UNIVERSITATEA BABEŞ BOLYAI, CLUJ NAPOCA, ROMÂNIA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

# Metode inteligente pentru inferența arhitecturilor software din codurile sursă ale aplicațiilor mobile

**Student doctorand: Dragoş Dobrean**
**Îndrumător: Prof. Dr. Laura Dioşan**

2021

## Abstract

Cu peste două treimi din populația lumii care le utilizează, telefoanele inteligente (smartphone-urile) au devenit cele mai personale dispozitive pe care le folosim. Smartphone-urile sunt folosite pentru comunicarea cu ceilalți, divertisment, analiza stării noastre de sănătate, comandarea mâncării, pentru a practica corect exerciții fizice, găsirea unei perechi și multe altele. Încet, au început chiar să înlocuiască portofelele, deoarece ne permit să efectuăm plăți, să stocăm bilete digitale și să validăm identitatea proprietarului. Odată cu creșterea popularității acestor dispozitive și a progreselor hardware, aplicațiile mobile au devenit unele dintre cele mai frecvent dezvoltate sisteme software.

Arhitectura software a unui aplicații mobile, în special în cazul celor complexe, care sunt întreținute și dezvoltate pentru perioade lungi de timp (ani), joacă un rol important în succesul lor. Prin implementarea corectă a unui anumit model arhitectural într-o aplicație mobilă, dezvoltatorii asigură testabilitatea, extensibilitatea și flexibilitatea acelui proiect, iar asta reprezintă un avantaj major într-o piață cu cerere ridicată și ritm alert. Deoarece corectitudinea arhitecturală a unui proiect mobile reprezintă un factor important în succesul său, această teză studiază diferite abordări în care problemele arhitecturale ar putea fi identificate și rezolvate înainte ca acestea să poată afecta produsul. Scopul acestei teze este de a deschide calea pentru construirea unui sistem capabil să identifice problemele arhitecturale în timpul fazei de dezvoltare sau în timpul rulării sistemelor de livrare integrată / continuă (Continuous Integration - CI / Continuous Delivery - CD) și de a creea o bază solidă de cunoștințe care ar putea să fie aplicate și altor platforme.

Fiecare arhitectură software are un set de constrângeri între componentele sale. Aplicațiile de prezentare (cum ar fi aplicațiile mobile) utilizează predominant arhitecturi software stratificate. În timp ce regulile arhitecturale dintre straturile unui sistem pot fi ușor definite, o problemă mai importantă este inferența straturilor arhitecturale din codul unei aplicații mobile. Pentru a crea un instrument care evidențiază automat problemele arhitecturale, avem nevoie de un sistem care să poată deduce componentele arhitecturale din cod și să le clasifice în straturi fără a fi nevoie de interacțiunea umană.

Abordările prezentate în această teză se folosesc de Kiturile de Dezvoltare Software (Software Development Kit - SDK) utilizate în construirea acestor tipuri de aplicații. Prin utilizarea informațiilor din aceste SDK-uri pot fi extrase date privind tipurile de componente din codul aplicației și stratul arhitectural în care ar trebui să se afle. Pe lângă informațiile din SDK-uri, tehnicile de învățare automată (Machine Learning - ML) sunt, de asemenea, implicate în unele dintre abordările propuse pentru a ajuta procesul de clasificare a componentelor unei aplicații mobile în straturi arhitecturale. Mai mult, atunci când am proiectat abordările, ne-am concentrat și asupra portabilității, toate abordările prezentate pot fi ușor extinse la alte platforme mobile sau non-mobile.

Cercetările prezentate în această lucrare se află la confluența a două domenii, Ingineria Software și Inteligența Artificială. Scopul acestui studiu este de a avansa domeniul ingineriei software prin îmbunătățirea procesului de dezvoltare a produselor software. Noile descoperiri sunt de obicei produsul secundar al interacțiunii dintre diferite domenii de studiu și credem cu tărie că metodele de inteligență artificială pot fi aplicate cu succes în ingineria software și ambele domenii pot fi îmbunătățite de această simbioză. Problema inferenței arhitecturilor software din codul unei aplicații mobile este puternic legată de domeniul ingineriei software, cu toate acestea, în timp ce dezvoltam abordări pentru îmbunătățirea acestui proces, am încercat să abordăm fiecare problemă pe care am întâmpinat-o din perspectiva inteligenței artificiale.

În această lucrare, am analizat cele mai comune modele arhitecturale utilizate pe platformele mobile, împreună cu avantajele și dezavantajele acestora. Am descoperit că precursorul unui număr mare dintre aceste modele arhitecturale este Model View Controller (MVC), in prezent

fiind încă unul dintre cele mai utilizate modele arhitecturale folosite in aplicațiile de prezentare, în special în cazul aplicațiilor mobile. Accentul în această lucrare inițială a fost studierea și găsirea modalităților de detectare automată a compoziției straturilor arhitecturale ale unei arhitecturi MVC dintr-o aplicație mobilă.

Prima noastră abordare – Mobile software architecture checker system (*mACS*), cea deterministă, a obținut rezultate bune în rezolvarea acestei probleme, utilizând informațiile disponibile în SDK-urile mobile și un set de euristici. Întrucât există alte modele arhitecturale mai complexe utilizate în dezvoltarea aplicațiilor mobile, ne-am mutat atenția asupra acestora și încercăm să venim cu o abordare care să ne permită să detectăm alte tipuri de straturi arhitecturale, nu numai cele prezente în MVC. A doua propunere – Clustering architectural layers (*CARL*) este abordarea nedeterministă, în care tehnicile de învățare automată sunt utilizate pentru împărțirea componentelor unei aplicații mobile în straturi arhitecturale. Această abordare nu a fost la fel de performantă ca prima (abordarea deterministă), dar a avut flexibilitatea suplimentară care a lipsit în cazul lui *mACS*. A treia abordare – Hybrid detection of architectural layers (*HyDe*) este una hibridă, în care am combinat abordarea deterministă cu cea nedeterministă. Această a treia încercare a avut o precizie îmbunătățită față de metoda nedeterministă și o flexibilitate mult mai bună decât cea deterministă și reprezintă un bun candidat pentru o soluție care ar putea analiza modele arhitecturale complexe.

Pe scurt, această teză a reușit să contribuie la domeniul arhitecturii software în mai multe arii de cercetare. Această lucrare investighează importanței arhitecturilor software într-un cadru academic (studenți și instructori), precum și în cazul programatorilor. În aceasta lucrare am analizat și a comparat cele mai utilizate modele arhitecturale pe platformele mobile și am studiat cele mai frecvente greșeli ale celui mai prolific model arhitectural – MVC. Mai mult, cu această lucrare, am formulat matematic problema identificării automate a straturilor arhitecturale din codul aplicatiilor mobile și am propus trei abordări (*mACS*, *CARL*, *HyDe*) pentru rezolvarea acesteia. În plus, pentru testarea abordărilor propuse și compararea acestora, a fost construit un set de aplicații (iOS și Android), iar cele trei abordări au fost comparate dintr-un punct de vedere intrinsec și extrinsec și validate statistic cu metrici specifice abordării și empiric folosind metoda interviului. În plus, am comparat metodele noastre de detectare cu cele deja existente din literatură, analizând modul în care sunt utilizate informațiile, modul în care informația este analizată, granularitatea și flexibilitatea lor. Metodele noastre au dovedit o flexibilitate sporită, ceea ce înseamnă că pot fi aplicate cu ușurință pe alte platforme și folosind alte limbaje de programare. În cele din urmă, am arătat prin experimente că metodele prezentate în această teză pot fi portate cu ușurință pe alte platforme mobile, sau chiar pe altele platforme non-mobile, cu rezultate excelente. Cu această lucrare, am deschis o mulțime de direcții de cercetare pentru viitor. Rezultatele prezentate în această teză ne fac să ne simțim încrezători în realizarea obiectivului nostru - crearea unui instrument de verificare a arhitecturii software mobile, care ar putea fi integrat în sisteme de CI / CD sau utilizat local de către dezvoltatori sau studenți. Prioritățile de cercetare pe termen scurt se învârt în jurul îmbunătățirii performanței abordărilor noastre. Deşi rezultatele obținute sunt promițătoare, mai sunt multe de făcut până când să avem o soluție automată care poate detecta straturile arhitecturale din codul aplicațiilor mobile. Planurile noastre pe termen lung se învârt în jurul instrumentului analiză a arhitecturilor software, aplicarea lui pe mai multe platforme și creerea de instrumente pentru îmbunătățirea cunoștințelor arhitecturale ale studenților.

**Cuvinte cheie**: arhitectură software pentru aplicații mobile; analiză statică automată; Model View Controller; analiza automată a arhitecturilor software; informații structurale și lexicale; clusterizare software; abordare hibridă; detectarea arhitecturii software; multi-platformă

# Cuprins