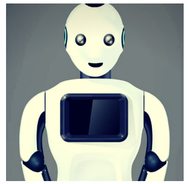
Let Your Questions Guide Your Analysis

Software Analysis using Natural Language Queries

Pooja Rani
Ph.D Student
Software Composition Group
University of Bern, Switzerland



Hello, Pooja

Hey! Mitra

Which classes are deprecated in my project?

I found 25 classes which are annotated with “Deprecated”

Which classes among these are not used by other classes?

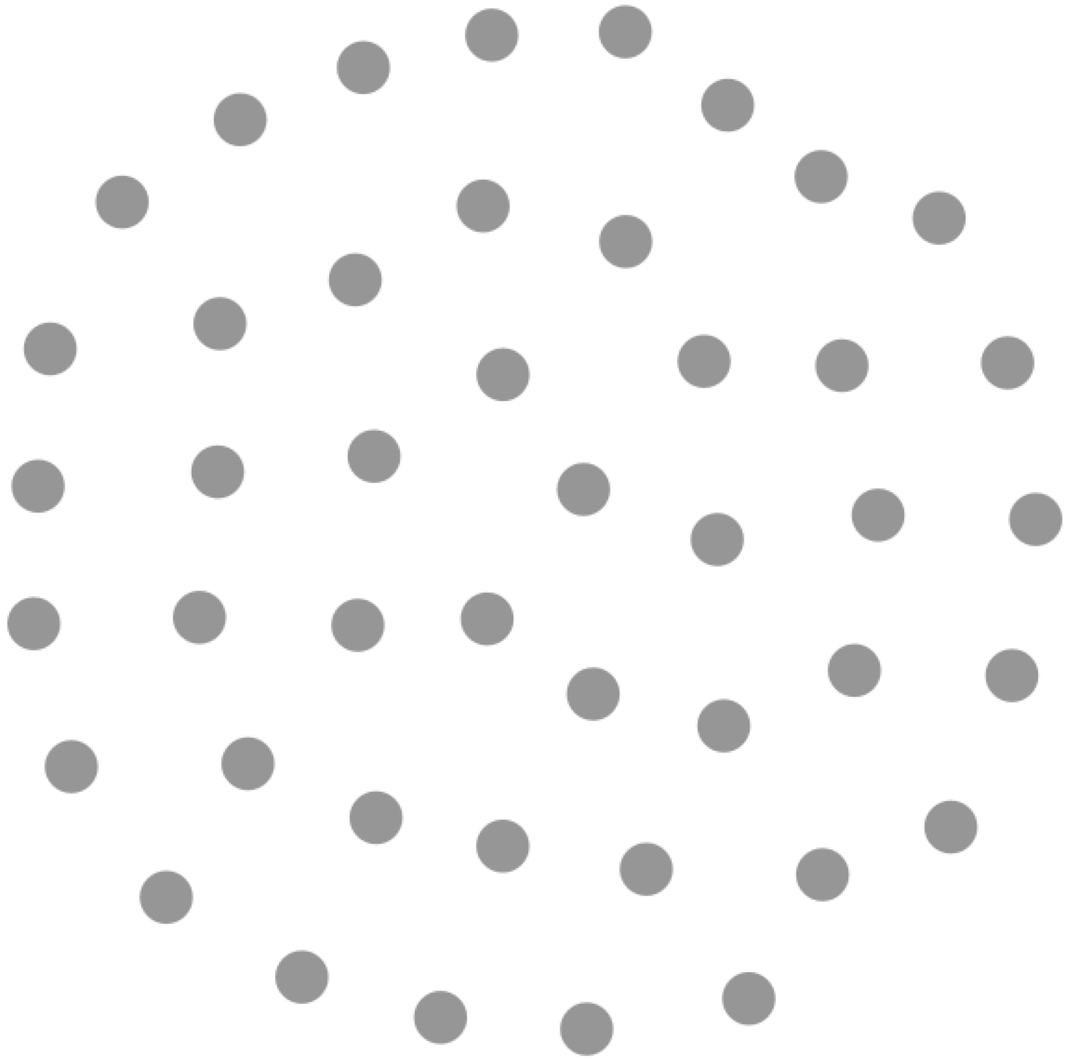
I found 14 deprecated classes which are not used by other classes.

Can you delete these 14 classes?

Yes, sure

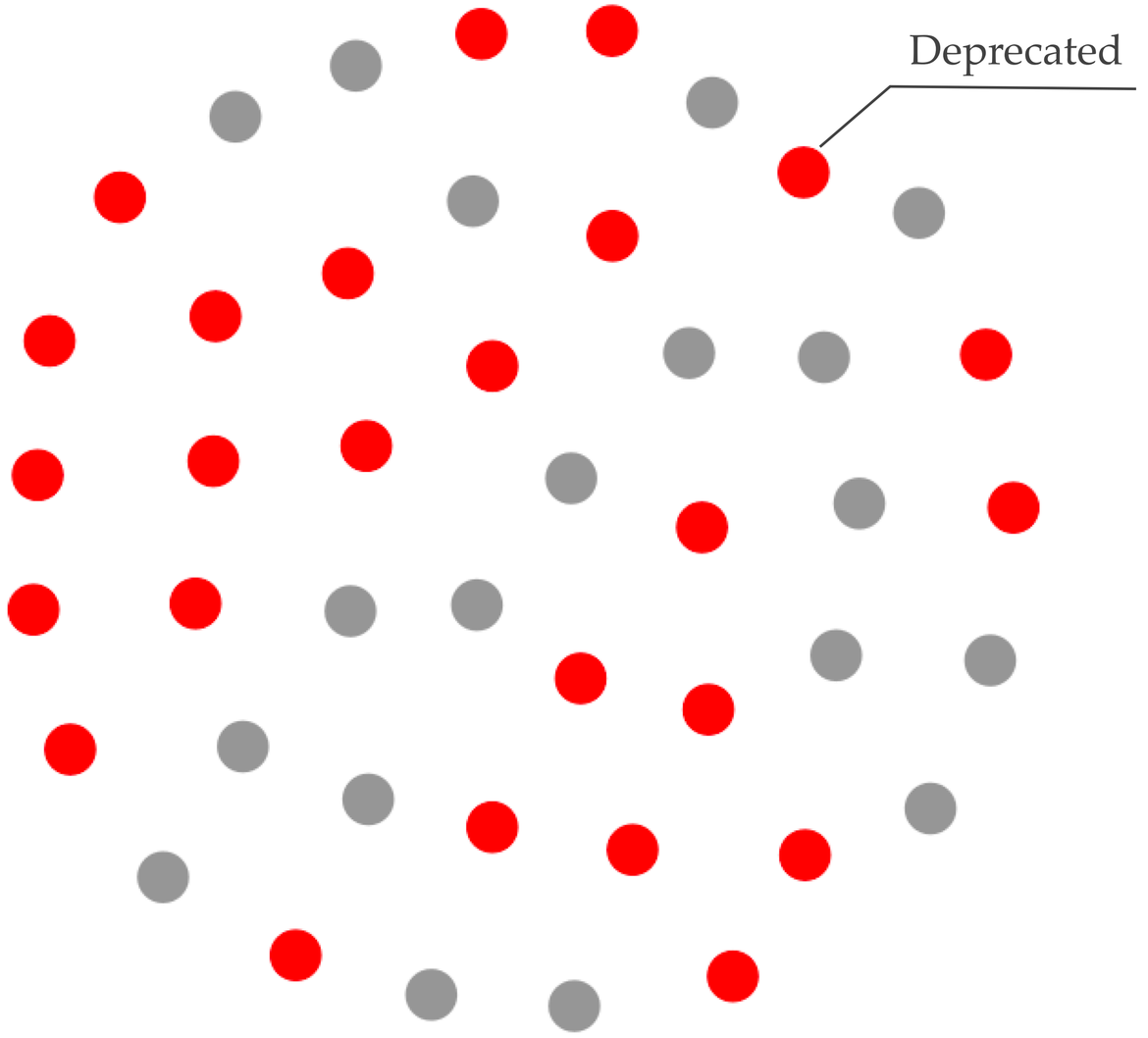
We want to remove deprecated classes

Find all the classes in your project



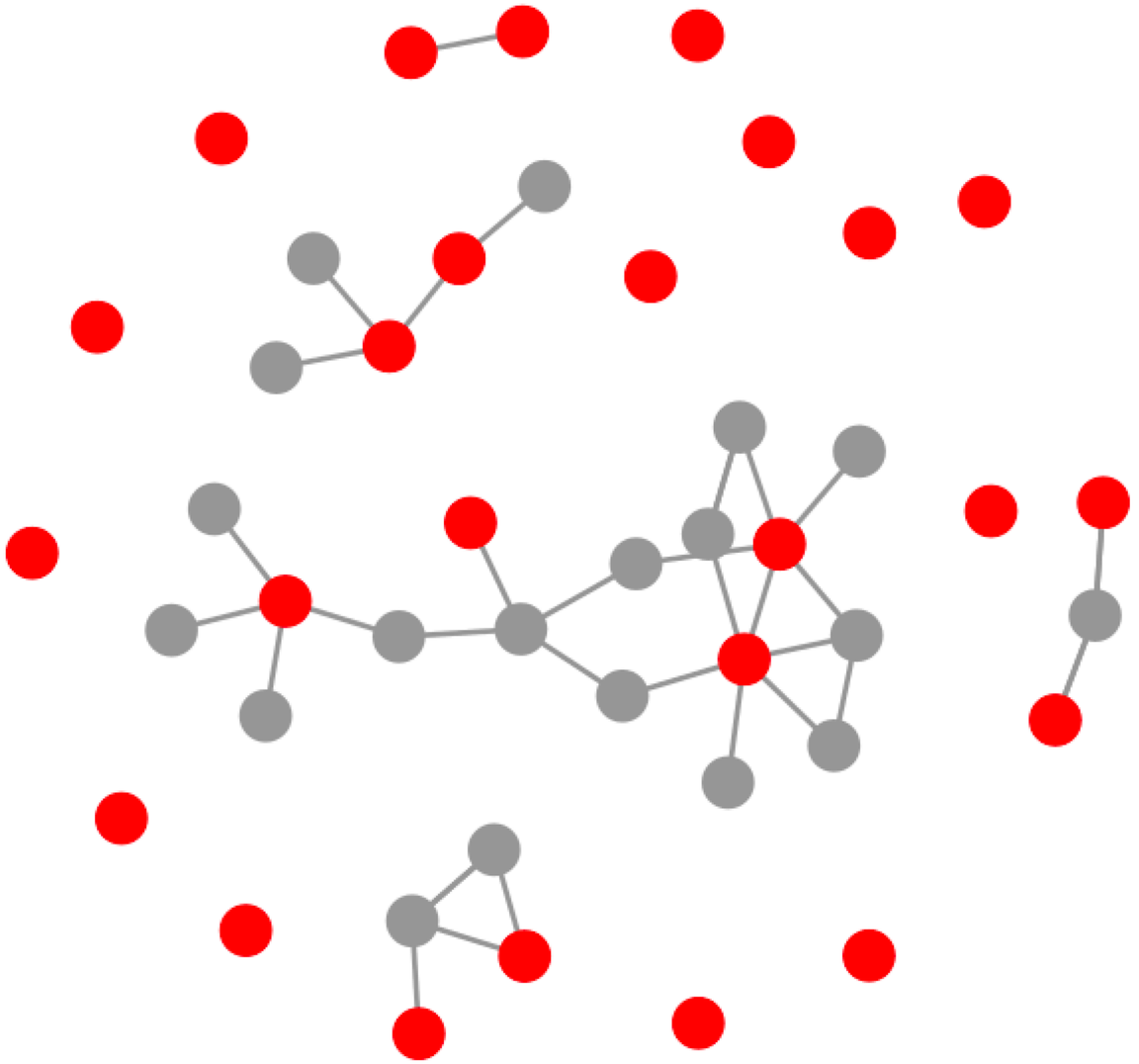
44 classes

Which classes are deprecated?



25 classes

Which other classes are using these deprecated classes?



Where are we now?

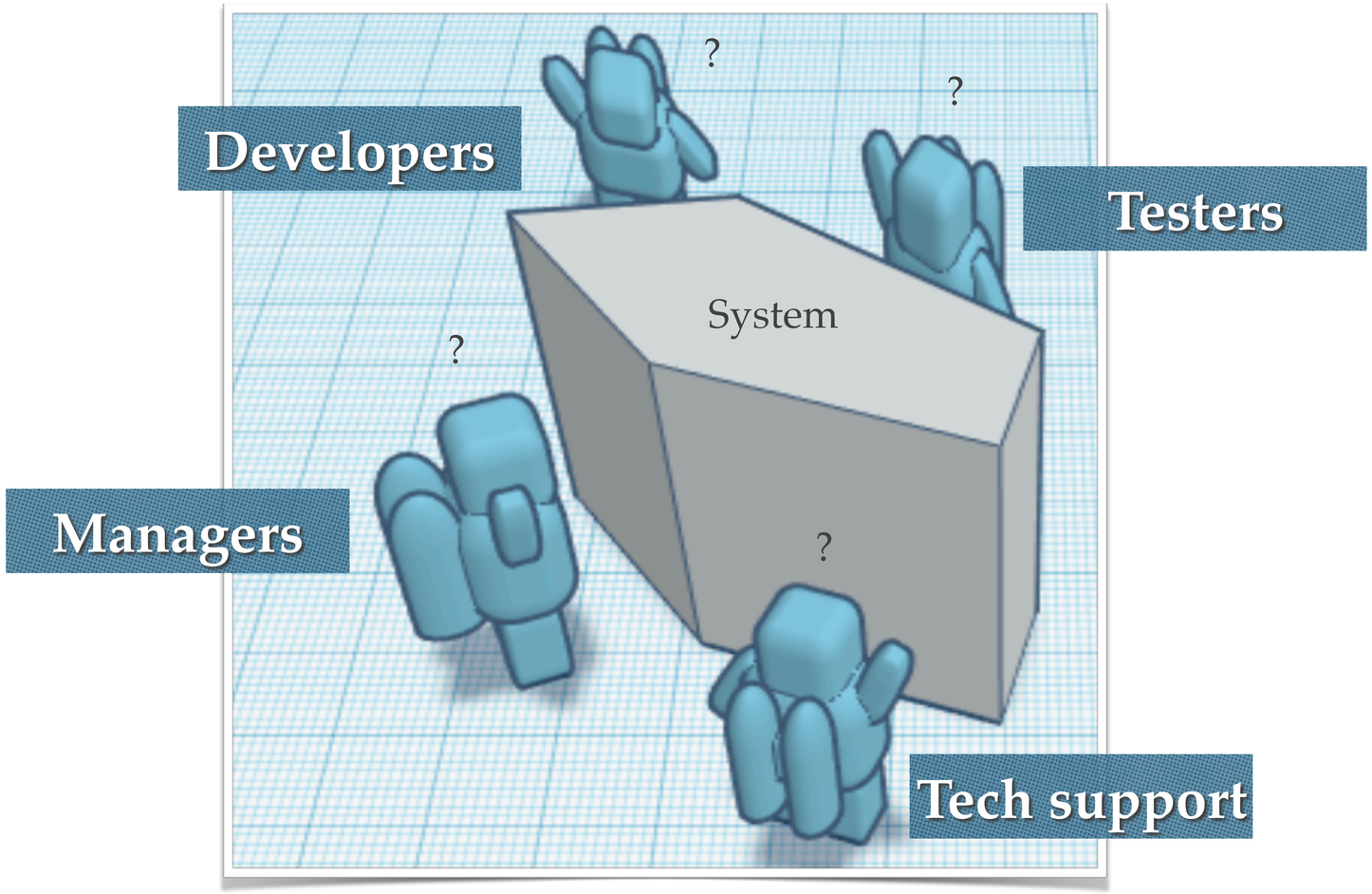
Why can not we analyse our software speculatively and so naturally?

Speculative

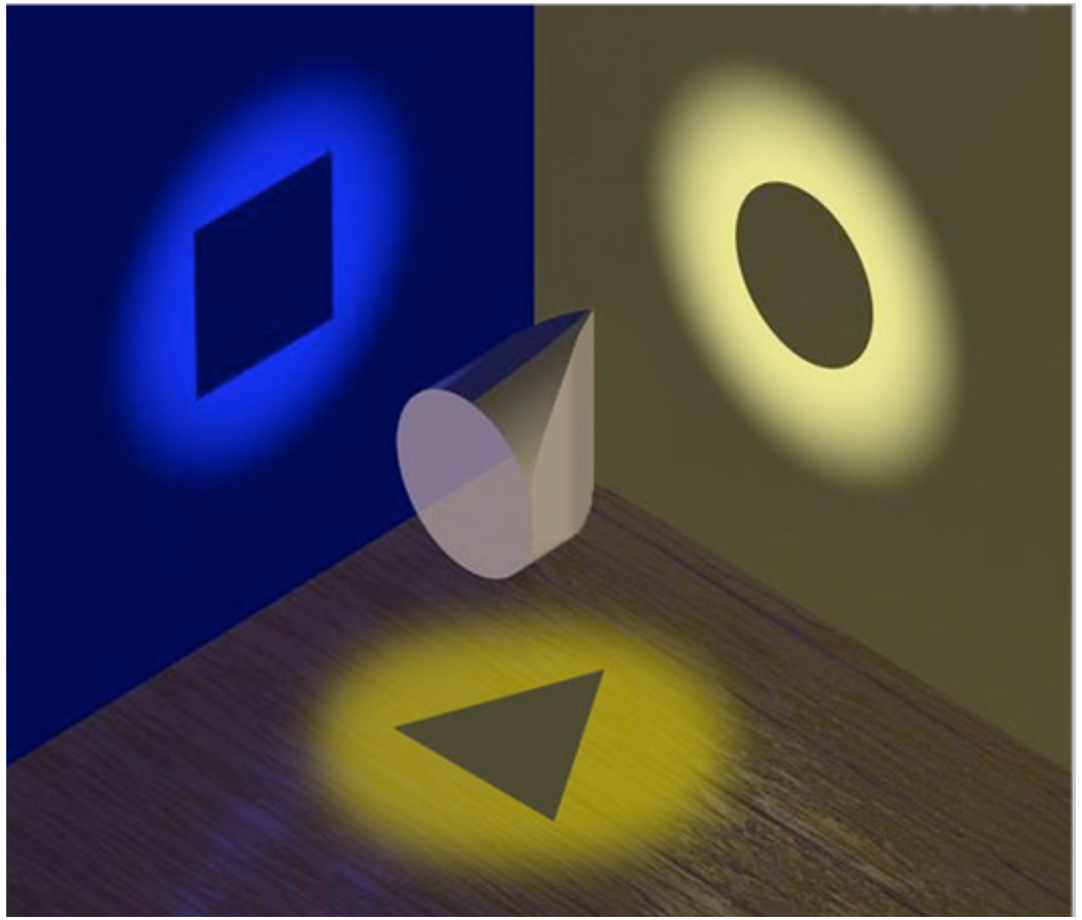
Done in order to make profit in the future but with risk too.

Here risk means computation power and initial hard work to train

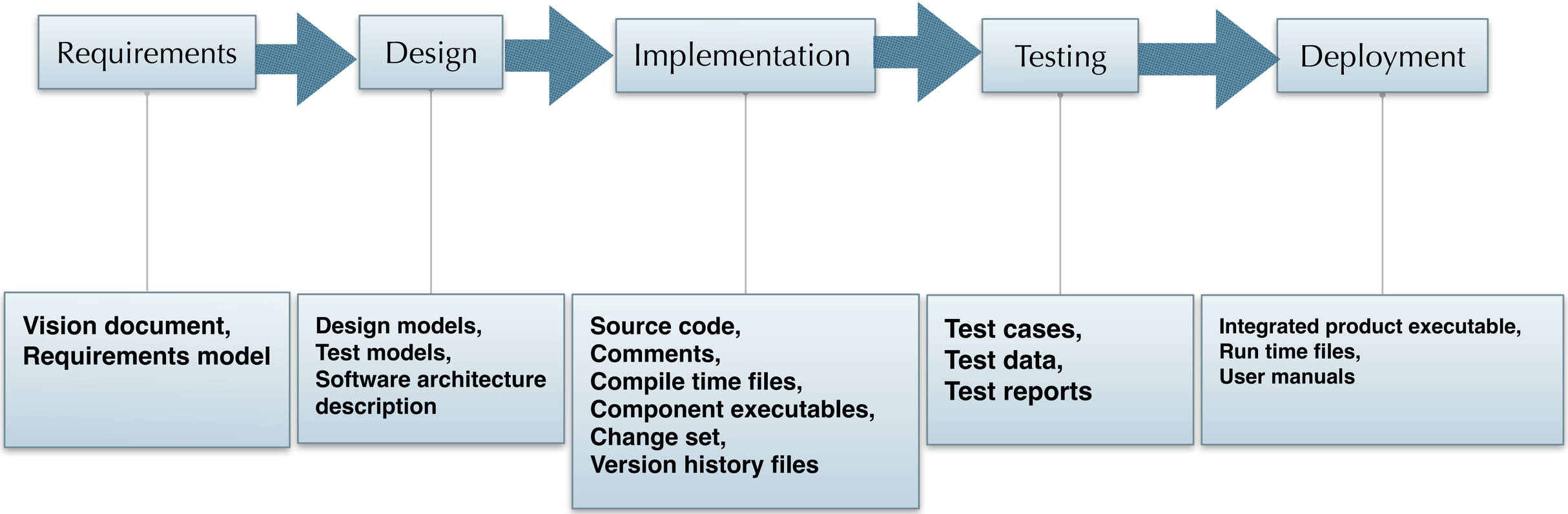
Loss means we can get negative result also as our semantic tools might give negative result


Developers
Testers
Managers
Tech support
System
?
?
?
?

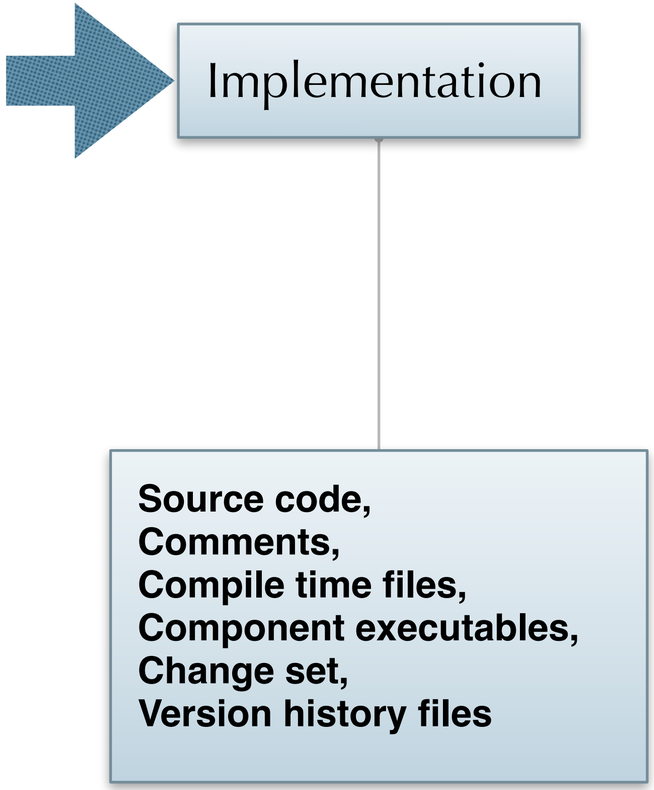
Tools for each perspective

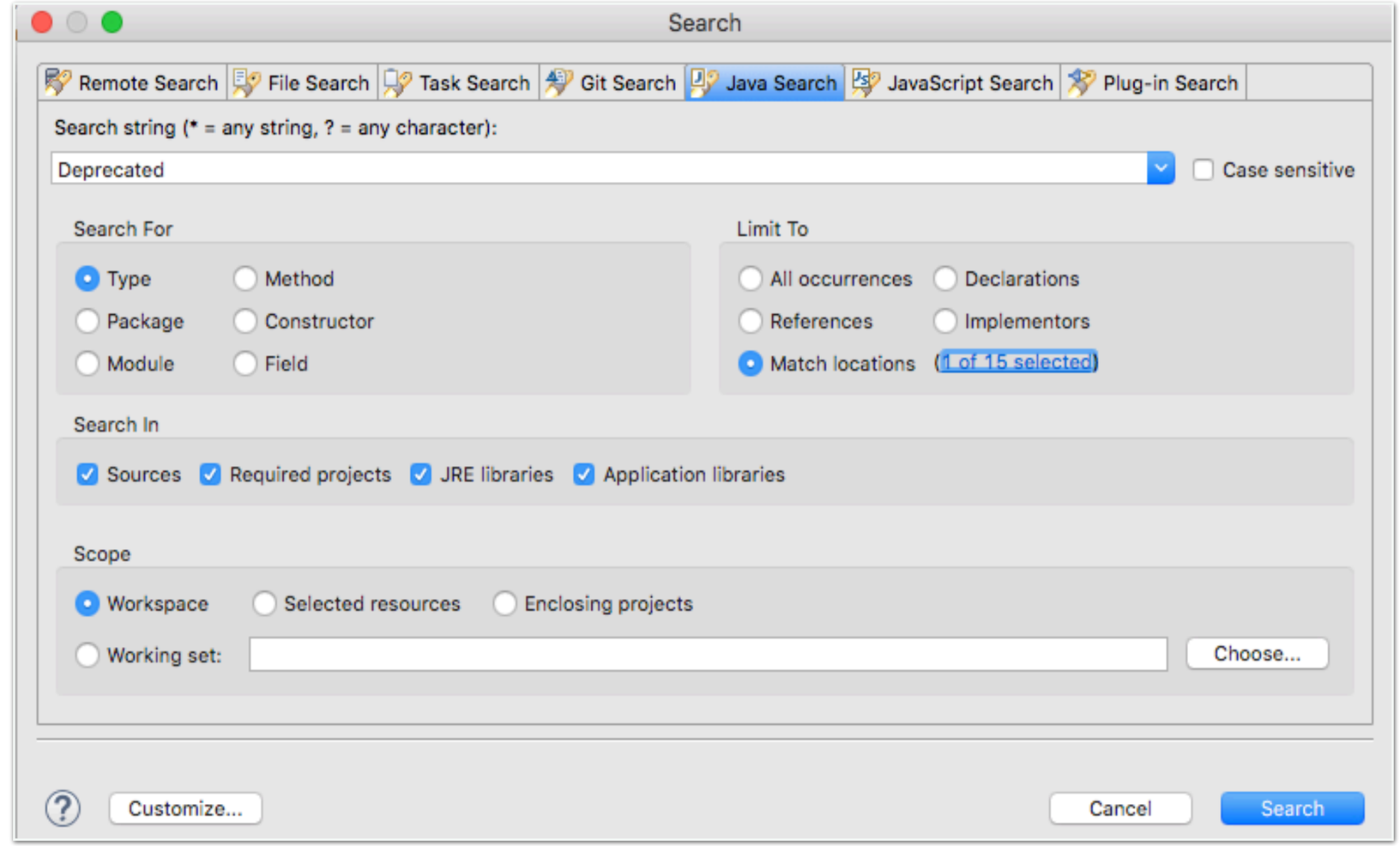


Multiple artifacts at each phase



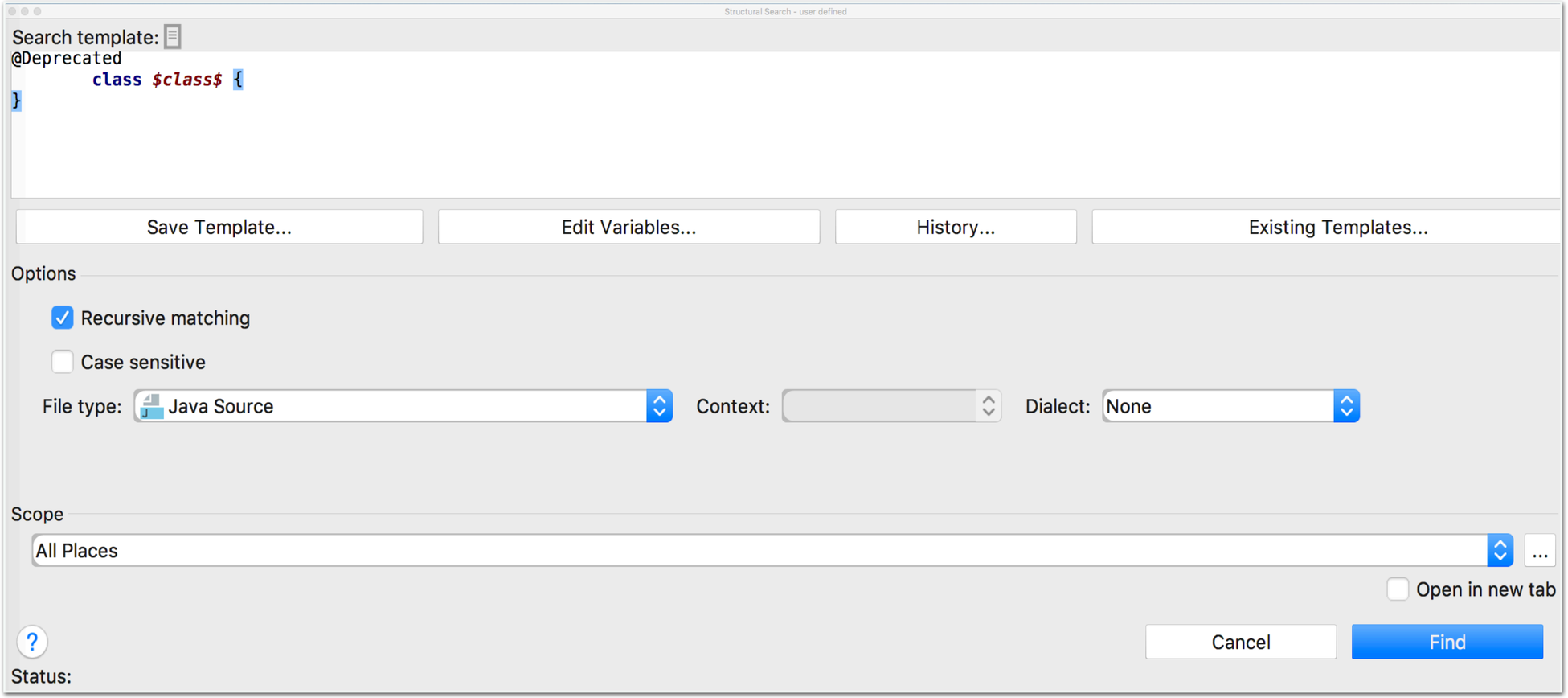
How do we analyse our source code?





Limits us to the given options

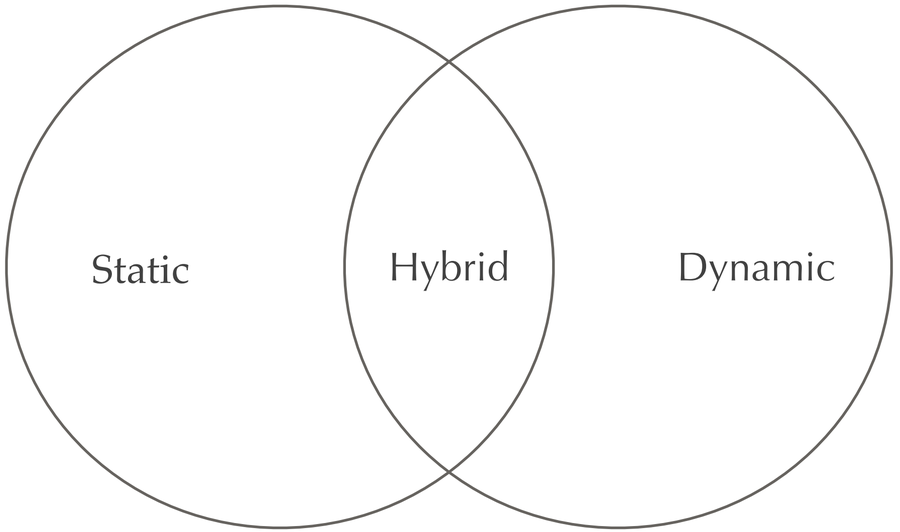
Intellij



Complex to express query

Software analysis requires us to have tools that go far beyond simple search bars

There are source code analyzers



Daikon

dynamic analysis tool

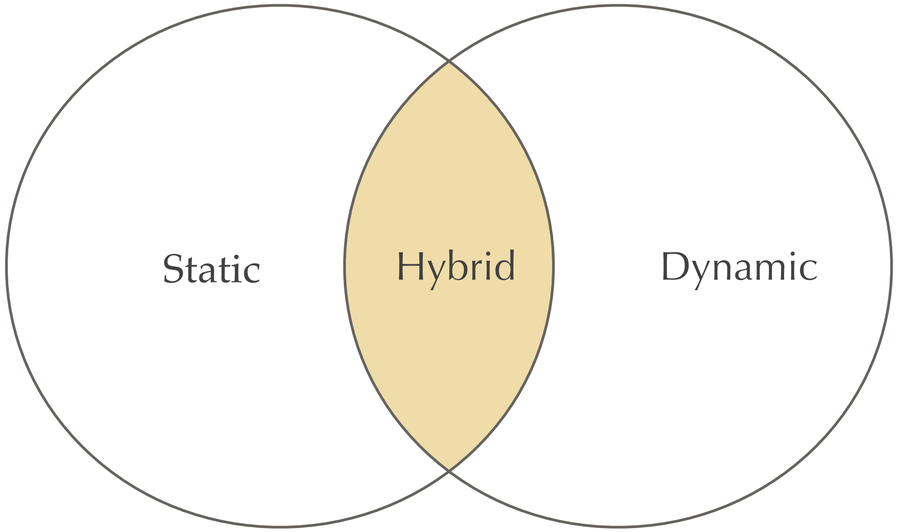
srcML

static analysis tool

Moose

software analysis platform

We will take a hybrid tool



Daikon

dynamic analysis tool

srcML

source code analysis tool

Moose

software analysis platform

Query in Moose

```
self allModelClasses
   select: [ :eachClass | eachClass isAnnotatedWith: `Deprecated`]
```

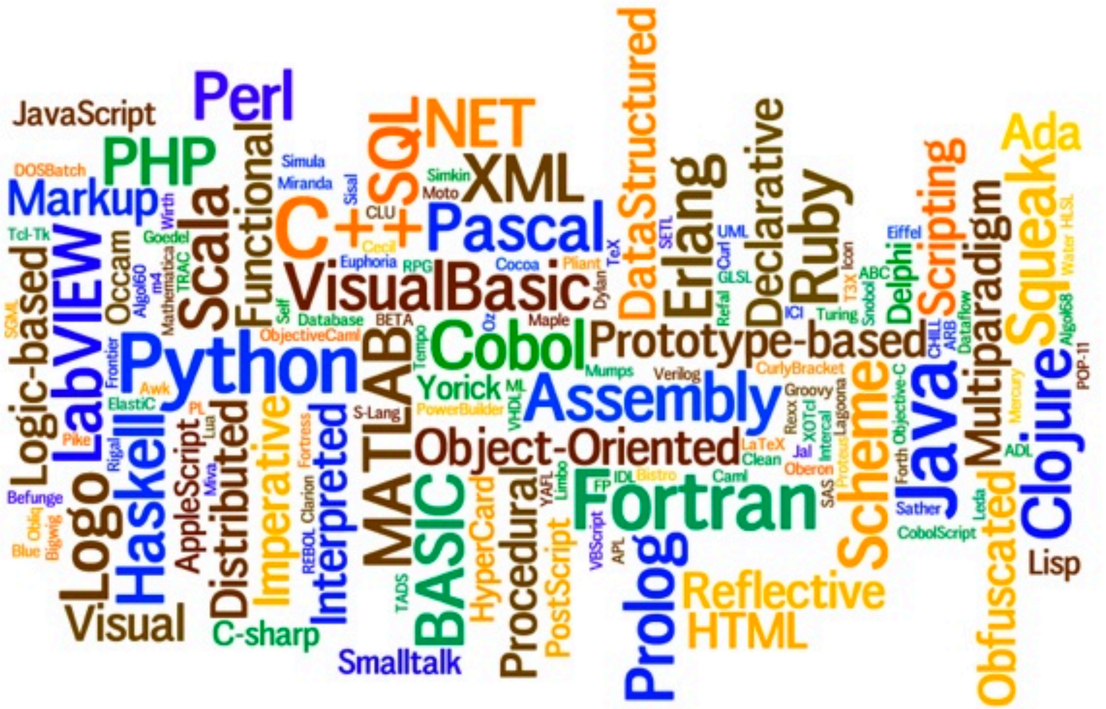
Query in srcML

```
%srcml –xpath=
“//src:class[.//src:annotation src:name=‘Deprecated’]src:annotation”
linux-4.0.3.tar.xz.xml -o class_annotation.xml
```

Each tool has its own language

We need to learn
a new language
to communicate with
these analysis tools.

Learning a new language
is time-consuming



Just learn one!

Your Own.

Natural Language

Ask questions

Which classes are the deprecated?

Find the class named “Date”?

Where is method named “searchForThreat” implemented?

Where is method named “getAmount” implemented?

Goal

Which classes are deprecated?

```
self allModelClasses select:
                [ :eachClass |
                eachClass isAnnotatedWith:`Deprecated’]
```

Translation seems easy, but is it really?

Which classes are deprecated?

All classes that should no longer be used?

} Semantically same

Which client classes are using the deprecated classes?

Which deprecated classes are used by any client class?

} Same words, different structure

Target language is a programming language

Recurrent Neural Networks (RNN)

Challenges

➔ Different analysis tools

➔ Challenge of communication and structure of tool’s query language

➔ Train neural network for each target query language

➔ Prepare corpus of questions

➔ Semantic tools are not ready for software engineering domain

Milestones

Literature review, interviews

Find common query pattern

Neural network based technique

Collect developer questions

Establish query pattern

Experimentation with various techniques

Syntax based technique

Work in progress

Evaluation

Translate natural language question from English to query language

Natural language interface for analysis tools

Communication barrier for new developers

This does not solve our problem completely

Why can not we analyse our software speculatively and so naturally?

This does not solve our problem completely

Why can not we analyse our software speculatively and so naturally?

This does not solve our problem completely

Why can not we analyse our software speculatively and so naturally?

This does not solve our problem completely

Why can not we analyse our software speculatively and so naturally?

Development context missing

Customization of tools

Analysis tools integration

Speculative Analysis

Analysis is an iterative and continuous process

Identify development context, mine and extract useful software information

present result and receive feedback from developer

Integrate analysis into development process

Conclusion

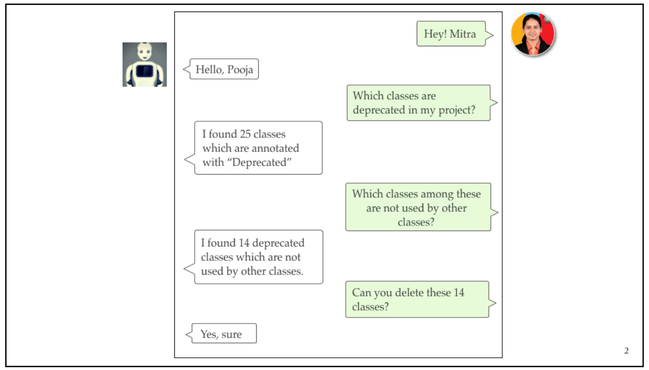
Various software roles - **Let us craft our own tools**

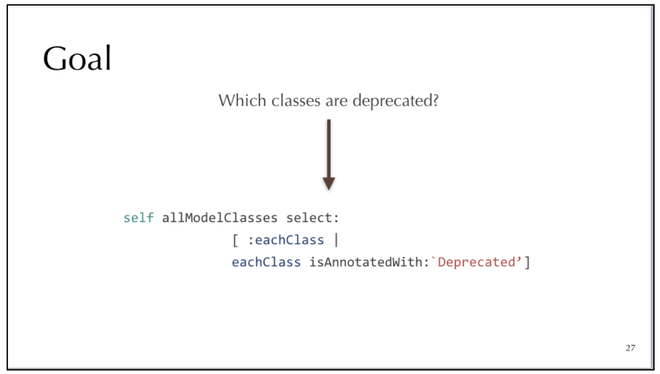
Different software Artifacts - **Customize your analysis according to data**

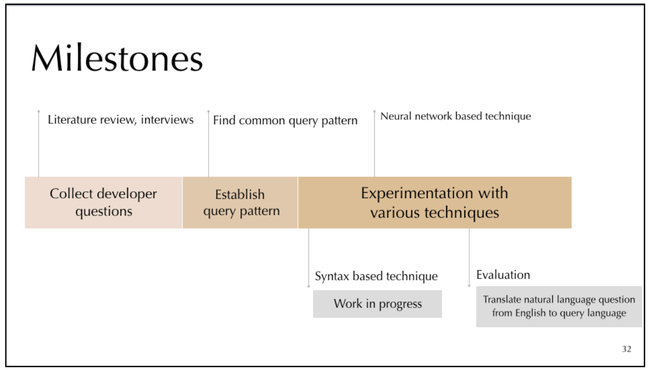
Complex interaction with the tools - **Unify the analysis tools**

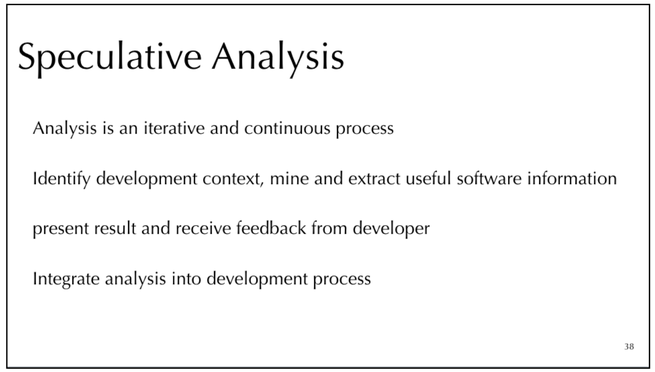
Continuous Analysis- **Integrate analysis tools into software development process completely**

Summary









scg.unibe.ch/staff/Pooja-Rani

pooja@inf.unibe.ch

twitter.com/poojaruhal