UCI University of California, Irvine

Improving the Quality of Software Using Testing and Fault Prediction

Professor Iftekhar Ahmed
Department of Informatics
https://www.ics.uci.edu/~iftekha/

The Ariane Rocket Disaster (1996)

https://youtu.be/PK_yguLapgA?t=50s

Root cause

- Caused due to numeric overflow error
  - Attempt to fit 64-bit format data in 16-bit space
- Cost
  - $100M's for loss of mission
  - Multi-year setback to the Ariane program
- Read more at http://www.around.com/ariane.html

Program invariants

- Invariant is a program fact that is true in every run of the program.
- An invariant at the end of the program is (z == c) for some constant c.
- What is c?

```
int p(int x) { return x * x; }

void main() {
    int z;
    if  (getc() == ‘a’)
        z = product(6*6) + 6;
    else
        z = product(-7*-7) – 7;
                                z=?
}
```

Program invariants

- Invariant is a program fact that is true in every run of the program.
- An invariant at the end of the program is (z == c) for some constant c.
- What is c?

```
int p(int x) { return x * x; }

void main() {
    int z;
    if  (getc() == ‘a’)
        z = product(6*6) + 6;
    else
        z = product(-7*-7) – 7;
     if (z != 42)
        disaster();
}
```

z=42

Disaster averted!

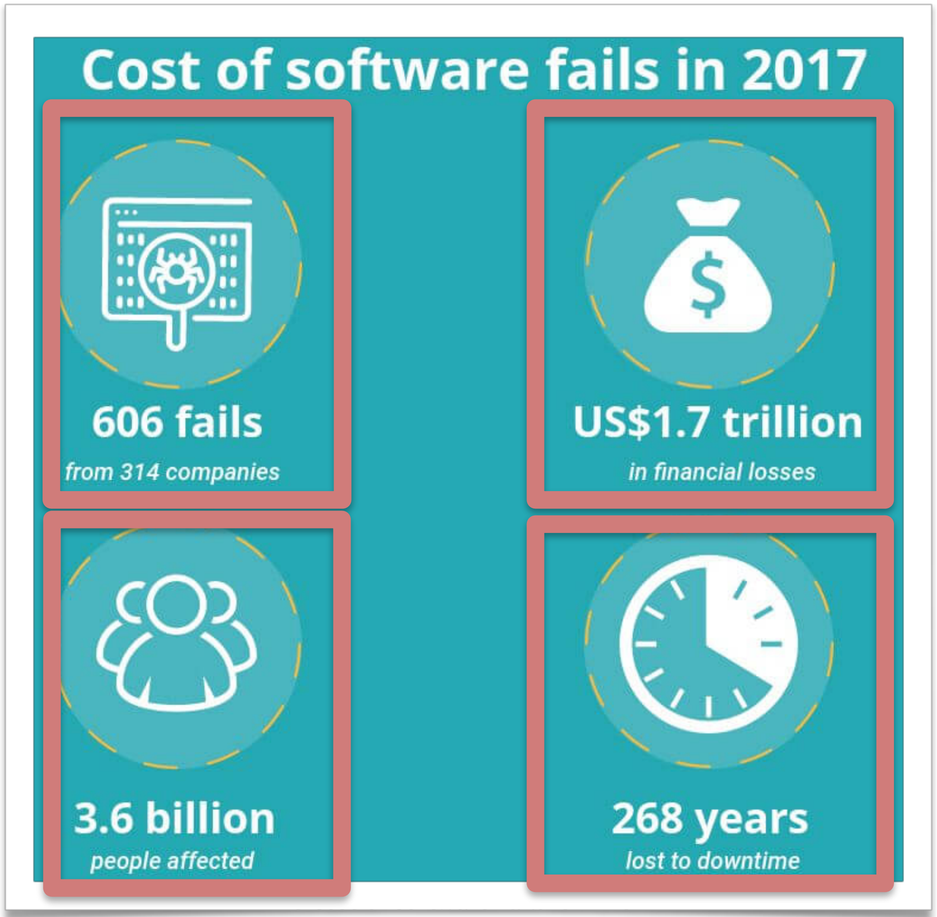
Software is a critical part of our life

Source: https://pbs.twimg.com/media/DWwOtruVMAAh1sD.jpg

Cost of software failure is increasing

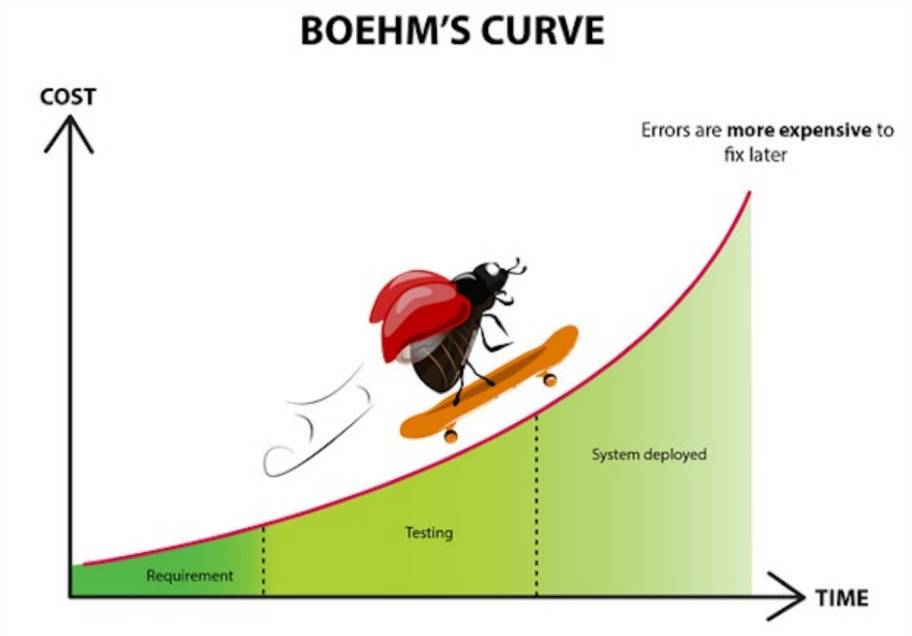
Ethiopian airlines flight 302 (Boeing 737 Max)- March 10, 2019

Uber - March 18, 2018

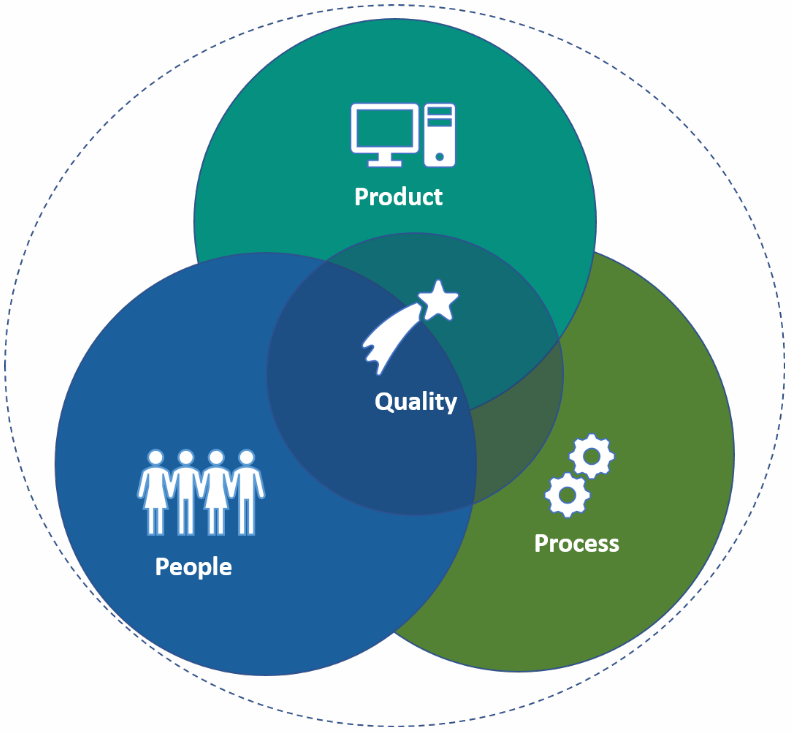


Source:Software Fail Watch

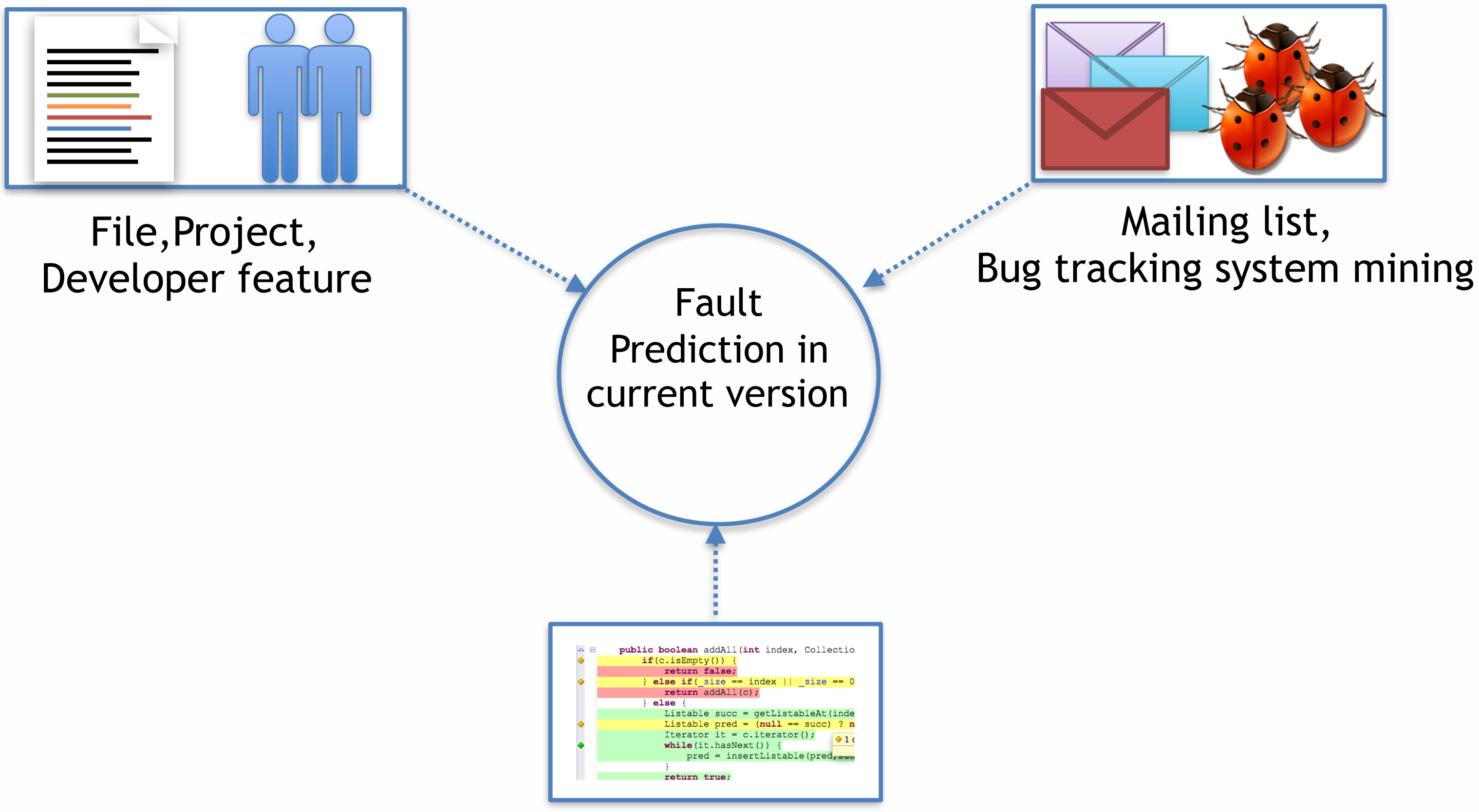
What do we do to make software better ?



My research



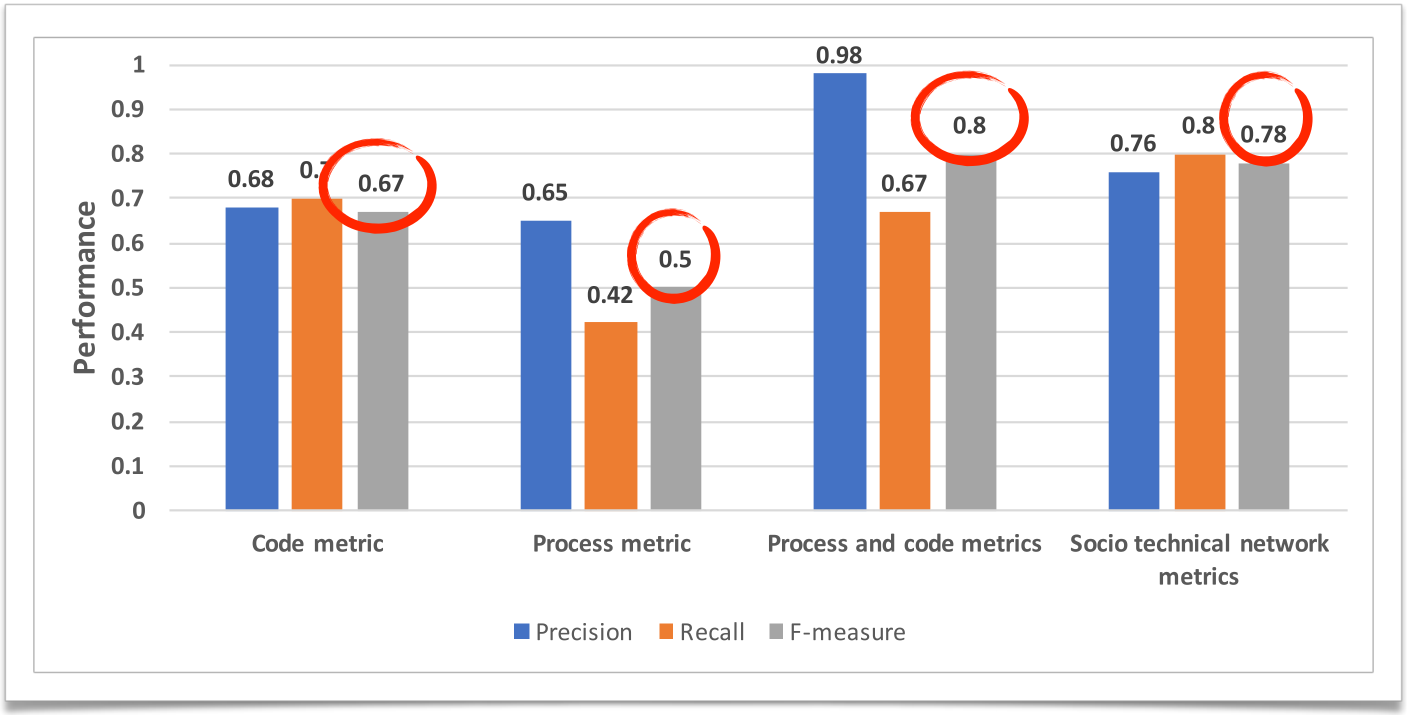
Fault prediction using factors impacting code quality



```
public boolean addAll(int index, Collectio
    if(c.isEmpty()) {
        return false;
    } else if(_size == index || _size == 0
        return addAll(c);
    } else {
        Listable succ = getListableAt(inde
        Listable pred = (null == succ) ? n
        Iterator it = c.iterator();
        while(it.hasNext()) {
            pred = insertListable(pred,suc
        }
        return true;
```

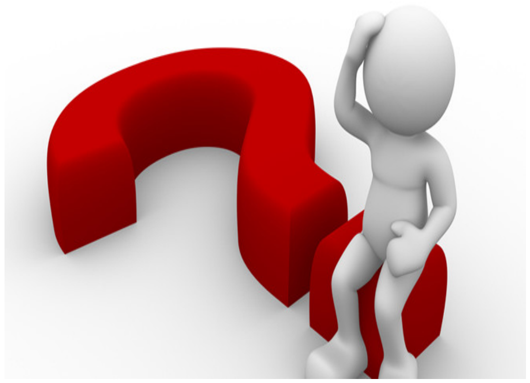
Source code analysis

Fault prediction current state



(Hall et al. 2012)

What about predicting parts of code that are most likely to have bugs in the **future versions**?



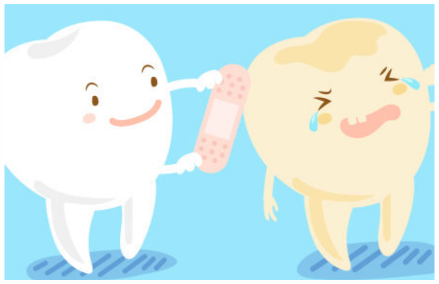
Bug Prediction vs. Bug-proneness Prediction

Analogy of visit to the dentist

Bug in our code - Digital equivalent of a cavity

Bug prediction: An X-ray can help a dentist spot **a hidden cavity**.

Bug-proneness prediction: Dentist discovering weaknesses in the enamel or plaque buildup.



On going projects

Does **understandability** of code has association with bug-proneness?

Can we more accurately identify if a change was for **bug fix or for some other purpose**?

Can we **recommend** a design documentation/email based on the current development context?

Data Mining+ (Deep learning+ Shallow learning)

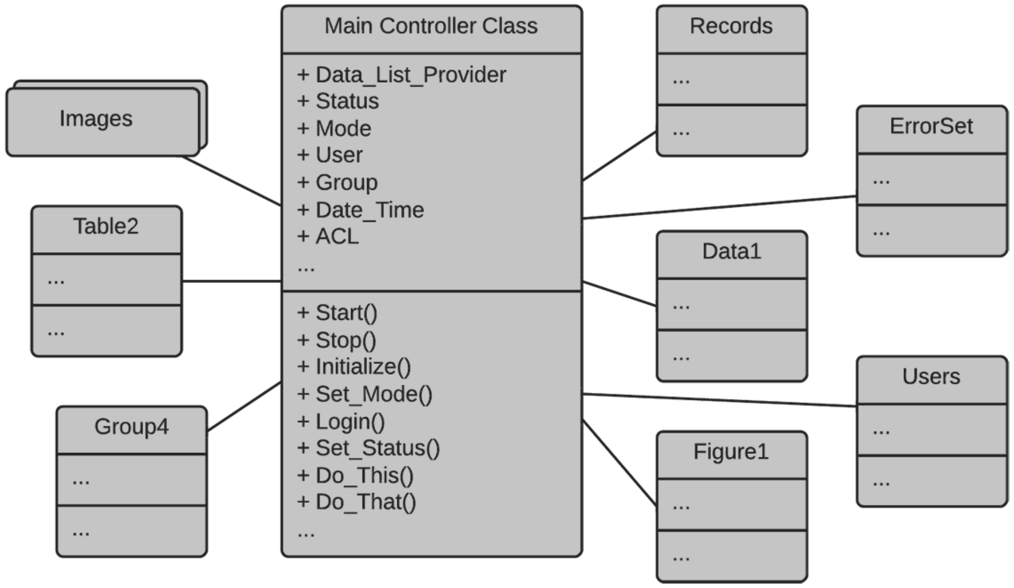
Code smell, a technical factor

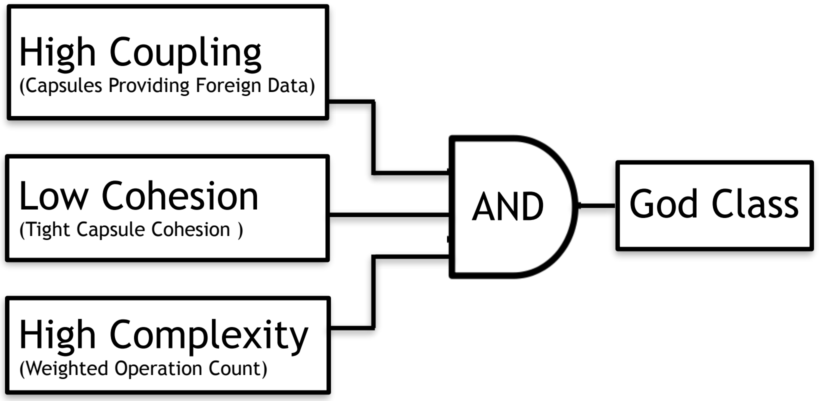
- Developed to identify ***future*** maintainability problems
- Neither syntax errors nor compiler warnings
- ***Symptoms*** of poor design or implementation choices

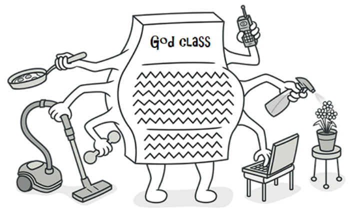
God class

"God class tends to concentrate functionality from several unrelated classes"

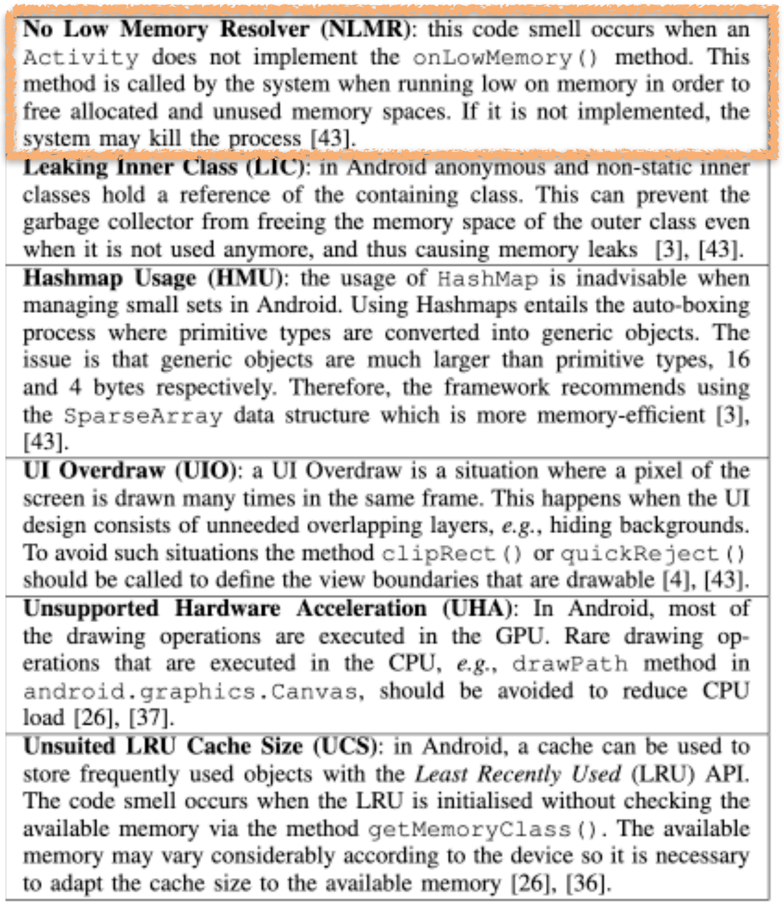
Arise when developers do not fully exploit the advantages of object-oriented design




High Coupling
(Capsules Providing Foreign Data)
Low Cohesion
(Tight Capsule Cohesion)
High Complexity
(Weighted Operation Count)
AND
God Class

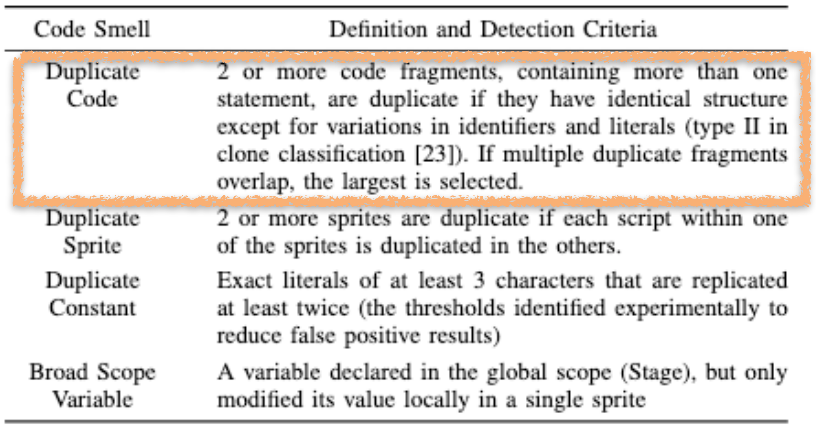


Android specific code smell

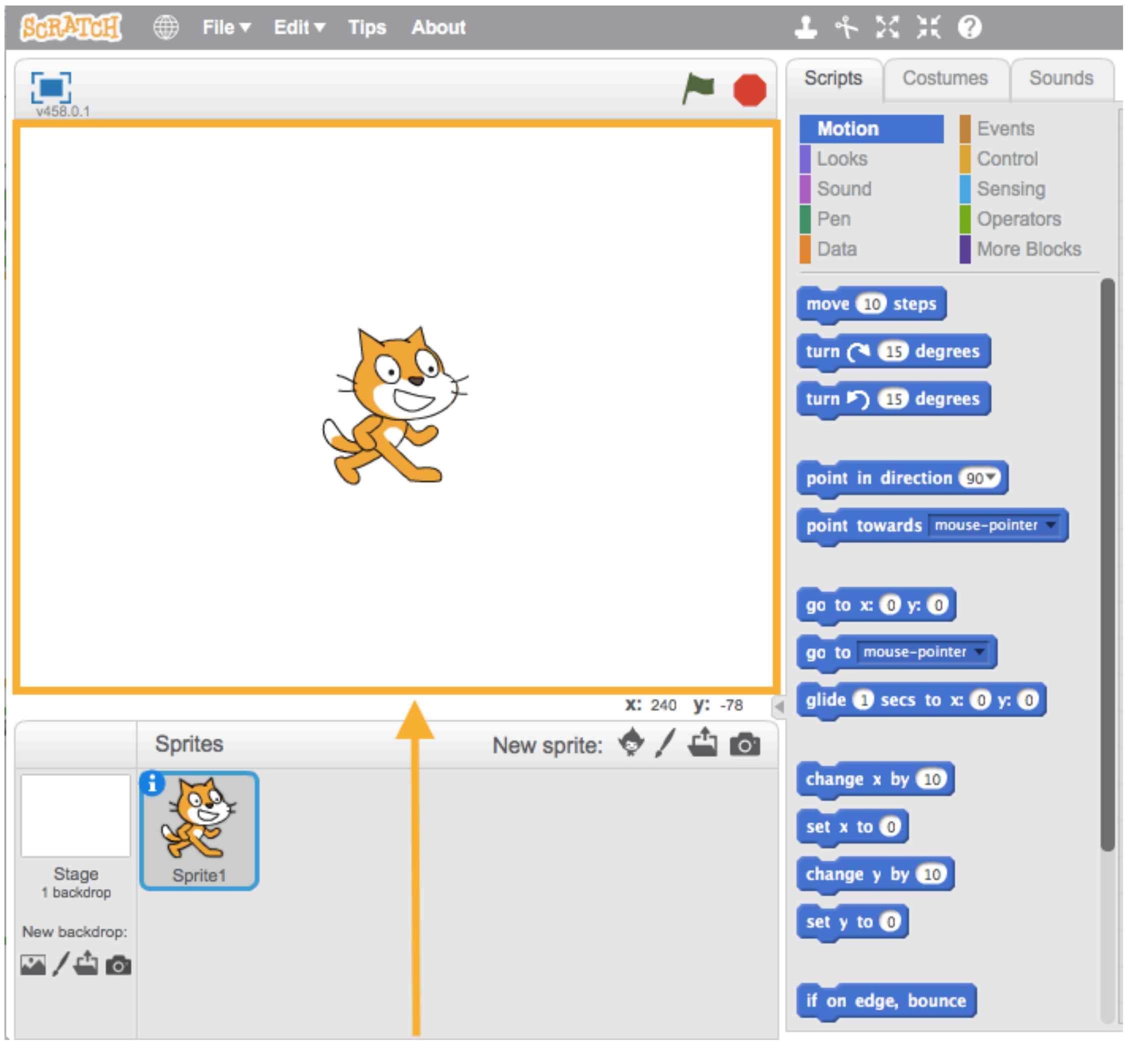


No Low Memory Resolver (NLMR): this code smell occurs when an `Activity` does not implement the `onLowMemory()` method. This method is called by the system when running low on memory in order to free allocated and unused memory spaces. If it is not implemented, the system may kill the process [43].
Leaking Inner Class (LIC): in Android anonymous and non-static inner classes hold a reference of the containing class. This can prevent the garbage collector from freeing the memory space of the outer class even when it is not used anymore, and thus causing memory leaks [3], [43].
Hashmap Usage (HMU): the usage of `HashMap` is inadvisable when managing small sets in Android. Using Hashmaps entails the auto-boxing process where primitive types are converted into generic objects. The issue is that generic objects are much larger than primitive types, 16 and 4 bytes respectively. Therefore, the framework recommends using the `SparseArray` data structure which is more memory-efficient [3], [43].
UI Overdraw (UIO): a UI Overdraw is a situation where a pixel of the screen is drawn many times in the same frame. This happens when the UI design consists of unneeded overlapping layers, *e.g.*, hiding backgrounds. To avoid such situations the method `clipRect()` or `quickReject()` should be called to define the view boundaries that are drawable [4], [43].
Unsupported Hardware Acceleration (UHA): In Android, most of the drawing operations are executed in the GPU. Rare drawing operations that are executed in the CPU, *e.g.*, `drawPath` method in `android.graphics.Canvas`, should be avoided to reduce CPU load [26], [37].
Unsuited LRU Cache Size (UCS): in Android, a cache can be used to store frequently used objects with the *Least Recently Used* (LRU) API. The code smell occurs when the LRU is initialised without checking the available memory via the method `getMemoryClass()`. The available memory may vary considerably according to the device so it is necessary to adapt the cache size to the available memory [26], [36].

Scratch specific code smell



Code Smell	Definition and Detection Criteria
Duplicate Code	2 or more code fragments, containing more than one statement, are duplicate if they have identical structure except for variations in identifiers and literals (type II in clone classification [23]). If multiple duplicate fragments overlap, the largest is selected.
Duplicate Sprite	2 or more sprites are duplicate if each script within one of the sprites is duplicated in the others.
Duplicate Constant	Exact literals of at least 3 characters that are replicated at least twice (the thresholds identified experimentally to reduce false positive results)
Broad Scope Variable	A variable declared in the global scope (Stage), but only modified its value locally in a single sprite



On going projects

Are code smells associated with security vulnerabilities in Android?

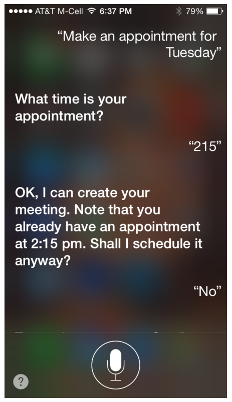
What are the common code smells across all languages? What does that tell us about developers?

Data Mining+ Security Analysis + Shallow learning

Automated Testing and its effectiveness in improving code quality

Conversational Agents (CA)

- Speech recognition/synthesis
- Question answering
  - From the web and from structured information sources.
- Simple agent-like abilities
  - Create/edit calendar entries
  - Reminders
  - Directions
  - Invoking/interacting with other apps



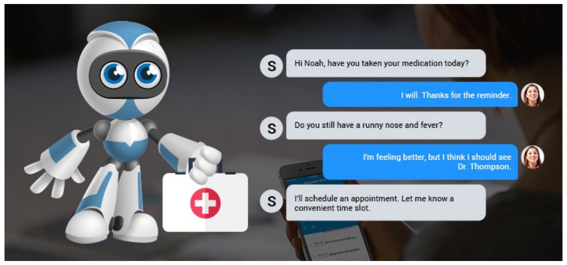
"Make an appointment for Tuesday"

What time is your appointment?

"215"

OK, I can create your meeting. Note that you already have an appointment at 2:15 pm. Shall I schedule it anyway?

"No"


S Hi Noah, have you taken your medication today?

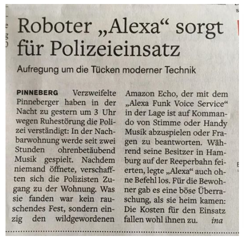
I will. Thanks for the reminder.

S Do you still have a runny nose and fever?

I'm feeling better, but I think I should see Dr. Thompson.

S I'll schedule an appointment. Let me know a convenient time slot.

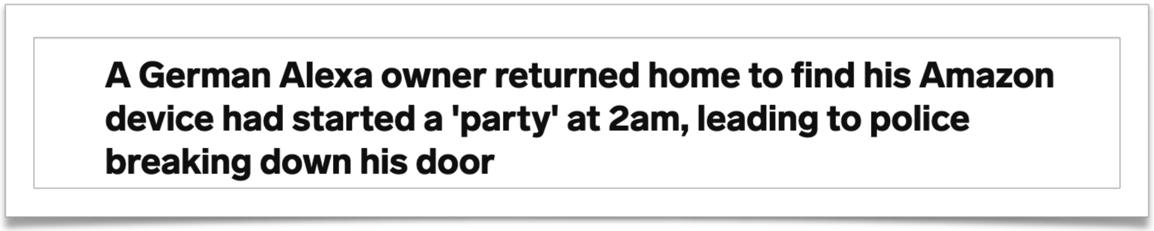
Bugs in Conversational Agents



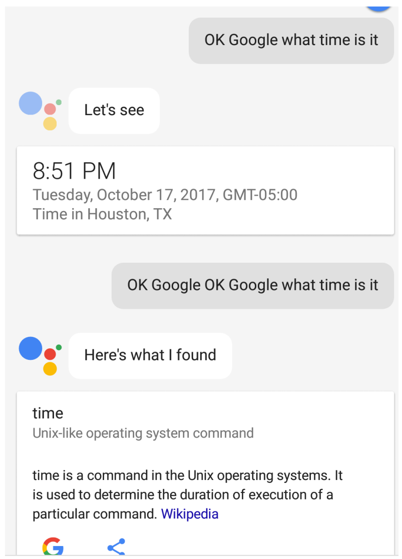
Roboter „Alexa“ sorgt für Polizeieinsatz

Aufregung um die Tücken moderner Technik

PINNEBERG Verzweifelte Pinneberger haben in der Nacht zu gestern um 3 Uhr wegen Ruhestörung die Polizei verständigt: In der Nachbarwohnung werde seit zwei Stunden ohrenbetäubend Musik gespielt. Nachdem niemand öffnete, verschafften sich die Polizisten Zugang zu der Wohnung. Was sie fanden war kein rauschendes Fest, sondern einzig den wildgewordenen Amazon Echo, der mit dem „Alexa Funk Voice Service“ in der Lage ist auf Kommando von Stimme oder Handy Musik abzuspielen oder Fragen zu beantworten. Während seine Besitzer in Hamburg auf der Reeperbahn feierten, legte „Alexa“ auch ohne Befehl los. Für die Bewohner gab es eine böse Überraschung, als sie heim kamen: Die Kosten für den Einsatz fallen wohl ihnen zu. *ina*



A German Alexa owner returned home to find his Amazon device had started a 'party' at 2am, leading to police breaking down his door



OK Google what time is it

Let's see

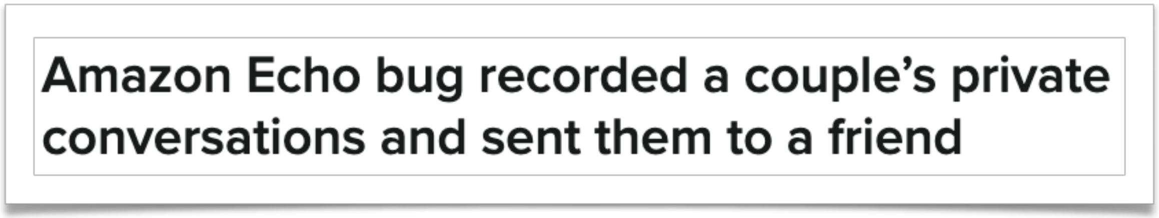
8:51 PM
Tuesday, October 17, 2017, GMT-05:00
Time in Houston, TX

OK Google OK Google what time is it

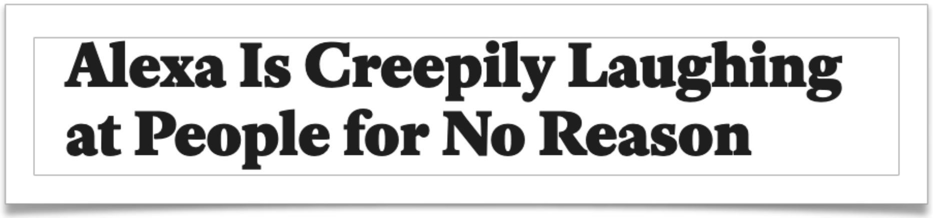
Here's what I found

time
Unix-like operating system command

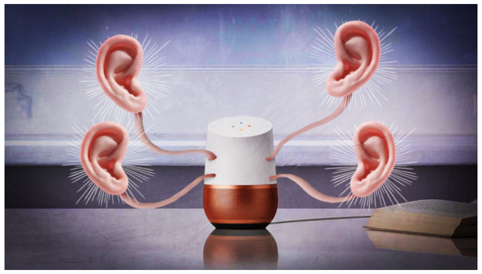
time is a command in the Unix operating systems. It is used to determine the duration of execution of a particular command. Wikipedia



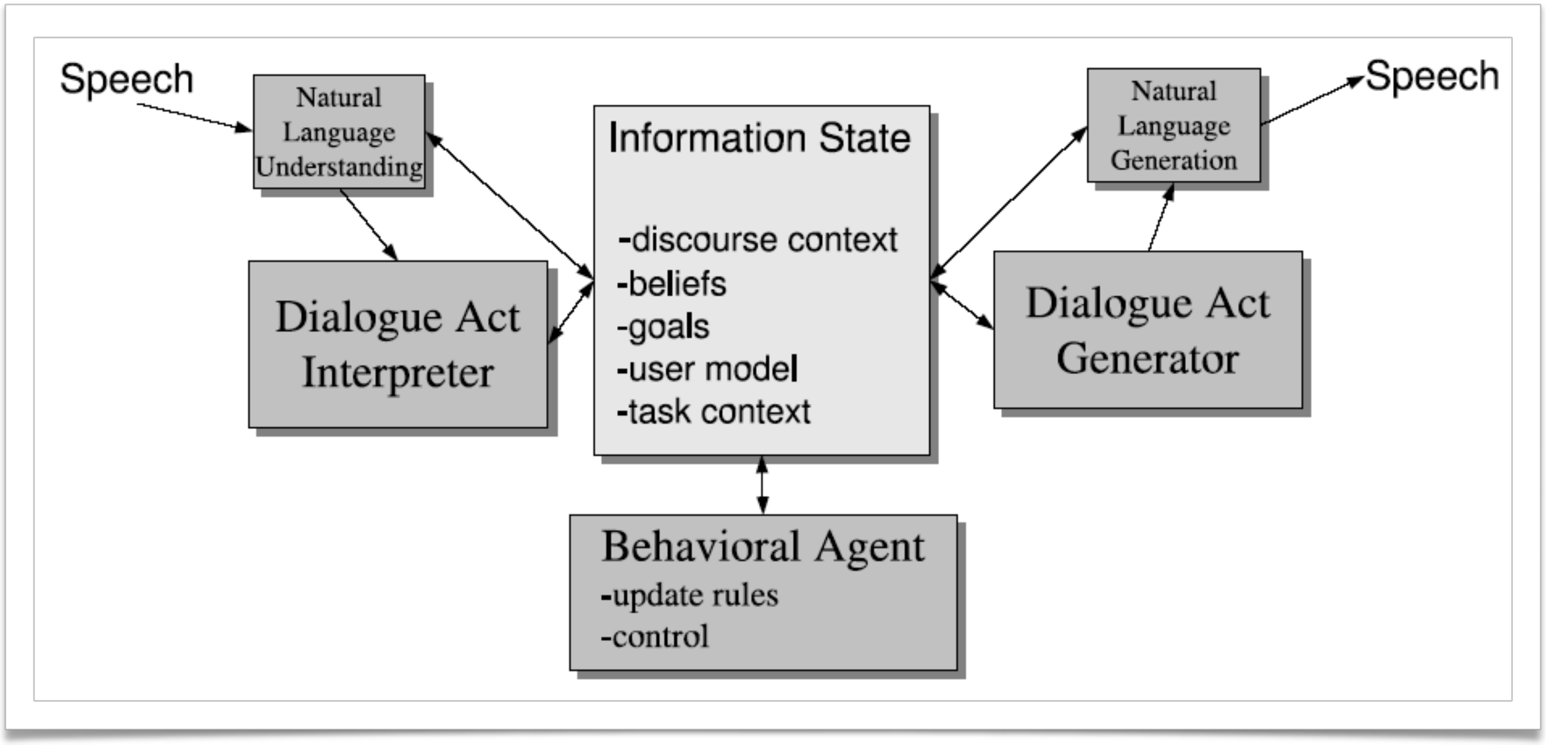
Amazon Echo bug recorded a couple’s private conversations and sent them to a friend



Alexa Is Creepily Laughing at People for No Reason

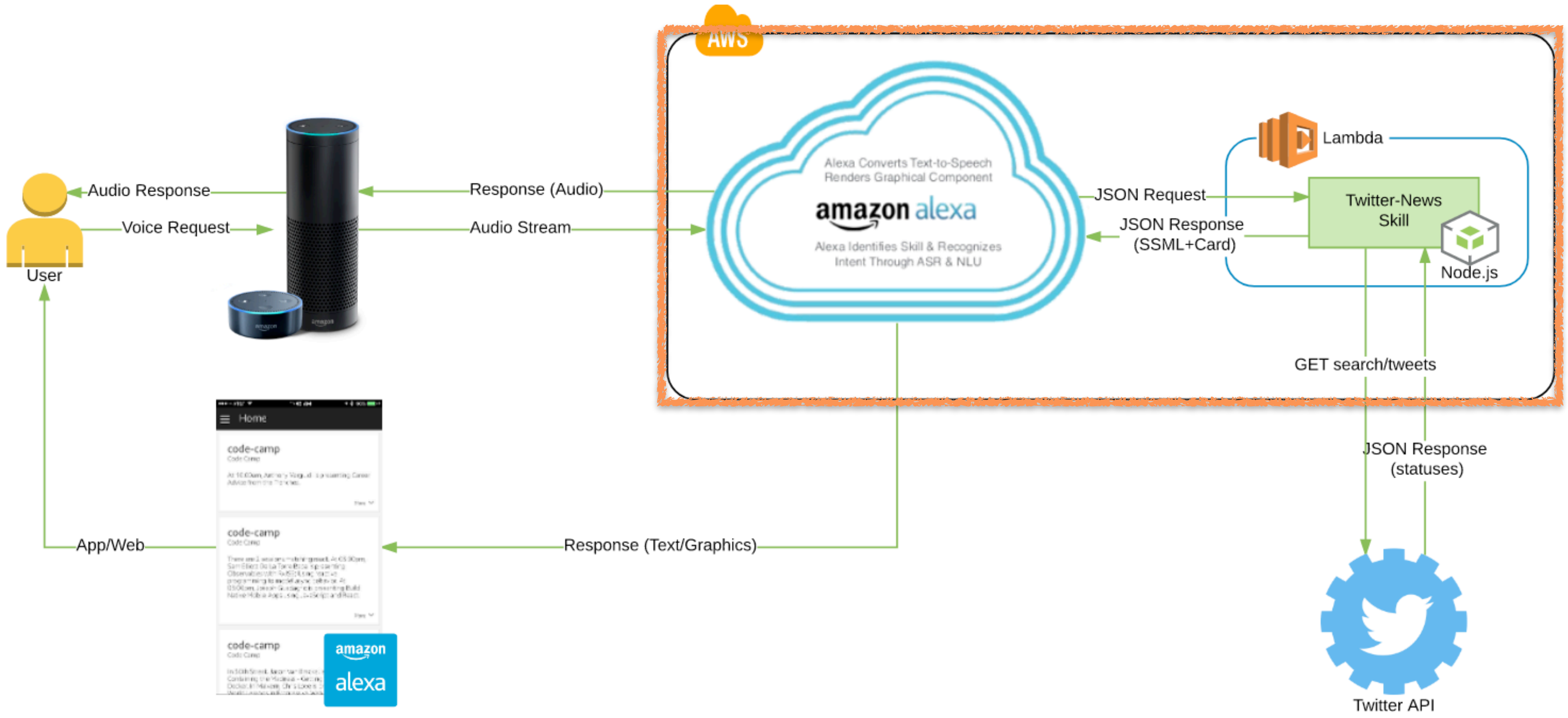


Structure of Conversational Agents



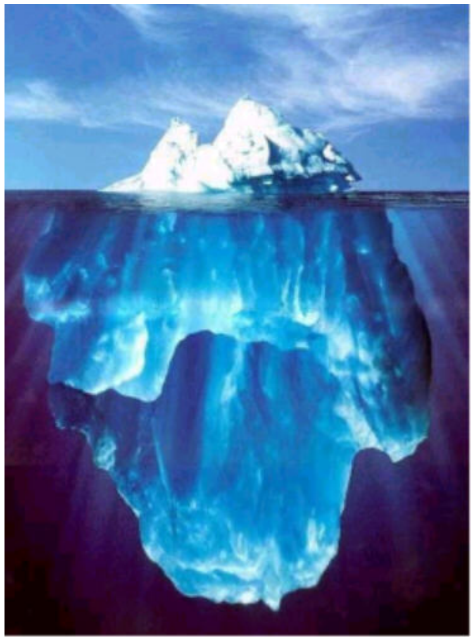
Reference: Jerome Bellegarda

Testing challenges in Conversational Agents



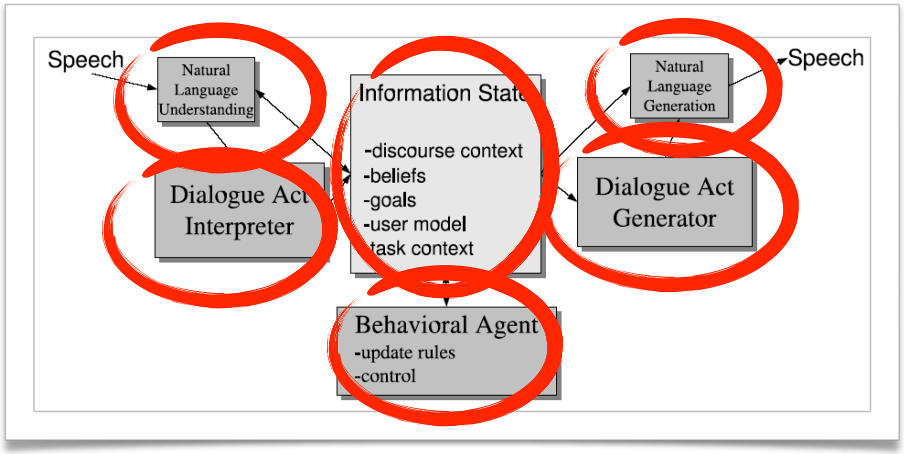
Testing challenges in Conversational Agents

- Difficult to automated testing mechanism due to **Turn-taking**
    - Task/circumstance dependencies
    - Linguistic/cultural differences
    - How do we take and give up turns automatically?
- Large number of possible **Para-Phrasing** of one **utterance**



Testing challenges in Conversational Agents

- Dialogue Act detection is hard, **making test oracle generation even harder**
- **Can you give me a list of the flights from Atlanta to Boston?**
    - This looks like an QUESTION.
    - It has a question-mark, starts with "can you"
    - If so, the answer is: YES.
    - But really it's a COMMAND, a polite form of:
    - Please give me a list of the flights…
- **What looks like a QUESTION can be a COMMAND**
- **What is a good coverage criteria?**



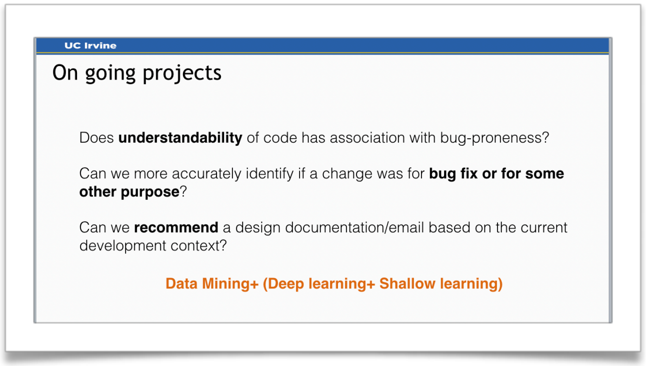
On going projects

What is a good coverage criteria?

Can we automate the testing using a framework?

Data Mining+ (NLP+ Shallow learning)

Conclusion


UC Irvine

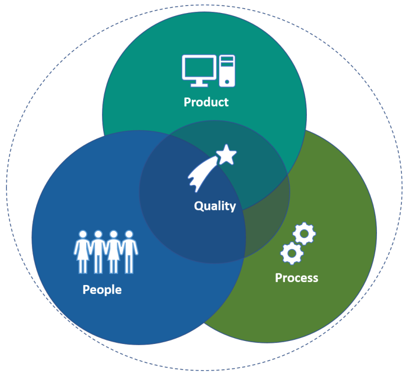
On going projects

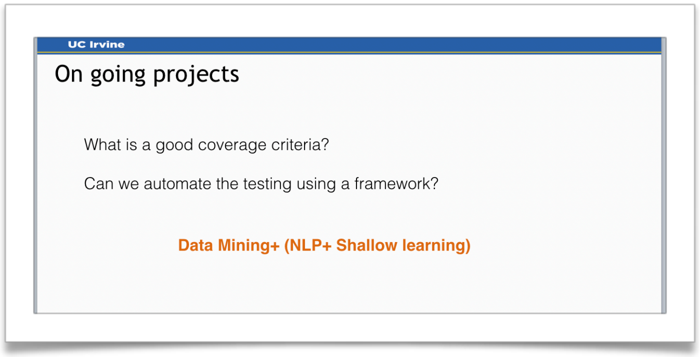
Does **understandability** of code has association with bug-proneness?

Can we more accurately identify if a change was for **bug fix or for some other purpose**?

Can we **recommend** a design documentation/email based on the current development context?

Data Mining+ (Deep learning+ Shallow learning)



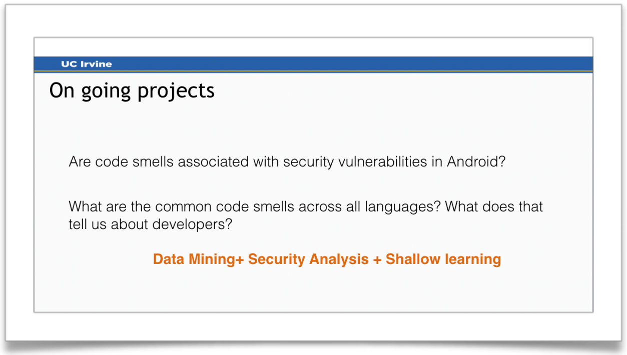

UC Irvine

On going projects

What is a good coverage criteria?

Can we automate the testing using a framework?

Data Mining+ (NLP+ Shallow learning)


UC Irvine

On going projects

Are code smells associated with security vulnerabilities in Android?

What are the common code smells across all languages? What does that tell us about developers?

Data Mining+ Security Analysis + Shallow learning