



**HAL**  
open science

## Detection of Anti-patterns in Mobile Applications

Geoffrey Hecht, Laurence Duchien, Naouel Moha, Romain Rouvoy

► **To cite this version:**

Geoffrey Hecht, Laurence Duchien, Naouel Moha, Romain Rouvoy. Detection of Anti-patterns in Mobile Applications. COMPARCH 2014, Jun 2014, Lille, France. hal-01029994

**HAL Id: hal-01029994**

**<https://inria.hal.science/hal-01029994>**

Submitted on 22 Jul 2014

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Detection of Anti-patterns in Mobile Applications

Geoffrey Hecht <geoffrey.hecht@inria.fr>, Laurence Duchien <laurence.duchien@inria.fr>, Naouel Moha <moha.naouel@uqam.ca>, Romain Rouvoy <romain.rouvoy@inria.fr>

## Motivations

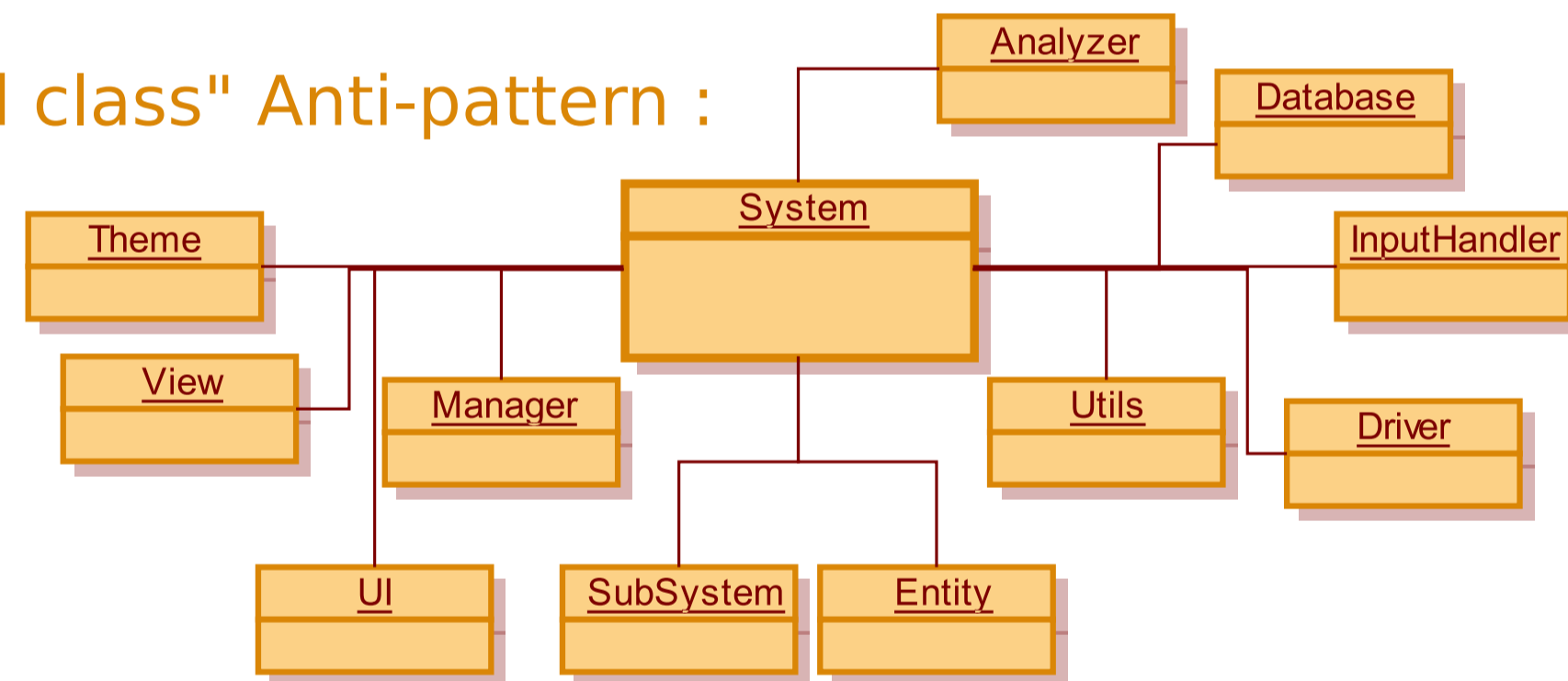
Lack of studies on Android Anti-patterns :

- Influence of the framework on common anti-patterns,
- Android specific anti-patterns,
- Patterns related to applications categories,
- Correlation with application ratings ,
- Evolution along updates ?

## Anti-Patterns

- Bad solutions to solve problems
- Hard to maintain and update
- Source of bugs

"God class" Anti-pattern :



## Metrics

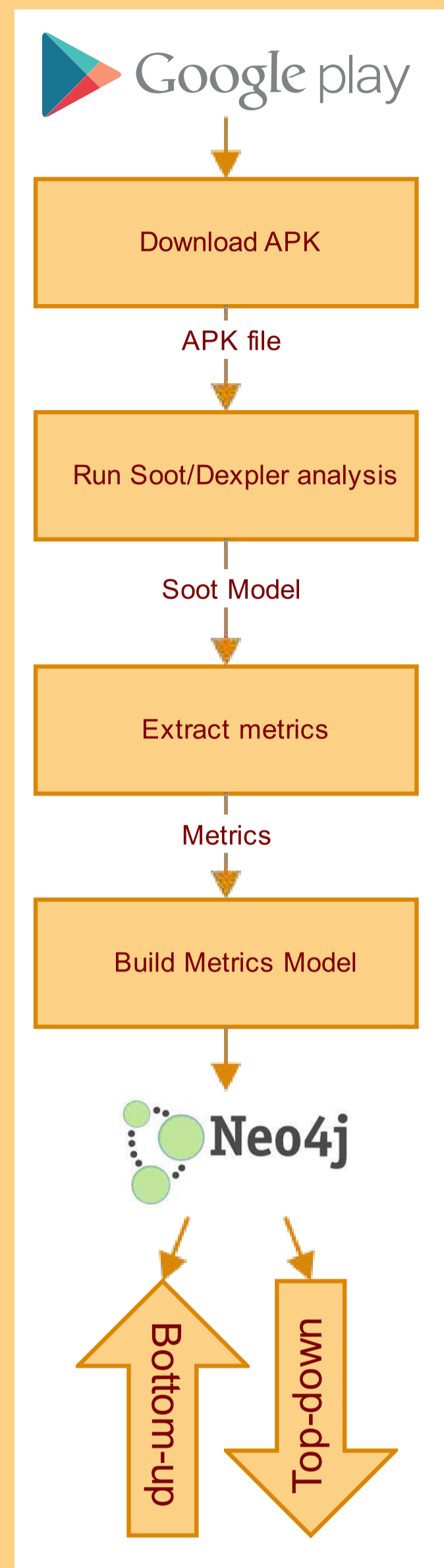
Main metrics :

- Lines of Code
- Cohesion of a class
- Coupling between classes
- Cyclomatic complexity
- Depth of Inheritance

**Anti-patterns = Combination of metrics**

"God class" = low cohesion, numerous lines of code, a high coupling

## Application Analysis

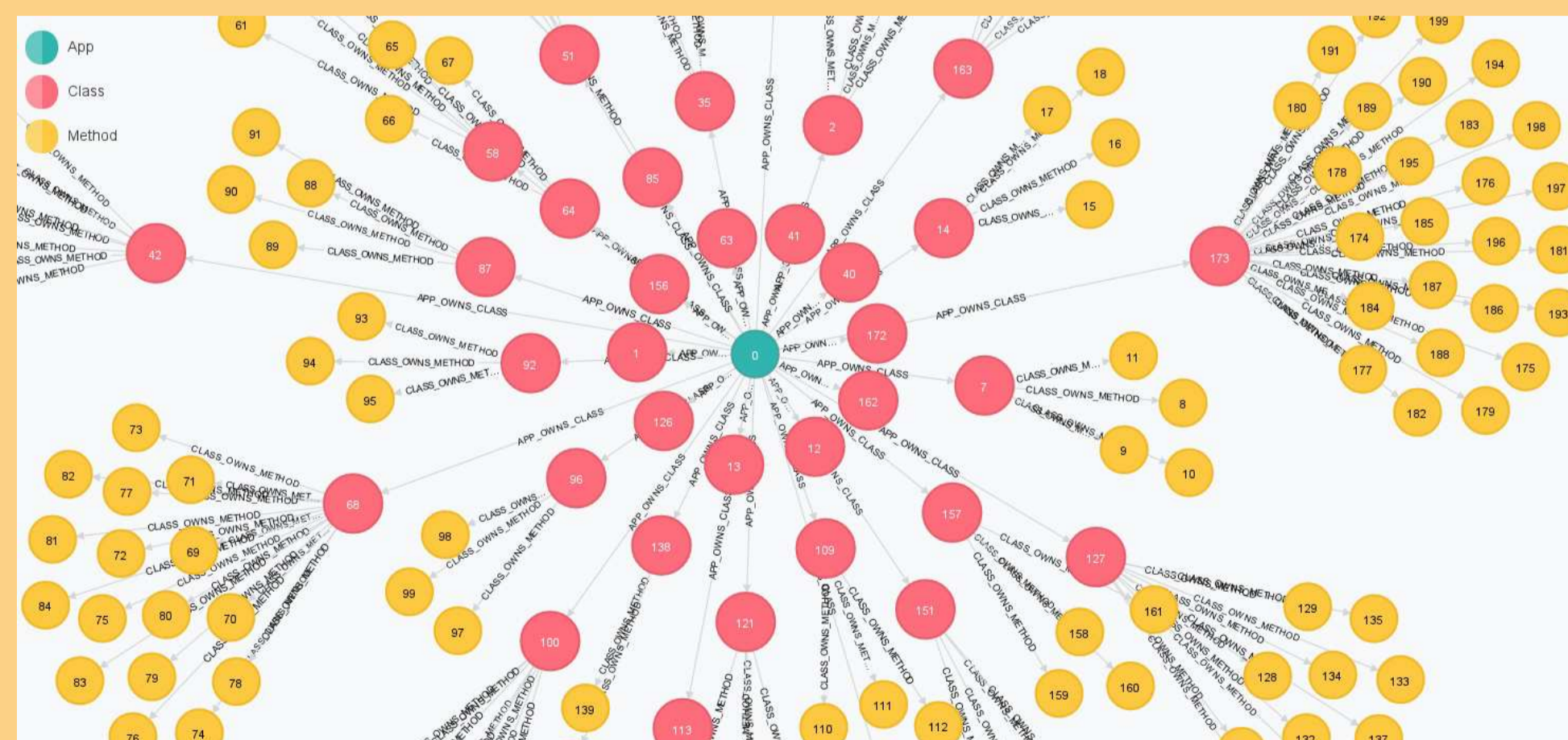


Analysis on **Android package**

Analyze of **binary code** with Soot and dexpler submodule

Metrics are extracted from Soot model to build a new one

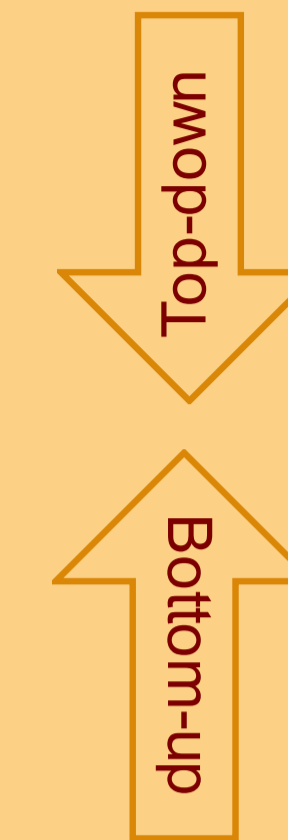
This model is then stored in a **graph database**



Nodes = app, method or class  
Edges = relation  
Properties = metrics

## Anti-Patterns Detection

Study on **3500+ apps** with an average of 7 versions



Top-down approach : Common anti-patterns with **thresholds** on metrics

Bottom-up approach : **Machine-learning** algorithms to detect new anti-patterns

## Anti-Patterns Fixing

- Increase maintainability and understandability
- Reduce energy consumption
- Improve responsiveness
- Decrease package size
- Less bugs

## References

- Bartel, Alexandre, et al. "Dexpler: converting android dalvik bytecode to jimple for static analysis with soot." Proceedings of the ACM SIGPLAN International Workshop on State of the Art in Java Program analysis. ACM, 2012.
- Linares-Vásquez, Mario, et al. "Domain matters: bringing further evidence of the relationships among anti-patterns, application domains, and quality-related metrics in Java mobile apps." Proceedings of the 22nd International Conference on Program Comprehension. ACM, 2014.
- Minelli, Roberto, and Michele Lanza. "Software Analytics for Mobile Applications--Insights and Lessons Learned." Software Maintenance and Reengineering (CSMR), 2013 17th European Conference on. IEEE, 2013.
- Shabtai, Asaf, Yuval Fledel, and Yuval Elovici. "Automated static code analysis for classifying Android applications using machine learning." Computational Intelligence and Security (CIS), 2010 International Conference on. IEEE, 2010.