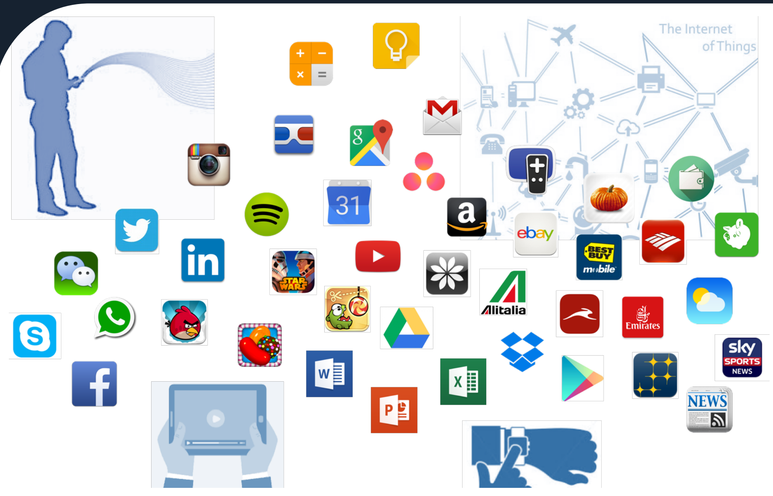
Antonio Ken Iannillo

Tutor: Domenico Cotroneo

XXX Cycle - II year presentation

Dependable Android



Mobile devices (including smartphones, tablets and wearables) assist people in their personal activities, and are today a fundamental resource to communicate and to benefit from cloud services: mail, data storage, e-commerce, banking, and social networking are only few examples. In the near future, they will become digital wallets and keepers of digital identity. Moreover, mobile devices are used in business contexts to access to sensitive enterprise data and services.

As a result, users expect a reliable platform, which should be responsive and avoid smartphone crashes and data losses.

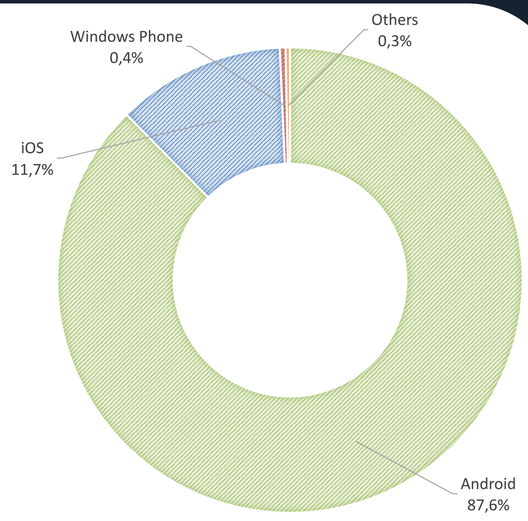
Assuring the reliability of mobile devices is a challenge for smartphone vendors: devices have become significantly complex and feature-rich, are upgraded at a fast pace, and are heavily customized by vendors in order to differentiate their products from competitors.

Android OS is currently dominating the market.

The Android OS grew up to more than 6 millions of lines of Java and C/C++. Moreover, previous studies showed that software complexity and vendor customizations have a negative impact on Android reliability in terms of bug density and vulnerabilities [1][2][3]. This reflects in poor quality perceived by users, and affects the popularity of mobile products on the market.

Thus, the goal of this PhD is to try to answer research questions such as:

- HOW CAN A MANUFACTURER ASSESS THE DEPENDABILITY OF ITS MOBILE DEVICES?
- WHAT KIND OF FAILURES CAN A SMARTPHONE EXPERIENCE? HOW DOES IT REACT?
- HOW CAN ANDROID DEPENDABILITY BE IMPROVED?
- HOW CAN MOBILE VENDORS' DEVICES BE COMPARED W.R.T. DEPENDABILITY?

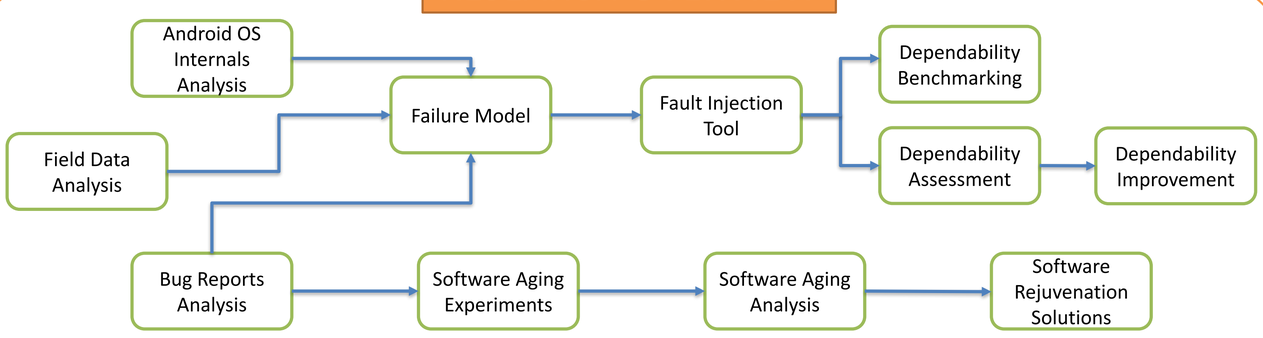


[1] A. K. Maji, K. Hao, S. Sultana, and S. Bagchi, "Characterizing Failures in Mobile OSes: A Case Study with Android and Symbian," in *Software Reliability Engineering (ISSRE), 2010 IEEE 21st International Symposium on*, 2010.

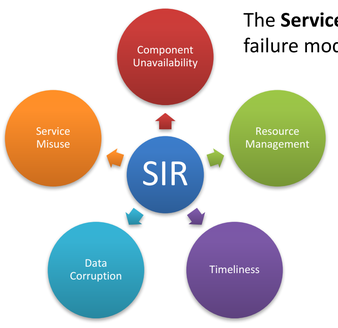
[2] A. K. Maji, F. A. Arshad, S. Bagchi, and J. S. Rellermeyer, "An Empirical Study of the Robustness of Inter-Component Communication in Android," in *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*, 2012.

[3] L. Wu, M. Grace, Y. Zhou, C. Wu, and X. Jiang, "The impact of vendor customizations on Android security," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013.

RESEARCH OVERVIEW



FAILURE MODEL AND FAULT INJECTION TOOL

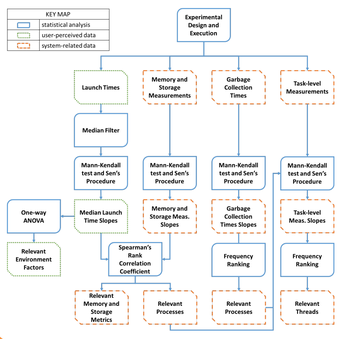


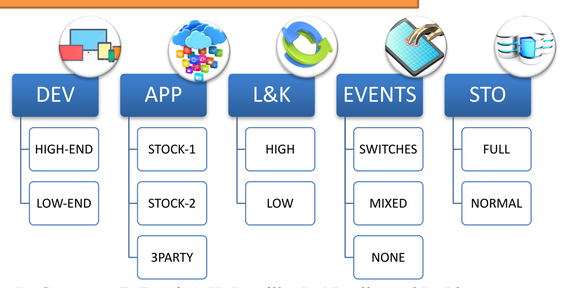
The **Service Interface and Resource** (SIR) method is defined to construct the failure model, based on five failure types.

To inject the failures defined in the failure model, the **fault injection tool** is able to cover the interfaces and resources of Android components. Currently, the tool:

- can inject in **4 subsystems** (phone, camera, sensors, and system server), **16 components**, and **75 fault injection targets** (interfaces and resources);
- supports for Android L and M, while the porting for Android N is under development;
- is about 17k lines of code (11k C++, 3.5k C, 2k Python).

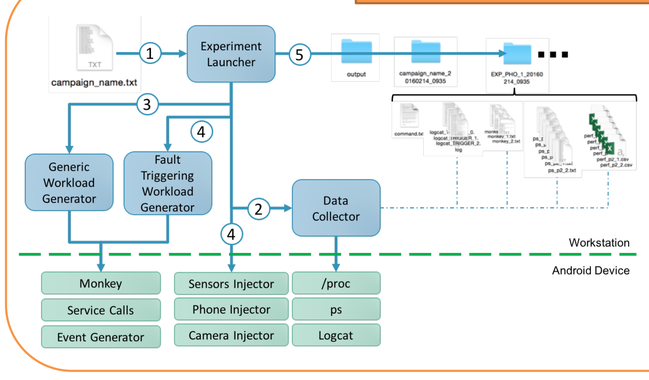
SOFTWARE AGING EXPERIMENTS





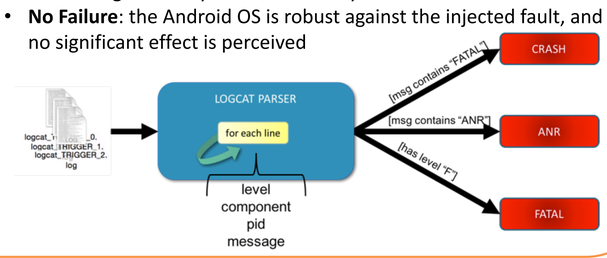
D. Cotroneo, F. Fucci, A.K. Iannillo, R. Natella, and R. Pietrantuono, "Software Aging Analysis of Android OS," in *Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on*, 2016

FAULT INJECTION TESTS



Logs are parsed to retrieve the outcomes, such as:

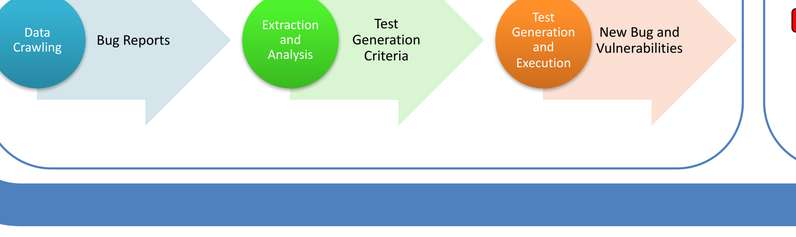
- **Crash**: a native process or a user app has crashed
- **Application Not Responding (ANR)**: a user app is stalled
- **Fatal**: a high-severity error is raised by the Android OS
- **No Failure**: the Android OS is robust against the injected fault, and no significant effect is perceived



SOFTWARE AGING ANALYSIS

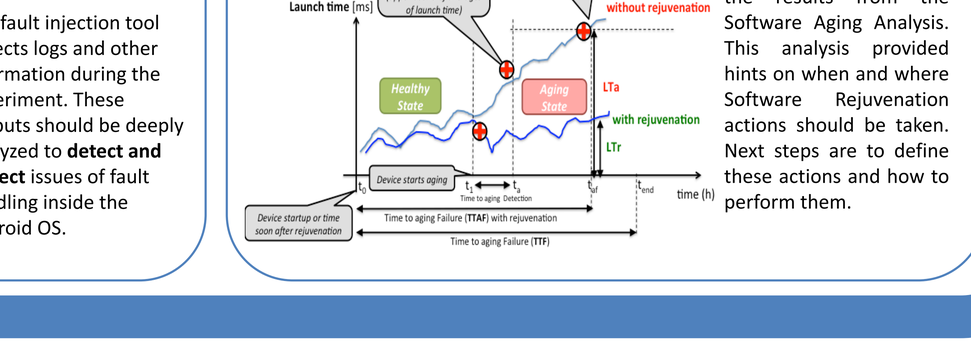
All experiments exhibit a statistically-significant positive trend in the Launch Time series

- Average LT trend: 9.15E-03 ms/s: estimated degradation of 659ms, on average, of the launch time after 20 hours of testing
- Worst LT trend: 6.39E-02 ms/s: estimated degradation of 4.6 seconds after 20 hours



- Android devices suffer from software aging
- Software aging in Android depends on the applied workload
- **System Server**, **System UI**, and **Surface Flinger** proportional page size (PSS) measurements are highly correlated with user-perceived aging
- **System Server** and **System UI** are exposed to performance degradation due to inflation and fragmentation of the heap
- Specific tasks inside the System Server process are more prone to aging
- Android rejuvenation should adopt a **measurement-based approach**

DEPENDABILITY BENCHMARKING



I'm a member of the **Dependable System and Software Engineering Research Team (DESSERT)**, formerly known as MobiLab group, at DIETI - UNINA. I collaborated with a **global manufacturer of Android smartphones** in a research project that aimed to evaluate dependability in the Android OS. I'm currently spending 7 months at Northeastern University, in Boston, supervised by prof. Cristina Nita-Rotaru.

DEpendable Systems & Software Engineering DESSERT Research Team

DIETI UNINA

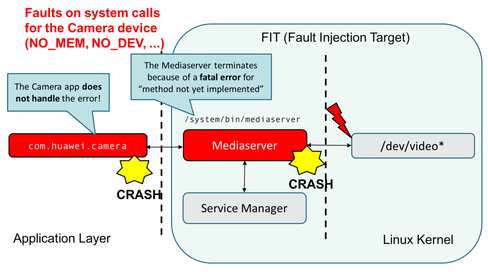
Northeastern University College of Computer and Information Science

BUG REPORTS ANALYSIS

During my stay in Boston, I'm going to work on a project in collaboration with my advisors from both University of Naples and Northeastern University. The main idea is to improve test efficiency by providing generation criteria based on previous bugs.

DEPENDABILITY IMPROVEMENT

Fault injection will be adopted to test the **fault tolerance mechanisms** of the system, in this case Android OS.



The fault injection tool collects logs and other information during the experiment. These outputs should be deeply analyzed to **detect and correct** issues of fault handling inside the Android OS.

SOFTWARE REJUVENATION

Android OS may be "rejuvenated" by using the results from the Software Aging Analysis. This analysis provided hints on when and where Software Rejuvenation actions should be taken. Next steps are to define these actions and how to perform them.