Recovering Trace Links Between Software Documentation And Code

Aciu Mălina
Găină Aurelian-Octavian
Ivan Bogdan
Mocanu Radu

FMI UNIBUC

Februarie 2026

Articol: *Recovering Trace Links Between Software Documentation And Code* (Keim et al.), ICSE 2024, Lisbon.
DOI: 10.1145/3597503.3639130

Cuprins

- Motivație
- Context și problemă
- Ideea principală: legături tranzitive prin artefacte intermediare
- Metoda: ArDoCo + ArCoTL $\Rightarrow$ TransArC
- Evaluare și rezultate
- Puncte forte / limite
- Contribuția noastră

Motivație: de ce e o problemă reală

- În dezvoltare apar multe artefacte: documentație, modele, cod.
- Relațiile dintre ele rămân adesea implicite ⇒ greu de menținut consistența.
- Trasabilitatea ajută în mentenanță, analiză de impact, localizare bug-uri, securitate, conformitate.
- Manual: consumă timp și produce erori ⇒ apare nevoia de *Traceability Link Recovery (TLR)*.

Motivație: de ce SAD $\leftrightarrow$ cod e mai greu

- *Software Architecture Documentation (SAD)* păstrează decizii și raționamente despre arhitectură.
- Documentația e la un nivel de abstractizare diferit față de cod $\Rightarrow$ *semantic gap*.
- Legături SAD $\leftrightarrow$ cod ar face mai ușoară găsirea rapidă a informației relevante pentru schimbări.
- Paper-ul propune folosirea *Software Architecture Model (SAM)* ca artefact intermediar.

Articolul: context și contribuții

- Ideea centrală: nu legăm direct SAD de cod, ci tranzitiv prin SAM.
- Contribuții declarate:
  - ArCoTL: legături SAM $\leftrightarrow$ cod,
  - TransArC: legături SAD $\leftrightarrow$ cod, compuse tranzitiv,
  - replication package (cod, baselines, date, rezultate).

Problema: ce vrem să recuperăm

- Țintă: legături între propoziții din SAD și elemente din cod (la nivel de fișier/structură).
- De ce e dificil:
  - vocabular diferit (ex. „persistence” vs „dataaccess”),
  - referințe indirecte în text (ex. „it”),
  - o componentă poate corespunde unui întreg pachet, nu unei singure clase.

Ideea-cheie: legături tranzitive, nu directe

- Direct SAD $\leftrightarrow$ cod: *semantic gap* mare $\Rightarrow$ risc ridicat de imprecizie.
- Propunere:
  - SAD $\leftrightarrow$ SAM (mai ușor: concepte arhitecturale similare) = ArDoCo
  - SAM $\leftrightarrow$ cod (mai ușor: structură/pachete/elemente) = ArCoTL
  - apoi compunem rezultatul $\Rightarrow$ SAD $\leftrightarrow$ cod = TransArC

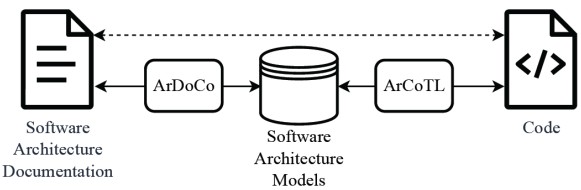


Figure 2: High-level view of the TransArC approach for linking SADs and code using ArDoCo [25] and our novel ArCoTL.

Intuiție: exemplul din paper

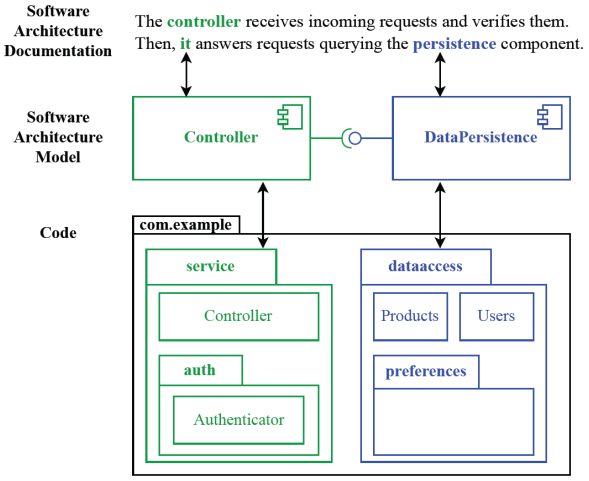


Figure 1: Running example: System with software architecture documentation, software architecture model, and code[1].

Întrebări de cercetare (RQ)

- RQ1: Cât de bine recuperează ArCoTL legături SAM $\leftrightarrow$ cod?
- RQ2: Cât de bine recuperează TransArC legături SAD $\leftrightarrow$ cod, tranzitiv prin SAM?
- RQ3: Cum se compară cu abordări *requirements-to-code* folosite ca baseline?

ArDoCo (SAD ↔ SAM): pe scurt

- Pipeline cu euristici, orientat pe consistență între documentație și model.
- Procesează textul (NLP: tagging, lematizare, dependency parsing).
- Extrage mențiuni (nume/tipuri), le grupează, apoi le conectează cu elementele din SAM pe bază de similaritate.

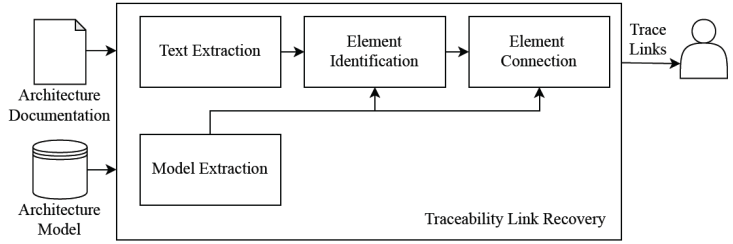


Figure 3: Overview of the ArDoCo approach [25]

ArCoTL (SAM $\leftrightarrow$ cod): ideea

- Două etape:
  1. Transformă SAM și cod în reprezentări intermediare (mai abstracte).
  2. Aplică euristici + agregare într-un *computational graph* pentru a obține potriviri stabile.
- Scop: să fie mai puțin dependent de limbajul de programare / limbajul de modelare.

Reprezentări intermediare: de ce ajută

- Pentru arhitectură: componente/interfețe/semnături + relații (provided/required, subcomponente).
- Pentru cod: un model inspirat din KDM (pachete, module, unități, clase/interfețe).
- Beneficiu: euristicile pot fi scrise la nivel conceptual (structură + nume), nu la nivel de sintaxă.

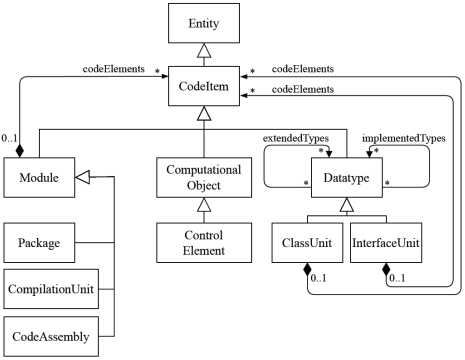


Figure 5: Intermediate model for code

Euristici + agregare (cum decide efectiv)

- Euristici (ex.): potrivire după nume, pachete, cale/structură.
- Euristici dependente: ajustează scoruri și elimină potriviri improbabile.
- Agregare: alege „cele mai bune” potriviri în ambele direcții (componentă $\leftrightarrow$ element cod).
- *Computational graph* controlează ordinea, filtrele și combinarea rezultatelor.

ArCoTL: graful computațional

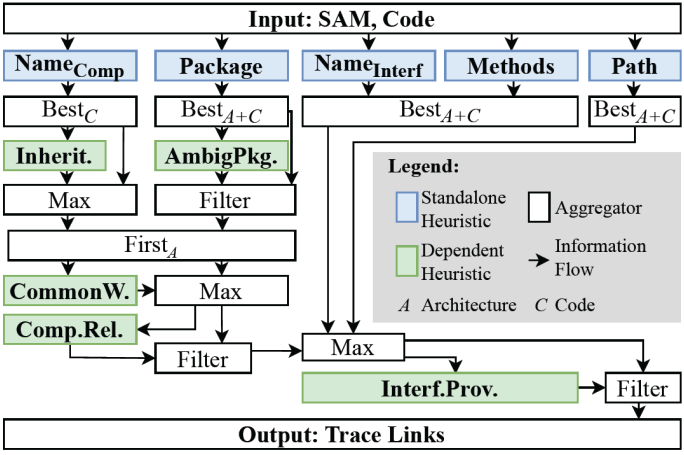


Figure 6: The computational graph of ArCoTL

TransArC: legături SAD $\leftrightarrow$ cod, tranzitiv

- Avem două mulțimi de legături:
  - propoziție SAD $\leftrightarrow$ element SAM (ArDoCo)
  - element SAM $\leftrightarrow$ element cod (ArCoTL)
- Le compunem prin elementul comun din SAM $\Rightarrow$ propoziție SAD $\leftrightarrow$ element cod.
- Autorii prioritizează precizia: evită completări „forțate” dacă SAD menționează entități de cod care nu apar în SAM.

Evaluare: dataset, metrici, rezultate

- 5 proiecte open-source: MediaStore, TeaStore, TEAMMATES, BigBlueButton, JabRef.
- Gold standards pentru SAM $\leftrightarrow$ cod și SAD $\leftrightarrow$ cod.
- Rezultate raportate:
  - ArCoTL (SAM $\leftrightarrow$ cod): F1 ≈ 0.98
  - TransArC (SAD $\leftrightarrow$ cod): F1 ≈ 0.82, peste baselines

Puncte forte / limite (din perspectiva noastră)

Puncte forte

- Reduce *semantic gap* printr-un artefact intermediar (SAM) și combină două soluții specializate.
- Obține rezultate foarte bune pe mai multe proiecte, cu comparație clară față de baseline-uri.
- Reprezentările intermediare fac abordarea mai ușor de adaptat la alte limbaje.

Limite

- Depinde de existența SAM (sau de un proces de obținere a lui).
- Se focalizează pe modele structurale component-based; generalizarea la alte modele poate schimba mult performanța.
- Cazurile în care SAD menționează entități de implementare „neprinse” în SAM rămân dificil de tratat fără a pierde precizie.

Reutilizarea TransArC

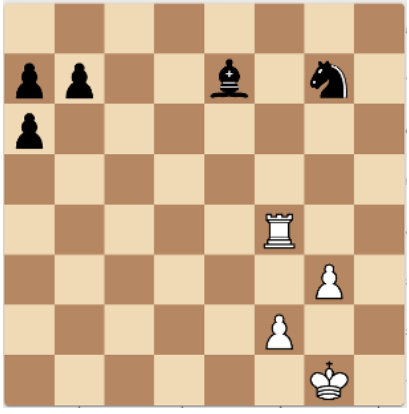
Ce este FEN?

Chess Position Validator

- Verifică dacă un FEN de intrare descrie o poziție corectă

Exemple:

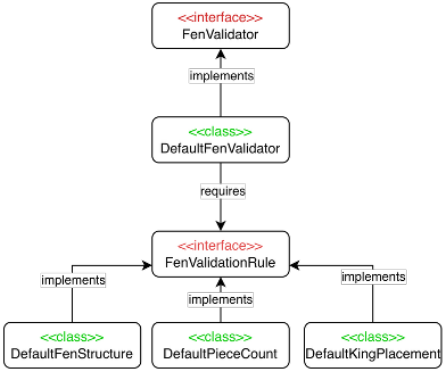
- `8/pp2b1n1/p7/8/5R2/6P1/5P2/6K1` Output: poziție incorectă



Chess Position Validator

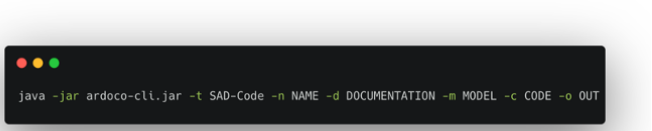
- `2b3k1/6bp/2n1p1p1/2p5/2N1N3/P2r1P2/5BPP/5RK1` Output: poziție corectă

Arhitectura sistemului Chess Validator



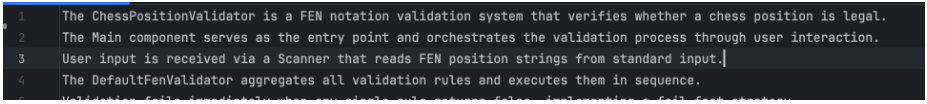
Reutilizarea utilitarului TransArC

- Utilitarul a fost rulat folosind imaginea Docker `ghcr.io/ardoco/icse24`
- Pentru reutilizarea TransArC, am folosit CLI-ul pus la dispoziție
- Am specificat:
  - task-ul dorit
  - fișierele de intrare necesare



Fișierul SAD

- Conține descrierea sistemului în limbaj natural
- O propoziție pe fiecare rând

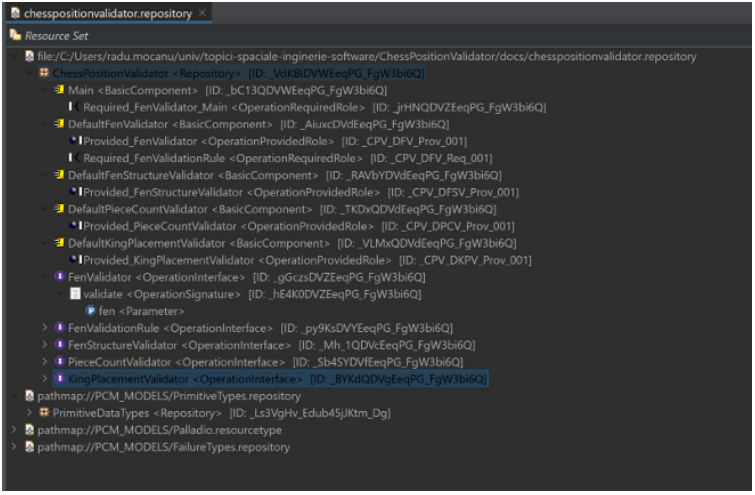


Fișierul SAM

- Conține interfețe și componente
- Descrie arhitectura sistemului

```
<interfaces__Repository
    xsi:type="repository:OperationInterface"
    id="_FenValidator"
    entityName="FenValidator">
    <signatures__OperationInterface
        id="_FenValidator_validate"
        entityName="validate"
        returnType__OperationSignature="BOOLEAN"/>
</interfaces__Repository>
```

Palladio Bench



Demo

Comenzi pentru rularea replication package

```
docker run -it -rm -v ~/Documents/project/ChessPositionValidator:/workspace/project
ghcr.io/ardoco/icse24 /bin/bash

java -jar ardoco-cli.jar -t SAD-Code -n ChessValidator -d /workspace/project/src/SAD_highlevel.txt -m
/workspace/project/src/SAM_chessValidator.repository -c /workspace/project/src/main/java -o
/workspace/project/transarc_output

java -jar ../evaluator/evaluator.jar -t
/workspace/project/transarc_output/samCodeTlr_ChessValidator.csv -g
/workspace/project/src/goldstandard_samCode.csv
```

High level vs Detailed SAD

```
java -jar ../evaluator/evaluator.jar -t
/workspace/project/transarc_output/sadCodeTlr_ChessValidator.csv -g
/workspace/project/src/goldstandard_highlevel_sadCode.csv

java -jar ardoco-cli.jar -t SAD-Code -n ChessValidator -d /workspace/project/src/SAD_detailed.txt -m
/workspace/project/src/SAM_chessValidator.repository -c /workspace/project/src/main/java -o
/workspace/project/transarc_output

java -jar ../evaluator/evaluator.jar -t
/workspace/project/transarc_output/sadCodeTlr_ChessValidator.csv -g
/workspace/project/src/goldstandard_detailed_sadCode.csv
```