Flexible Lightweight Scalable Software Analysis

Rob Hackman

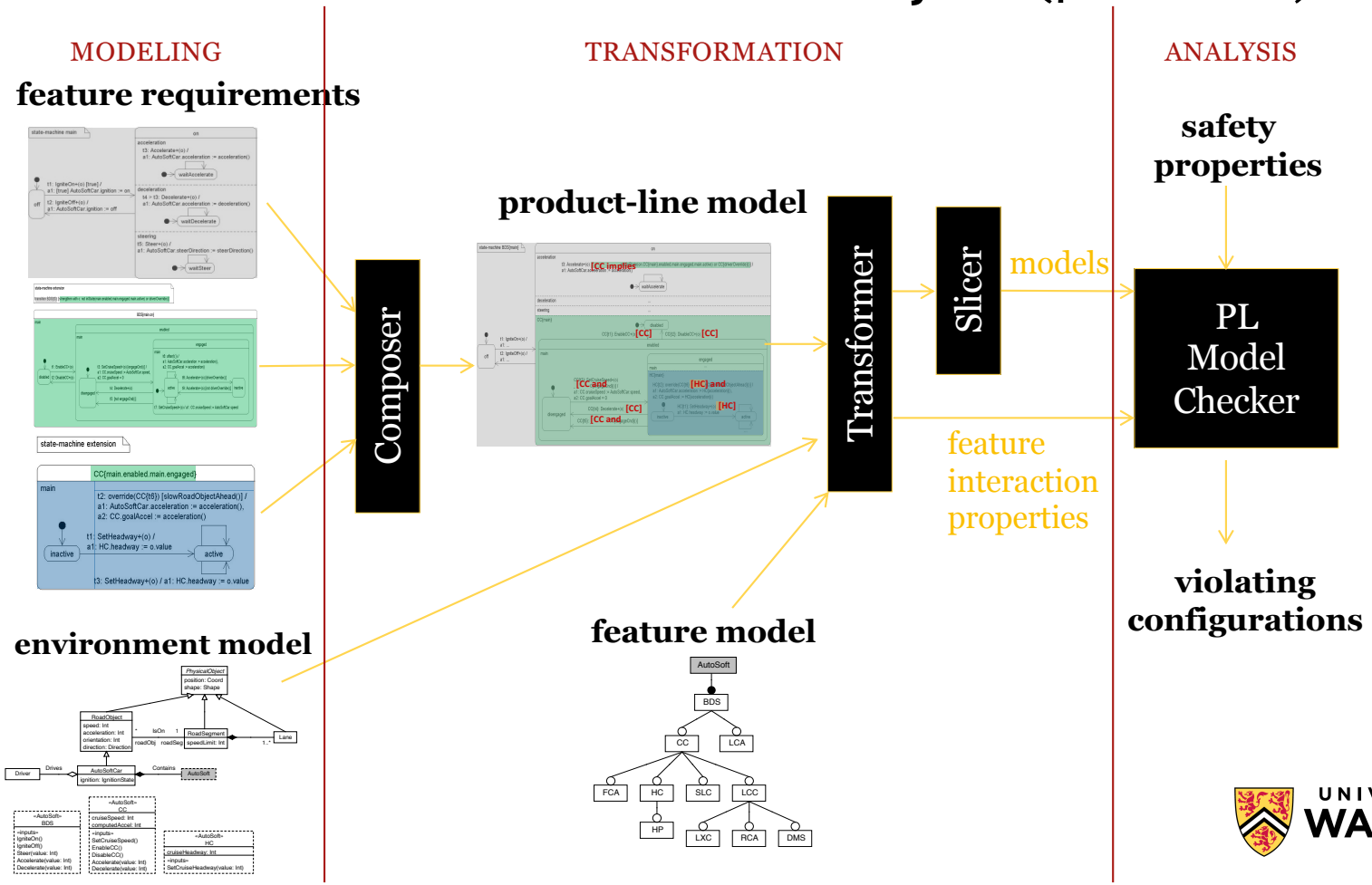
Prof. Jo Atlee

Prof. Mike Godfrey

Davood Anbarnam

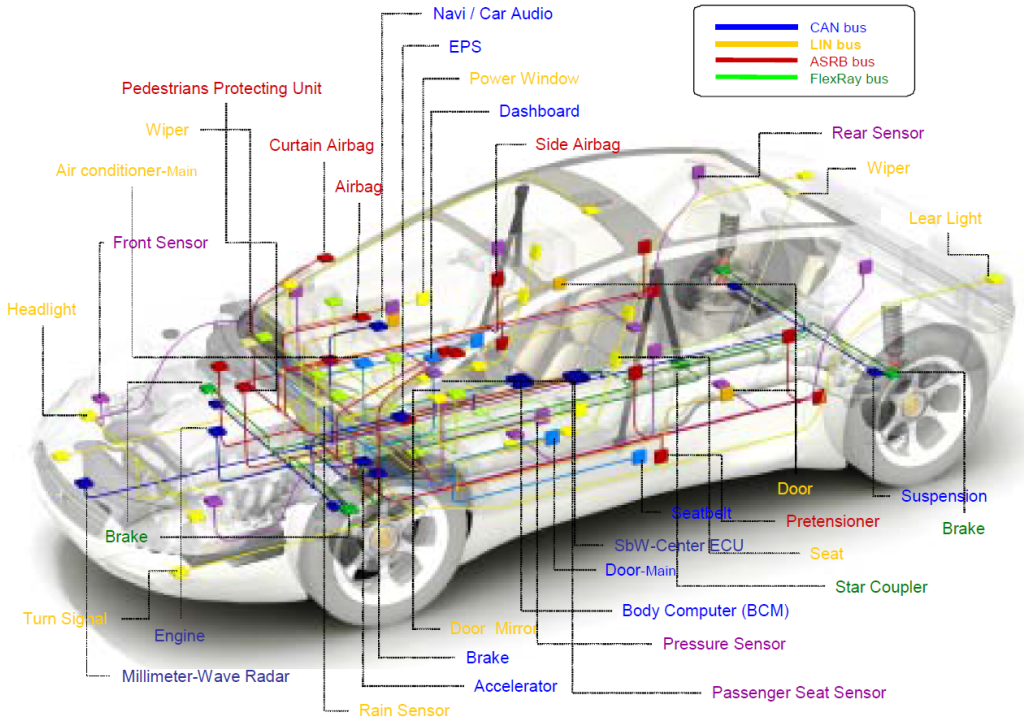
UNIVERSITY OF WATERLOO

model-based feature interaction analysis (past work)



UNIVERSITY OF WATERLOO

GOAL: system-wide interaction analysis

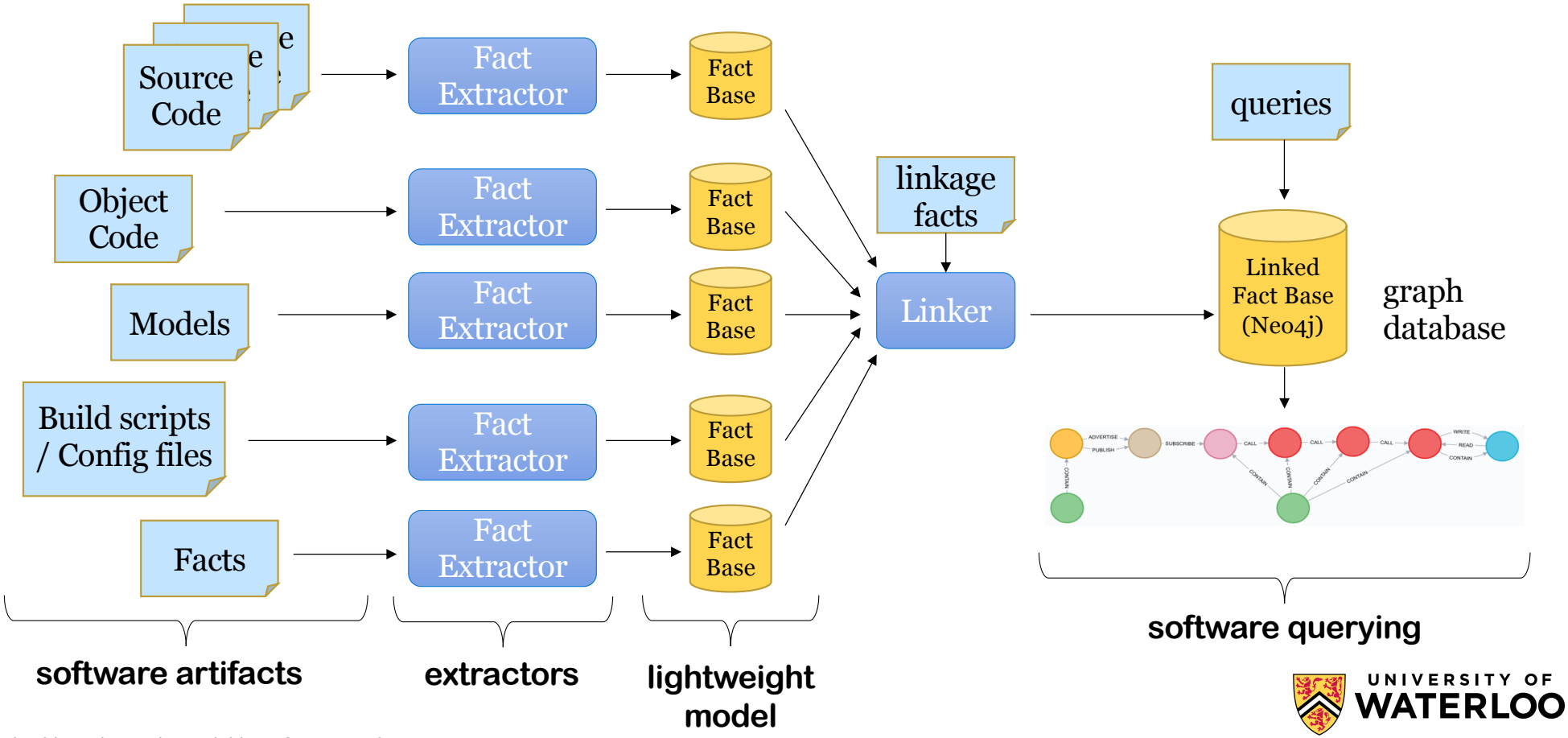


- No system-wide model
- Heterogeneous components
  - legacy, generated, third-party
  - distributed ECUs
  - bus-based communications
- 100 million lines of code (roughly)
- High variability (SPL)

http://www.flexautomotive.net/EMCFLEXBLOG/post/2015/09/08/can-bus-for-controller-area-network)

UNIVERSITY OF WATERLOO

flexible, lightweight, scalable software analysis



UNIVERSITY OF WATERLOO

facts (entities, relations, attributes)

entities
components
classes
functions
variables
ROS nodes

relationships
ROS node **receives** message from ROS node
variable on **RHS** of variable assignment
function **calls** function
function **reads** variable
function **writes** variable
class **contains** function

attributes
function is a callback
variable used in control-flow decision

UNIVERSITY OF WATERLOO

facts (entities, relations, attributes)

C++ Code

```
class Square {
public:
    int getArea() {
        int area = size * size;
        return area;
    }

    int size;
};
```

Factbase

FACT TUPLE :

$INSTANCE	Square	class
$INSTANCE	size	variable
$INSTANCE	getArea	function
$INSTANCE	area	variable

contain	Square	getArea
contain	Square	size
contain	getArea	area
varAssign	size	area
write	getArea	area

FACT ATTRIBUTE :

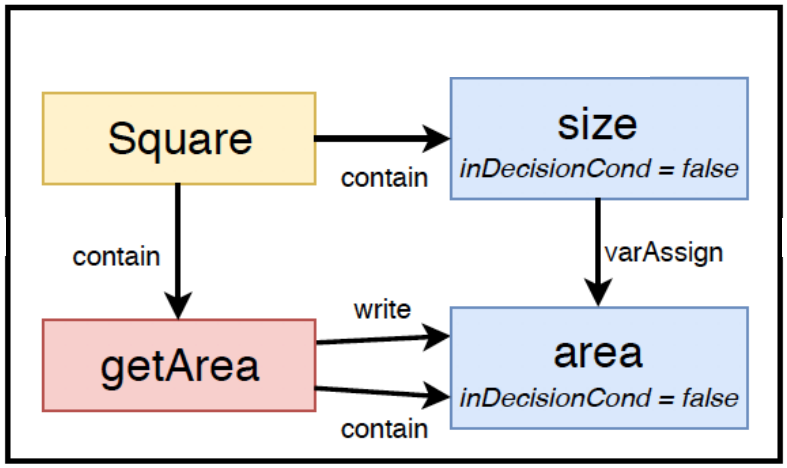
size { inDecisionCond = false }
area { InDecisionCond = false }

entities

relations

attributes

facts (graph database)



Factbase

FACT TUPLE :

$INSTANCE	Square	class
$INSTANCE	size	variable
$INSTANCE	getArea	function
$INSTANCE	area	variable

entities

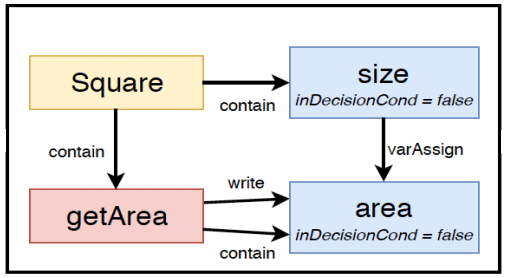
contain	Square	getArea
contain	Square	size
contain	getArea	area
varAssign	size	area
write	getArea	area

relations

FACT ATTRIBUTE :
size { inDecisionCond = false }
area { InDecisionCond = false }

attributes

query language (over a graph database)



General Path Queries

- Call flows
- Information flows (assignments, parameter passing, message passing)

Interaction Queries

- Communication loops (possible delays, nontermination)
- Race conditions (possible nondeterminism)
- Multiple inputs (possible livelock)
- Control-flow alteration (variable assignment flows to control-flow decision)

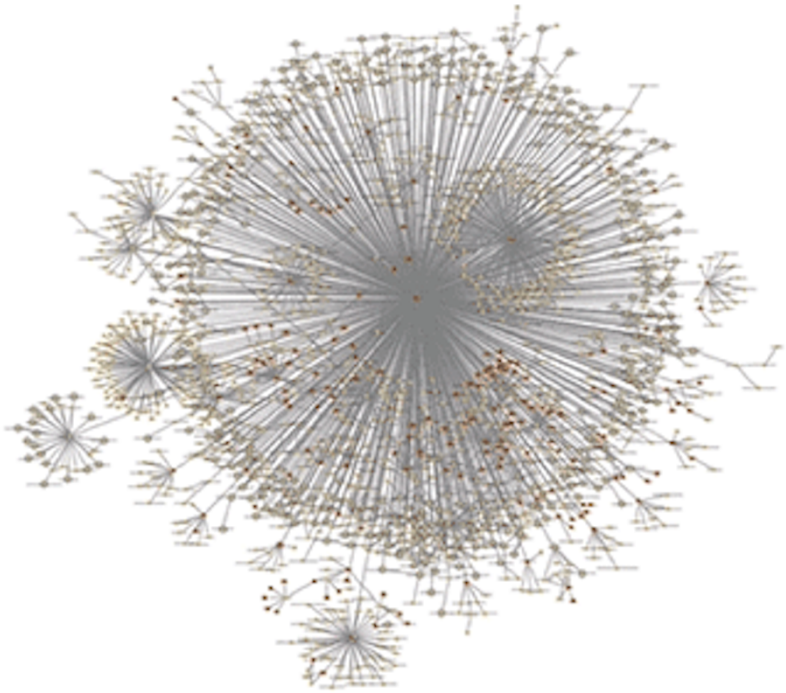
UNIVERSITY OF WATERLOO

Demo

scalability (large case study)

Source Code Files (#)	9,289
Source code (LoC)	1,475,643*
Header Files (#)	10,131
Header Files (LoC)	623,068*
Generated Nodes (#)	206,531
Generated Relationships (#)	590,463

*generated using David A. Wheeler's 'SLOCCount'



Nodes = "module" = group of files with same prefix
Edges = inter-module communication

scalability (large case study)

Example: Control-Flow Alteration Query

Detect whether or not a function call to some function **`bar`** depends on some global variable **`globVar`**, which could have been written to by another function **`foo`**

```
int globVar;
void foo() {
  . . .
  globVar = . . .;
  . . .
}

. . .
if (globVar) bar();
```

UNIVERSITY OF WATERLOO

scalability (large case study)

Interactions among Modules (#) **pairs of components (not paths)**	587, 998
Behaviour Alterations among Functions (#) **paths**	6,231,572
Behaviour Alterations among Functions (#) **paths (without loops)**	32,287
Behaviour Alterations among Functions (#) **paths (without loops &** **involving more than one module)**	27,479

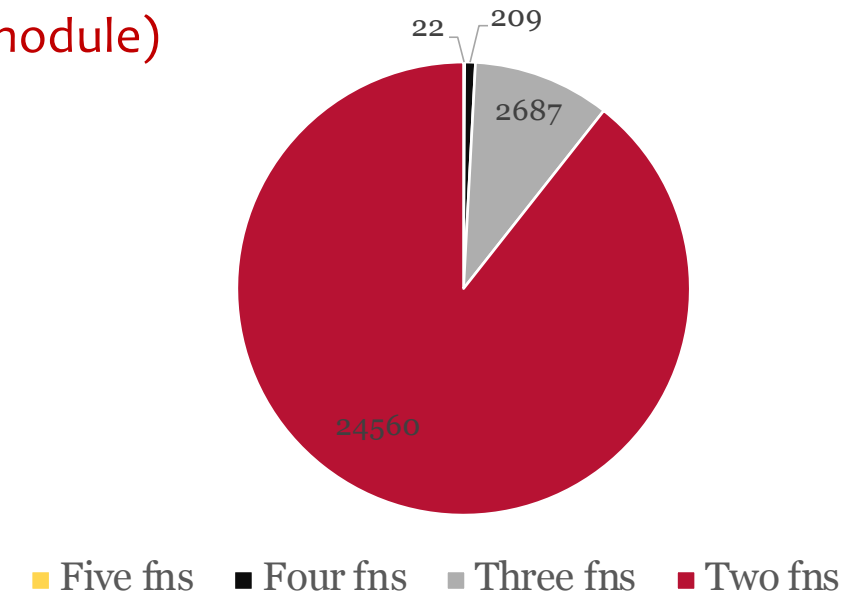
UNIVERSITY OF WATERLOO

scalability (large case study)

behaviour alterations among functions

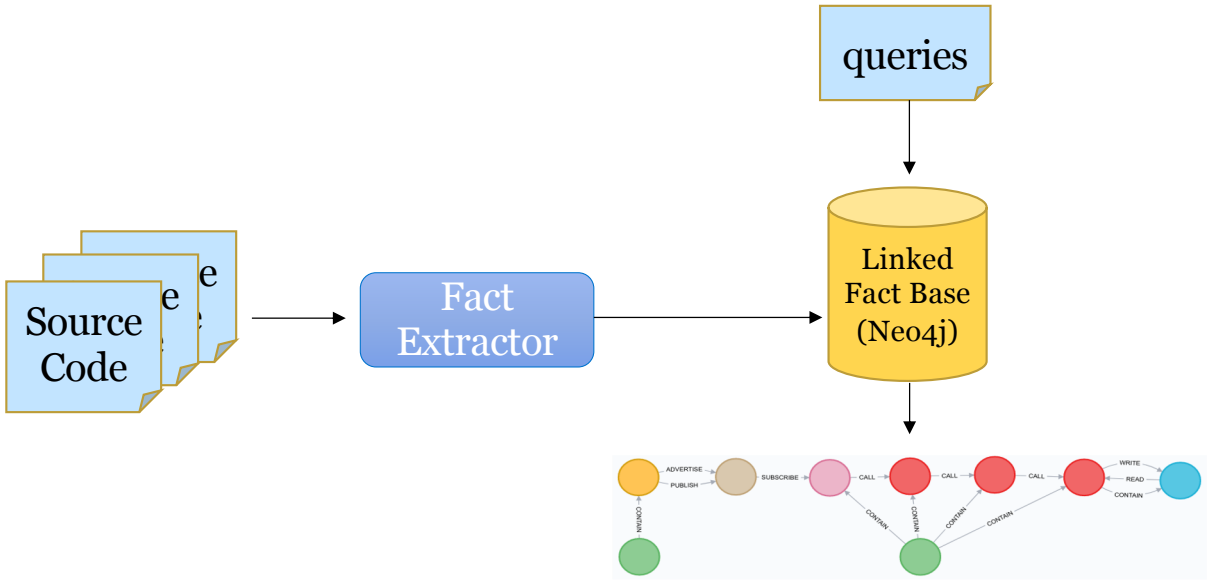
paths (without loops & involving more than one module)

- Of the 27,479 detected paths, most are small
- **Triage:** Higher path rate may indicate higher chance of error



current work (precision)

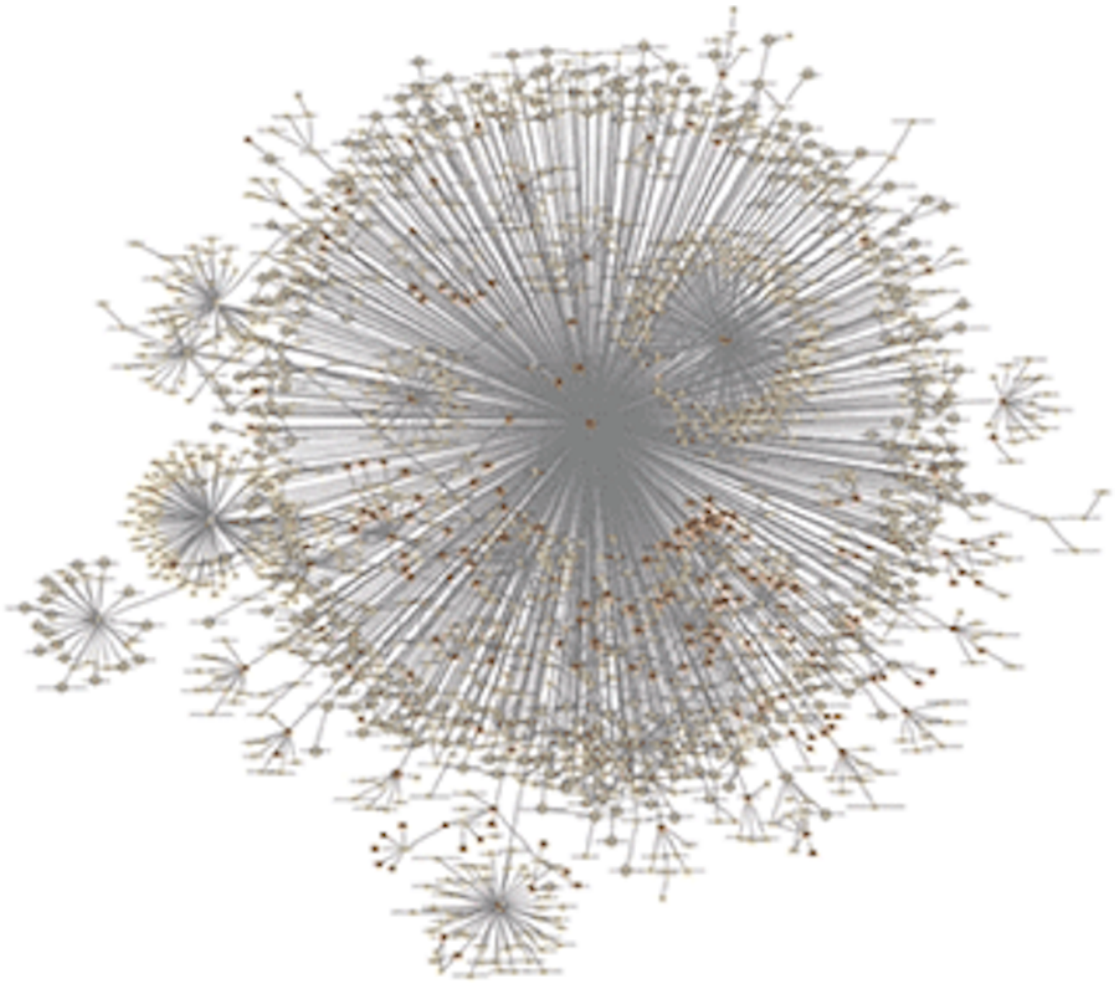
1) Improving the **precision** of the fact extraction and static analysis



UNIVERSITY OF WATERLOO

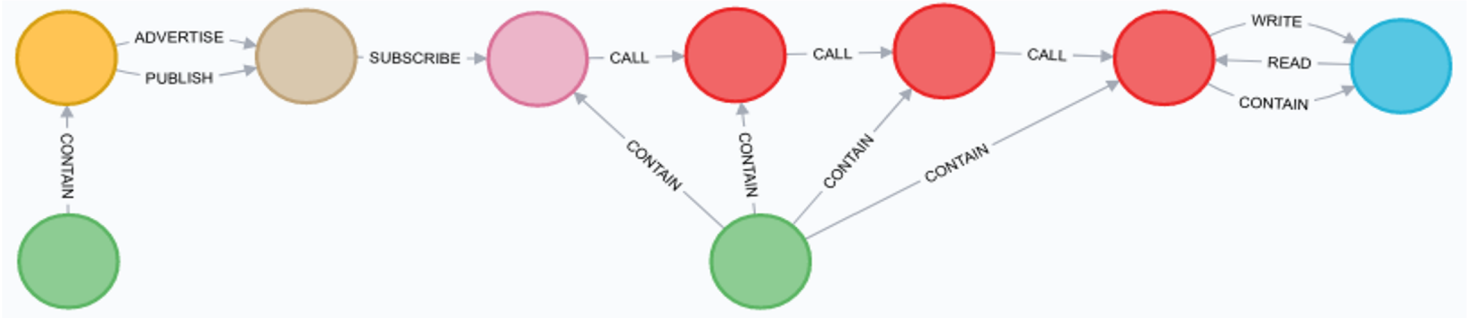
current work (scalability)

2) Improving the **scalability** of path queries



current work (user experience)

3) Reporting and **visualizing** query results



4) Improving the **user experience** in making follow-up queries

UNIVERSITY OF WATERLOO

current work (variability)

5) Investigating **variability**-aware interaction analysis

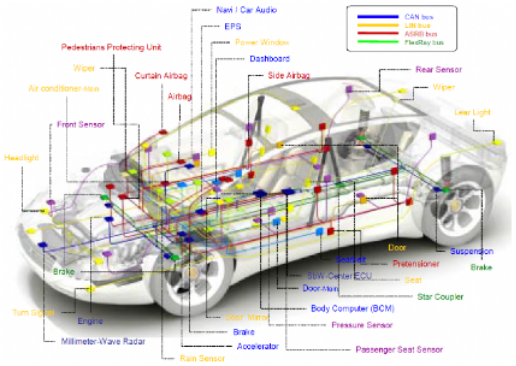
```
1  class BaseFeature { ... };
2  class FeatureA : public BaseFeature { ... };
3  class FeatureB : public BaseFeature { ... };
4
5  bool config_A; // Given a value somehow
6
7  int main ( ) {
8      BaseFeature *p;
9      if ( config_A ) {
10         FeatureA *a = new FeatureA;
11         p = a;
12     } else {
13         FeatureB *b = new FeatureB;
14         p = b;
15     }
16     p->execute ( );
17 }
```

- extracting the conditions under which facts hold
- querying over configurations of factbases
- exploring configurations of factbases

Shahin, Chechik, Salay. “Lifting Datalog-Based Analyses to Software Product Lines”, FSE19.

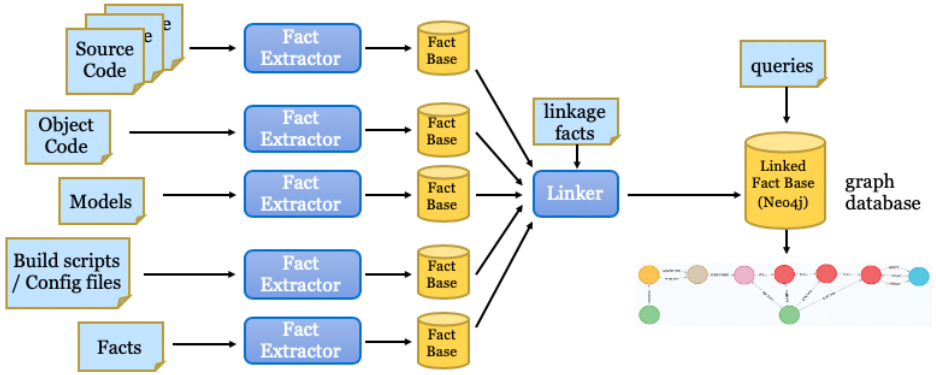
UNIVERSITY OF WATERLOO

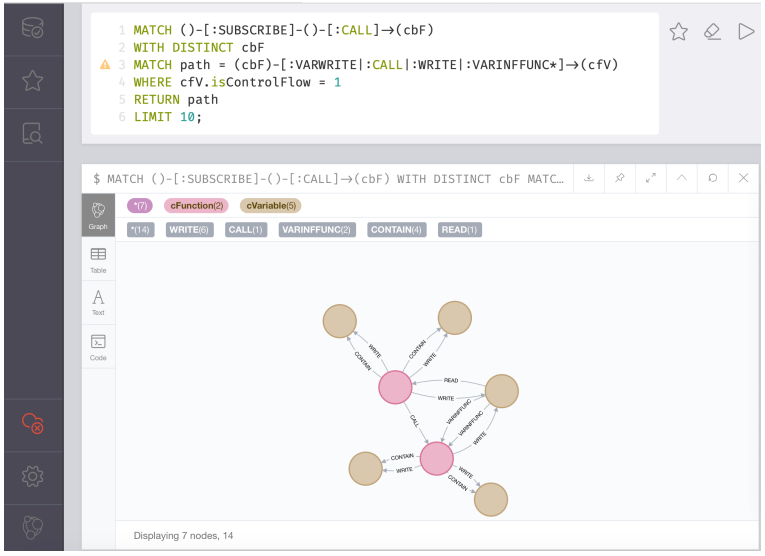
GOAL: system-wide interaction analysis



- No system-wide model
- Heterogeneous components
  - legacy, generated, third-party
  - distributed ECUs
  - bus-based communications
- 100 million lines of code (roughly)
- High variability (SPL)

flexible, lightweight, scalable software analysis

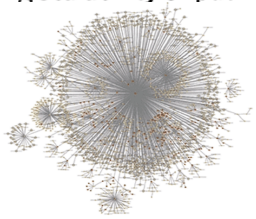




current work

Improving **precision** of the extraction and analysis

Improving **scalability** of path queries



Reporting and **visualizing** query results

Improving the **user experience** in making follow-up queries

Supporting **variability**-aware extraction, analyses, and exploration