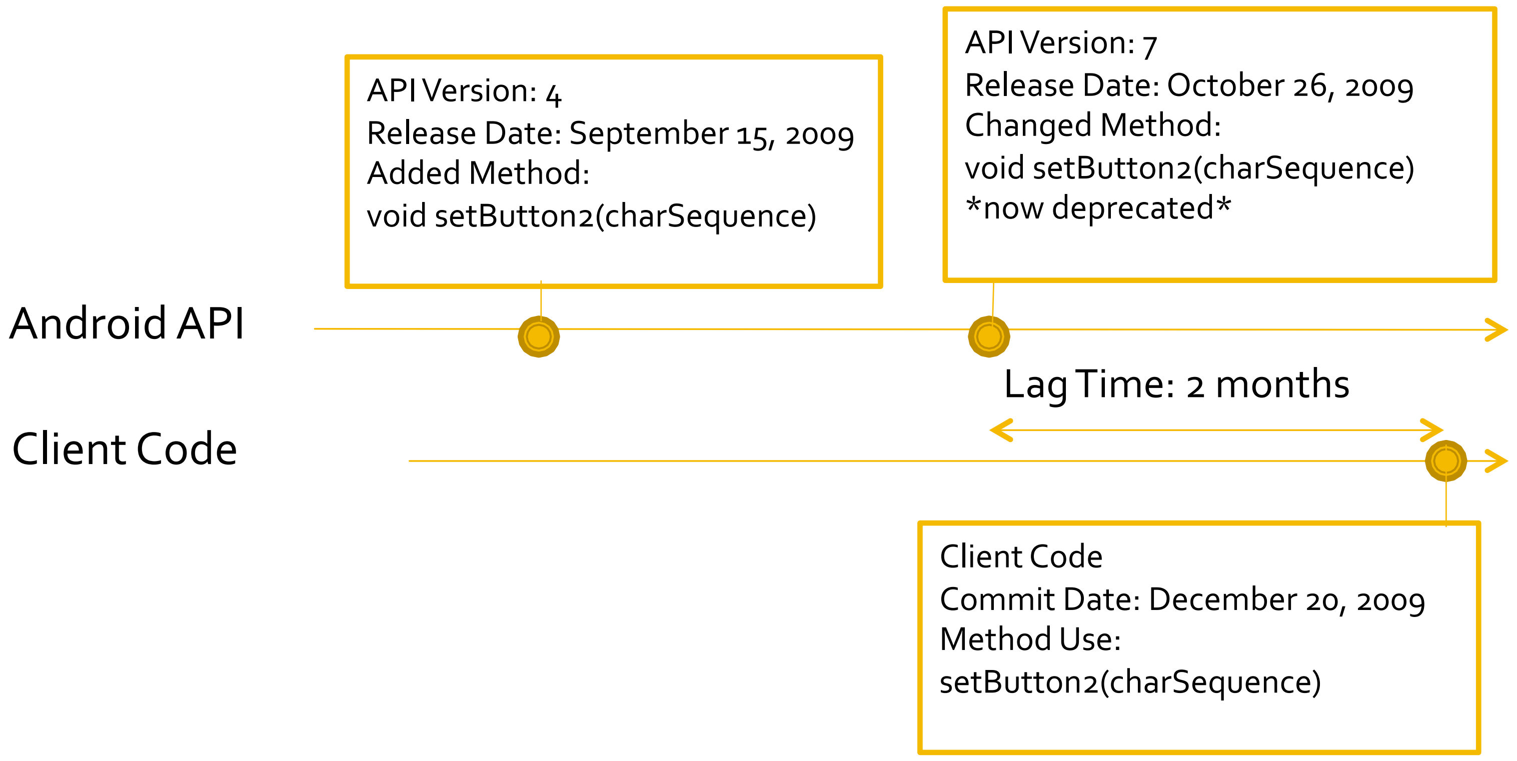
	Revision	LOC	Author	% Android Refs.
Congress Tracker	1359	13349	7	30%
Apollo M	9	15783	1	35%
Cyanogen	109	28972	20	24%
Google Analytic	926	52932	23	26%
LastFM	212	9771	7	16%
mp3Tunes	104	9608	1	22%
OneBusAway	497	51784	5	22%
ownCloud	665	25109	12	30%
RedPhone	116	21315	5	19%
XMBCremote	928	92893	24	22%

Around 25% of all method and field references in client code use Android APIs.

Research Questions

- Q1: What is the lag time between client code and the most recent Android API?
- Q2: How quickly do API changes propagate throughout client code?
- Q3: What is the relationship between API updates and bugs in clients?
- Q4: What is the relationship between API stability and adoption?

Q1: What is the lag time between client code and the most recent Android API?

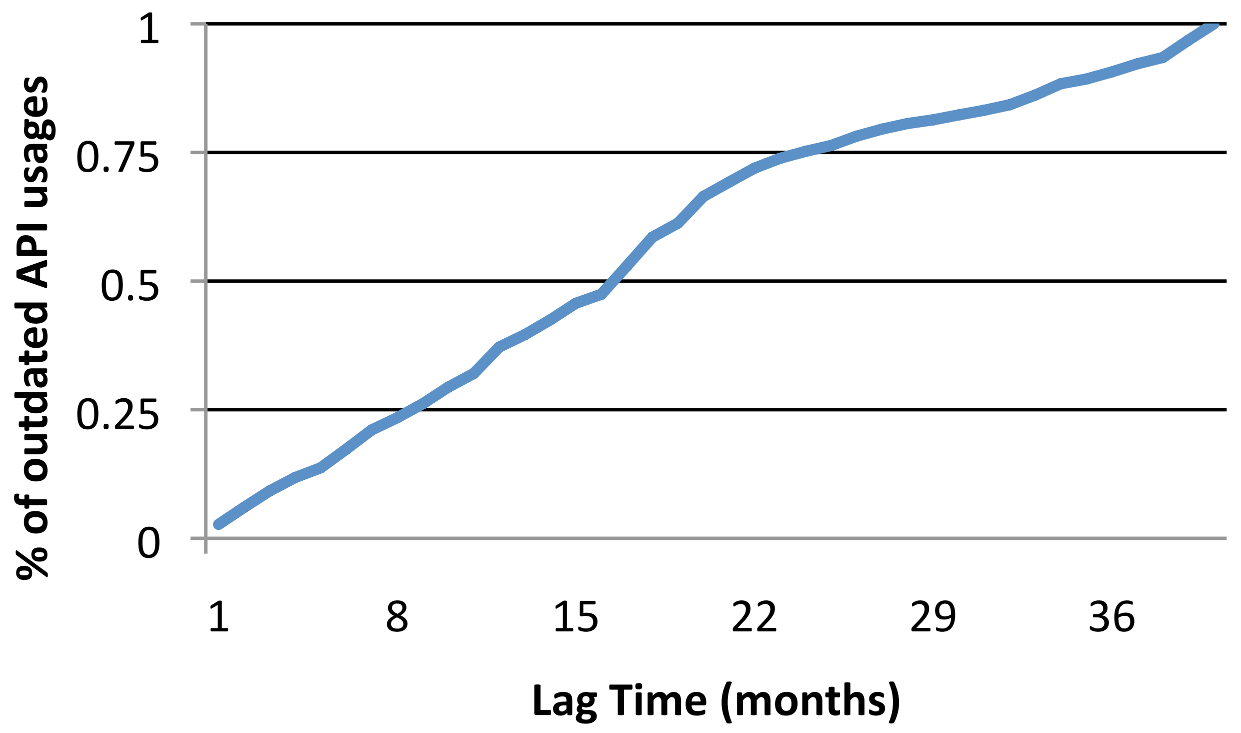


Lag time: the number of months elapsed between the release of the new version and the commit time of the outdated API usage

Q1: What is the lag time between client code and the most recent Android API?

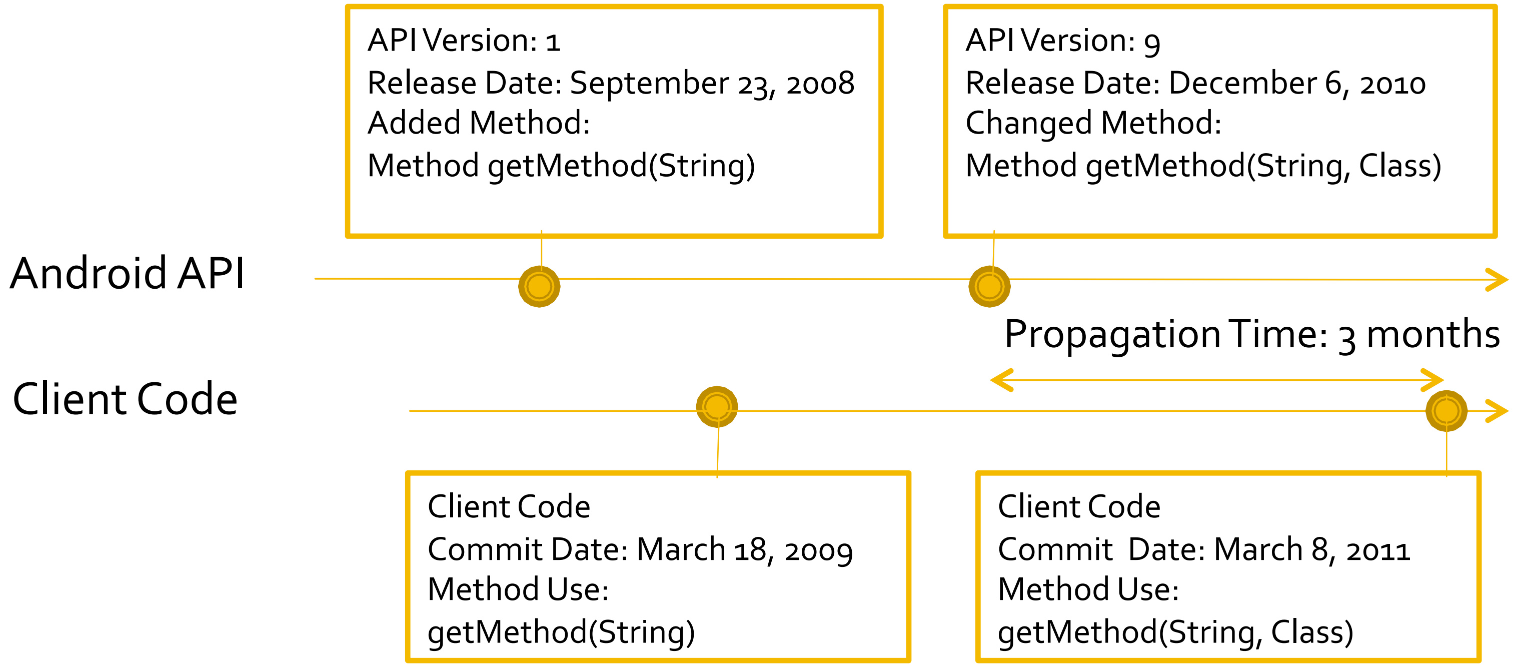
	Lagging API references(%)
Congress Tracker	18%
Apollo M	72%
Cyanogen	12%
Google Analytic	37%
LastFM	43%
mp3Tunes	5%
OneBusAway	3%
ownCloud	18%
RedPhone	43%
XMBCremote	15%
Average	28%

Q1: What is the lag time between client code and the most recent Android API?



A half of all outdated API references are lagging behind by 16 months or more.

Q2: How quickly do API changes propagate throughout client code?

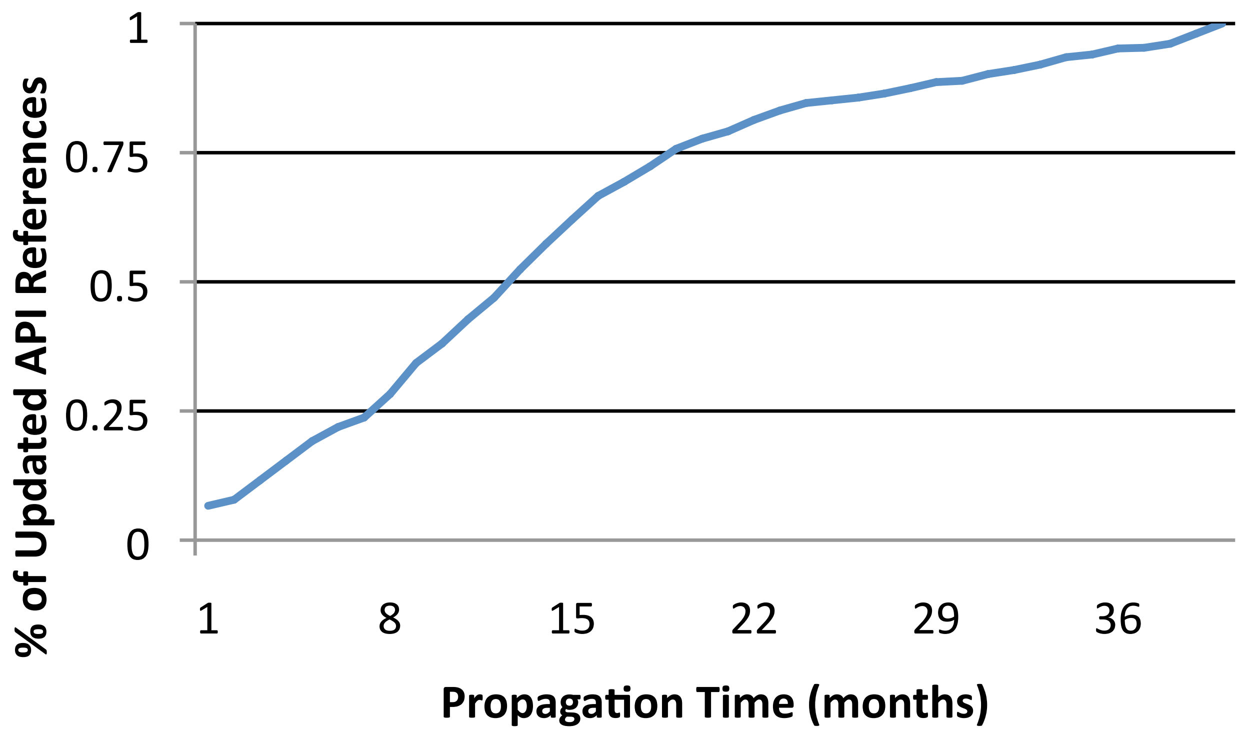


Propagation time: time difference in months between the API release and the timing of client adaptation

Q2: How quickly do API changes propagate throughout client code?

	% of outdated usages that were upgraded to use newer APIs
Congress Tracker	45%
Apollo Music	0%
Cyanogen	27%
Google Analytic	34%
LastFM	5%
mp3Tunes	0%
OneBusAway	12%
ownCloud	29%
RedPhone	39%
XMBCremote	33%
Average	22%

Q2: How quickly do API changes propagate throughout client code?



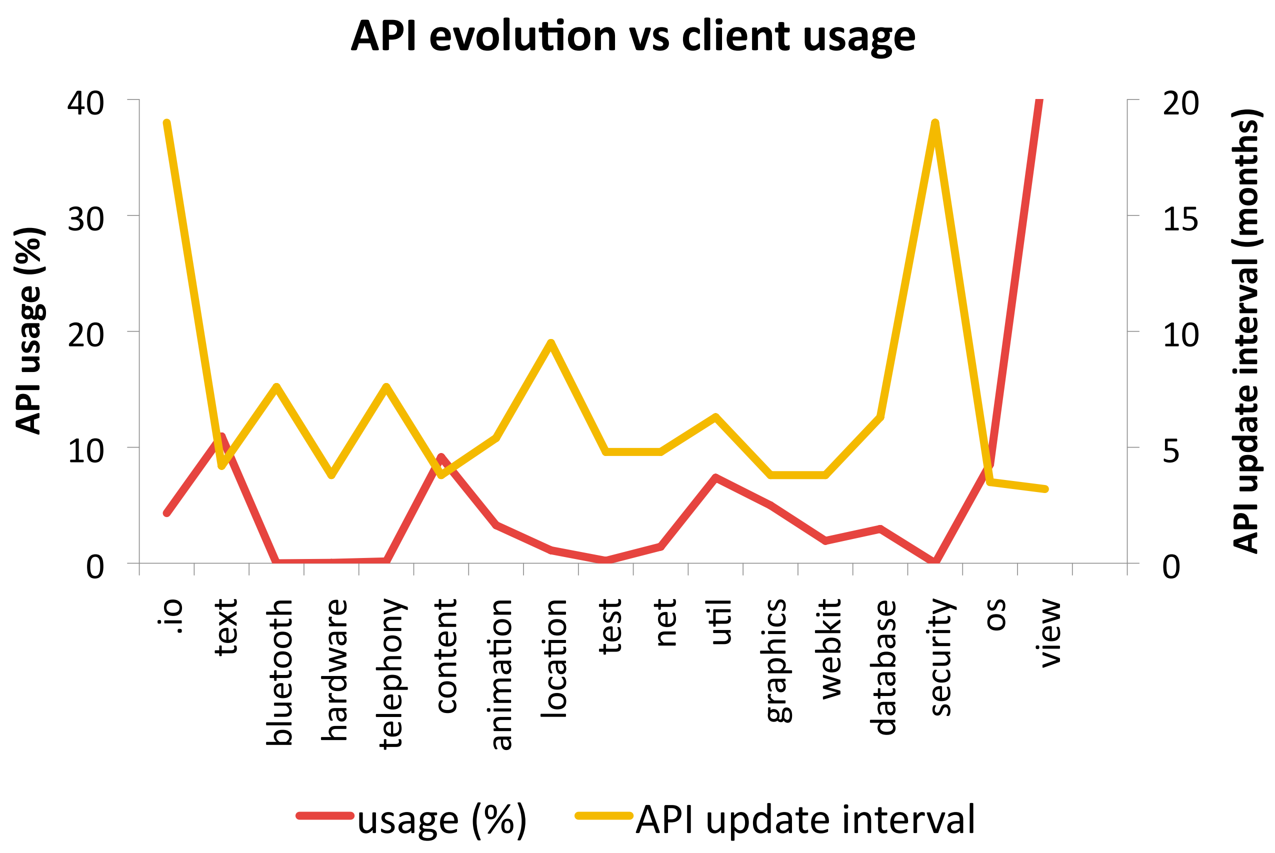
The median propagation time is 14 months. Outdated API usages upgrade to newer APIs but at a much slower pace than the API release rate.

Q3: What is the relationship between API updates and bugs?

	Spearman Correlation with bugs		
	CLOC	API Update	Non API Update
Congress Tracker	0.39	0.56	0.39
OneBusAway	0.26	0.46	0.25
RedPhone	0.23	0.24	0.23
XMBCremote	0.34	0.62	0.33
Google Analytic	0.36	0.54	0.31
ownCloud	0.43	0.55	0.42
Cyanogen	0.58	0.63	0.58
LastFM	0.42	0.37	0.43

Files with API usage adaptations are defect-prone in all applications except LastFM.

Q4: What is the relationship between API stability and usage?



Correlation between API usage (%) and API update interval: -0.47
Fast evolving APIs are used more by clients.

Study Limitations and Future Work

- False negatives and positives in detecting API usage updates.
- Our method of detecting lagging methods does not take into account multi-version API support.
- We study the correlation between API usage, adoption, and bugs, but not causation.
- External validity beyond studied mobile apps from github.

Summary and Future Work

- We study on the co-evolution of Android OS and its clients.
  - 28% of Android references are lagging behind the latest version with a median lagging time of 16 months.
  - 22% of outdated API references upgrade to use newer APIs. The median propagation time is 14 months.
  - Fast-evolving APIs are used more.
  - API updates are more defect prone than other types of changes in client code.

Summary and Future Work

- Various stakeholders affect the process of API adoption in the software ecosystem. We need to identify factors affecting API adoption.
- Our goal is to automate required API adaptations in client applications using our example-based program transformation approach [Meng et al. 2013.]

Questions?