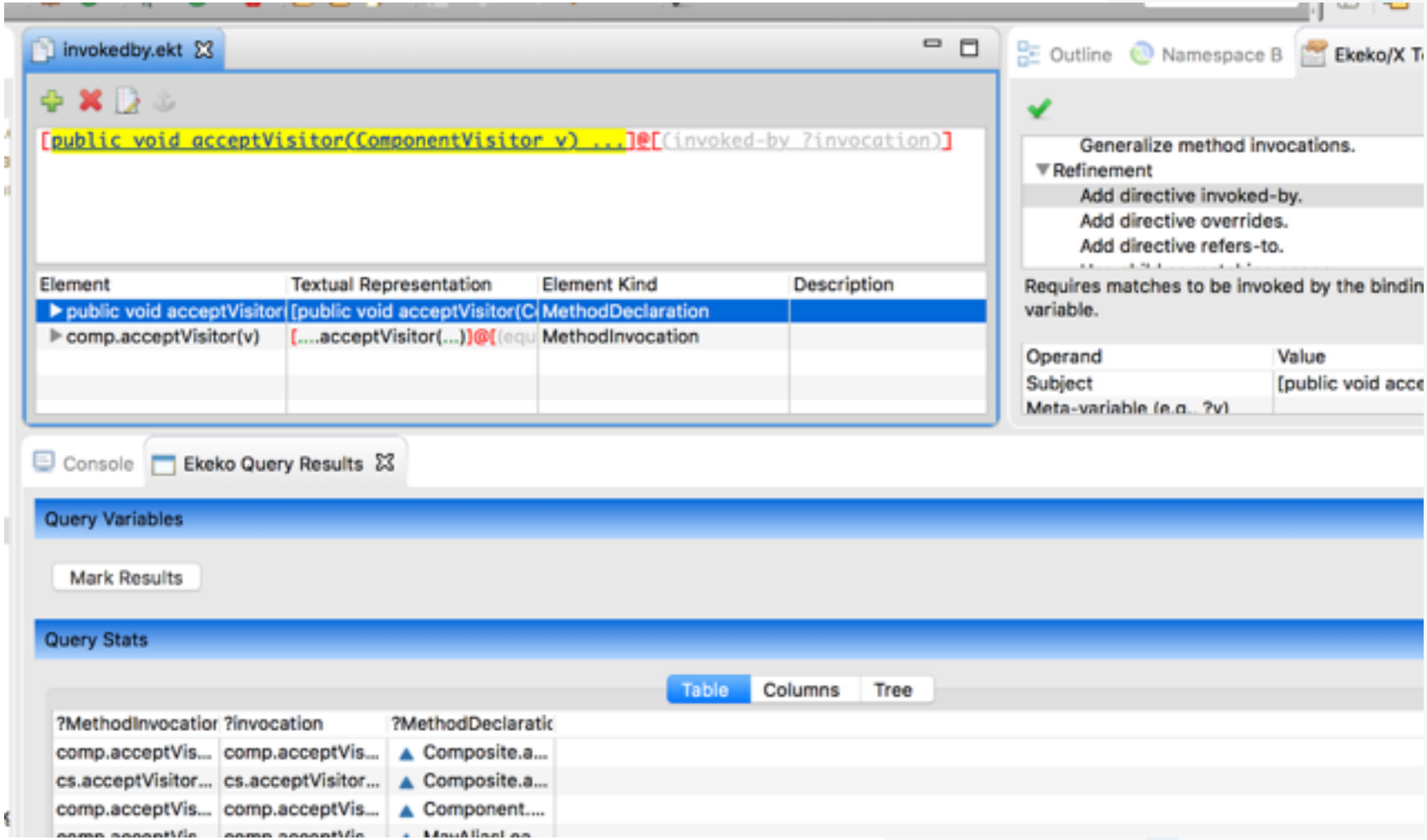
A Recommender System for Cha-Qeko/X Source Code Templates

Presenter: Tim Molderez

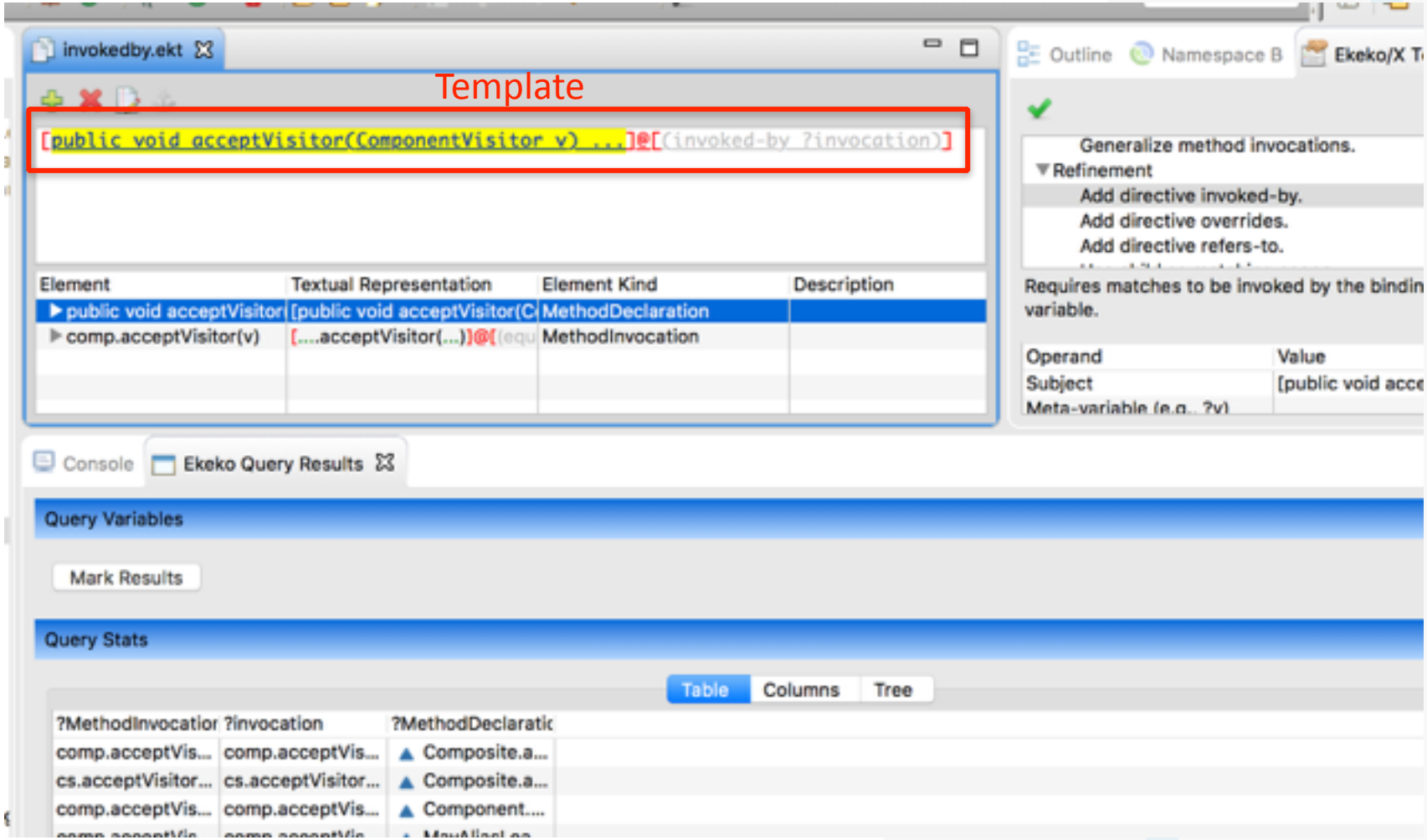
Universiteit Antwerpen

Vrije Universiteit Brussel

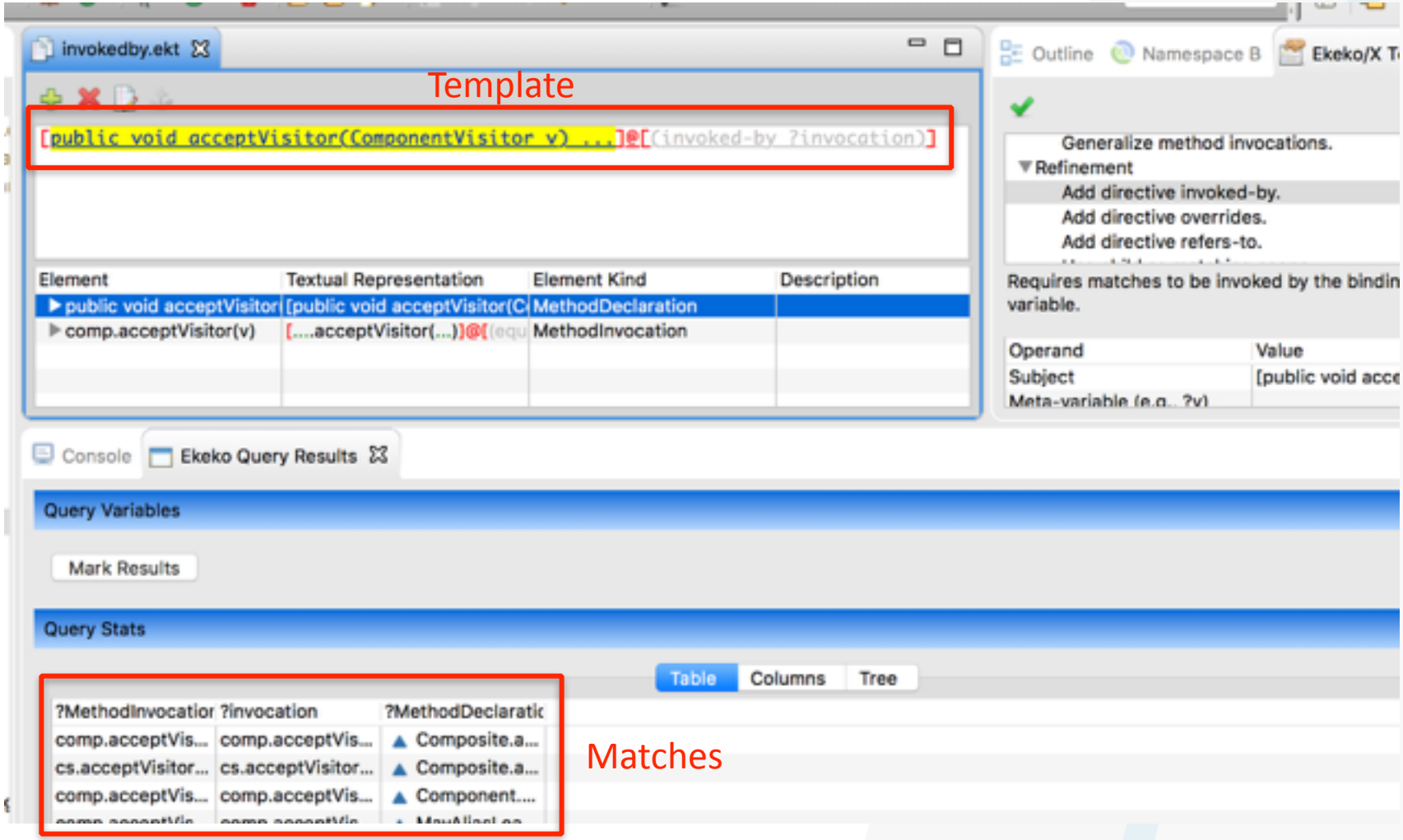
Source code templates in Cha-Qeko/X



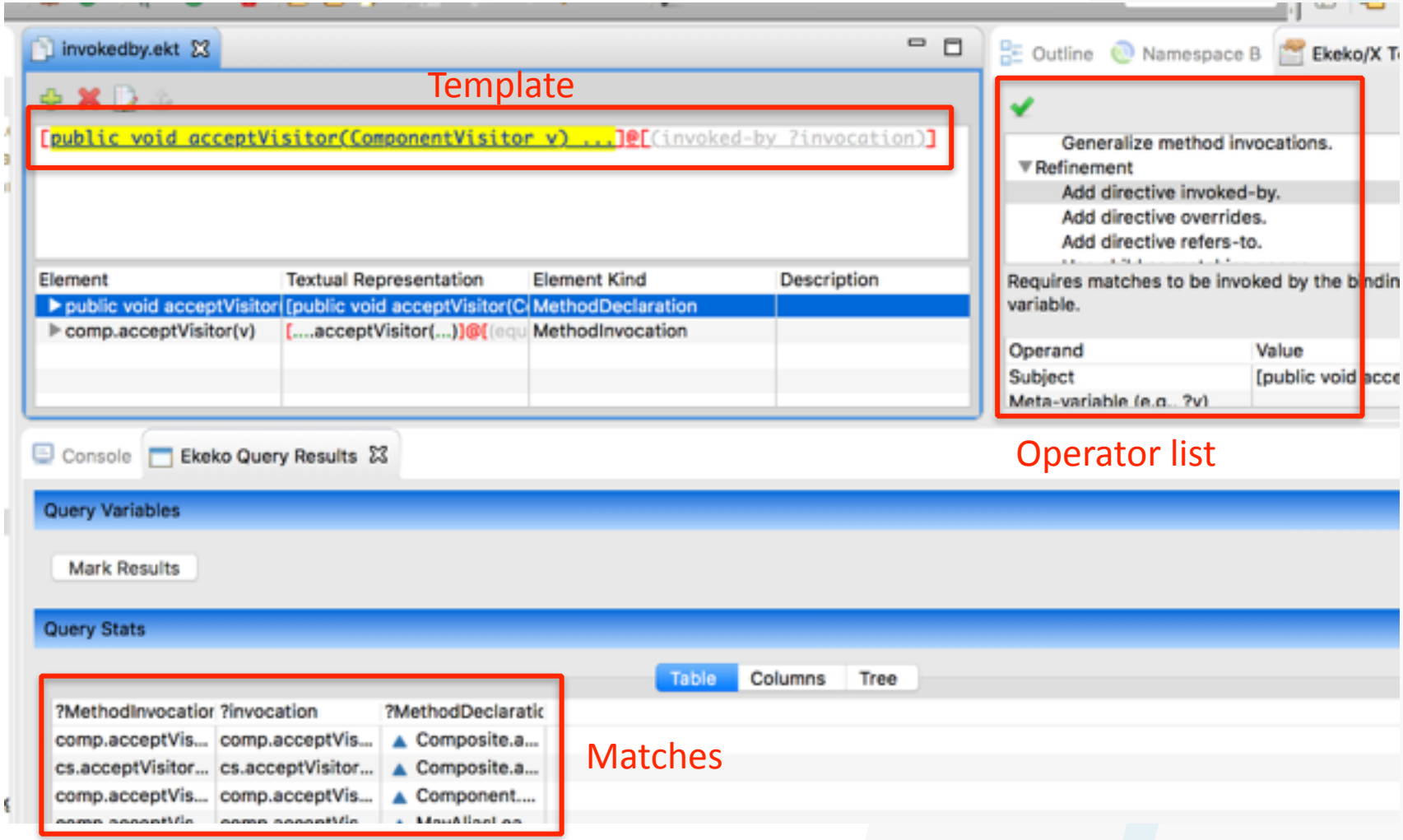
Source code templates in Cha-Qeko/X



Source code templates in Cha-Qeko/X



Source code templates in Cha-Qeko/X



Motivation

- Code templates lower the learning curve of specifying program searches/modifications
- Hard to master: not always evident to produce only the desired matches

Example

An acceptVisitor declaration-call pair:

```
public void acceptVisitor(ComponentVisitor v) {
  Iterator<Component> i=elements.iterator();
  while (i.hasNext()) {
    Component comp=(Component)i.next();
    comp.acceptVisitor(v);
  }
}

cs.acceptVisitor(vstor);
```

Example

And another:

```
public void acceptVisitor(ComponentVisitor v) {
     Iterator<Component> i = elements.iterator();
     while (i.hasNext()) {
         Component comp = (Component) i.next();
         comp.acceptVisitor(v);
     }
}

comp.acceptvisitor(v);
```

Example

And another:

```
public void acceptVisitor(ComponentVisitor v) {
     System.out.println("Prototypical.");
     v.visitPrototypicalLeaf(this);
}

comp.acceptvisitor(v);
```

Example

… and several others!

How to write a template that describes all of these?

Example

Initial attempt:

```
public void acceptVisitor(...) ...

comp.acceptvisitor(v);
```

Too few matches!

Example

Second attempt:

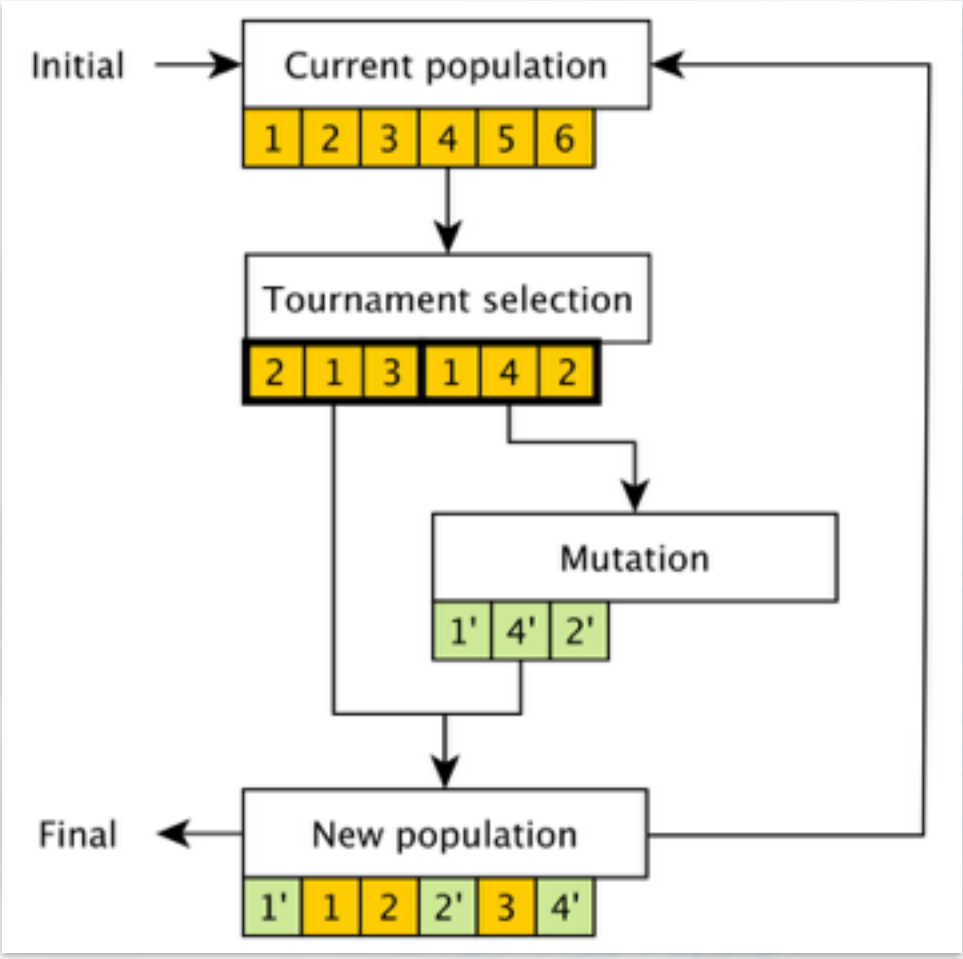
```
public void acceptVisitor(...) ...
....acceptvisitor(...);
```

Too *many* matches!

Recommending edits

- How to assist users if a template produces too few/many matches?
- Genetic search approach: Find a sequence of edits leading to more desired matches
- User marks what is desired/undesired

Genetic algorithm



Fitness function

- Determines “how good” a template is
- Produces a number between 0 and 1
- Consists of two components:
  - F-score: how many (un)desired matches?
  - Partial score: For each desired match, what % of the template corresponds to the desired match?

Recommender system

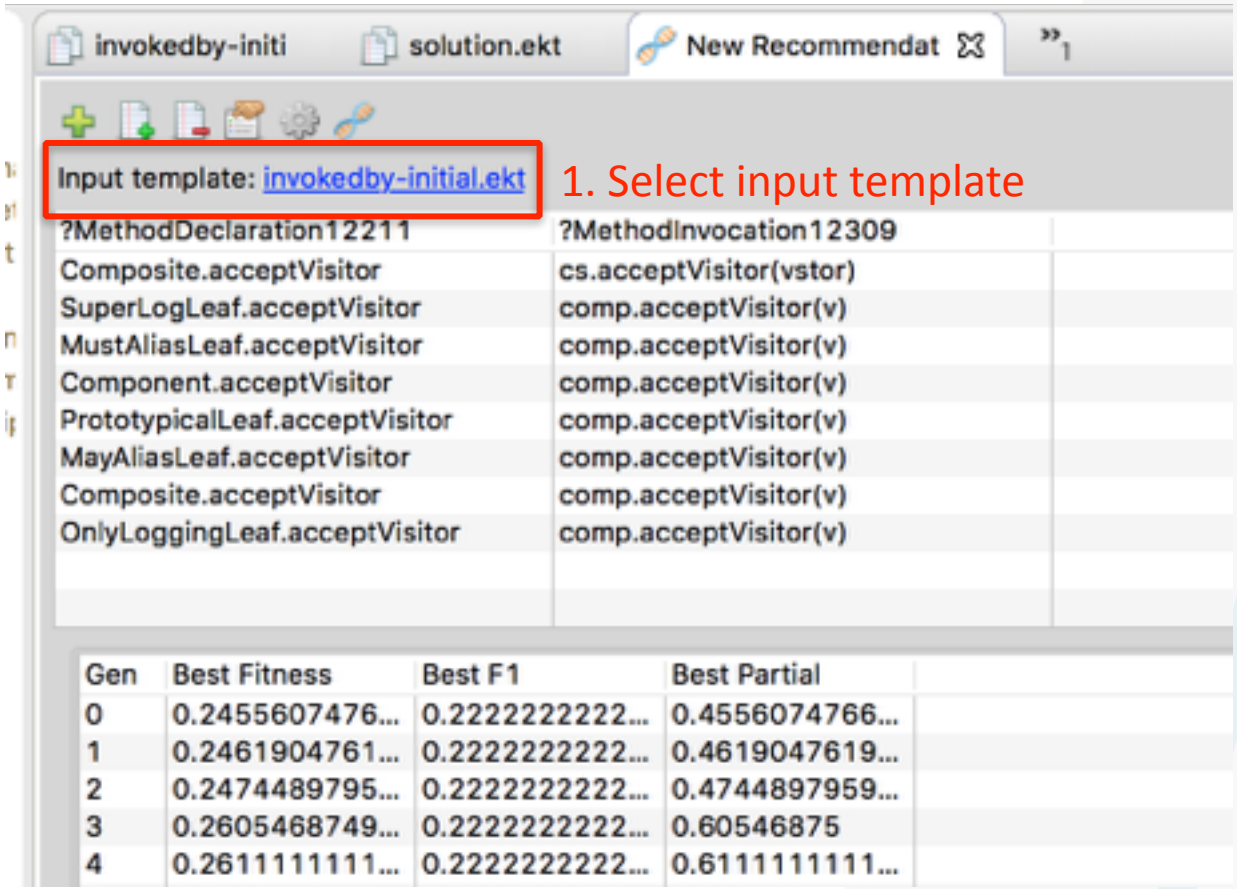
invokedby-initi | solution.ekt | New Recommendat

Input template: invokedby-initial.ekt

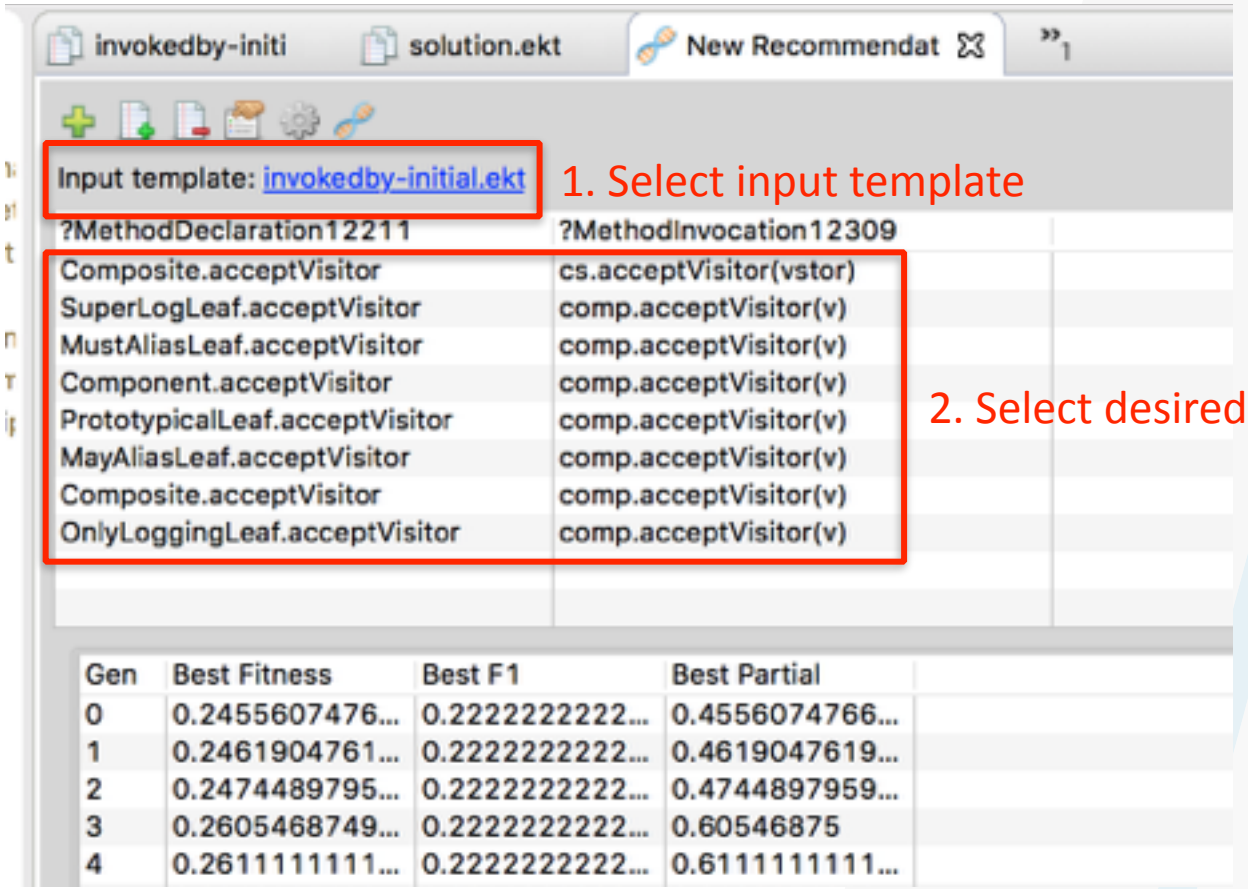
?MethodDeclaration12211	?MethodInvocation12309
Composite.acceptVisitor	cs.acceptVisitor(vstor)
SuperLogLeaf.acceptVisitor	comp.acceptVisitor(v)
MustAliasLeaf.acceptVisitor	comp.acceptVisitor(v)
Component.acceptVisitor	comp.acceptVisitor(v)
PrototypicalLeaf.acceptVisitor	comp.acceptVisitor(v)
MayAliasLeaf.acceptVisitor	comp.acceptVisitor(v)
Composite.acceptVisitor	comp.acceptVisitor(v)
OnlyLoggingLeaf.acceptVisitor	comp.acceptVisitor(v)

Gen	Best Fitness	Best F1	Best Partial
0	0.2455607476...	0.2222222222...	0.4556074766...
1	0.2461904761...	0.2222222222...	0.4619047619...
2	0.2474489795...	0.2222222222...	0.4744897959...
3	0.2605468749...	0.2222222222...	0.60546875
4	0.2611111111...	0.2222222222...	0.6111111111...

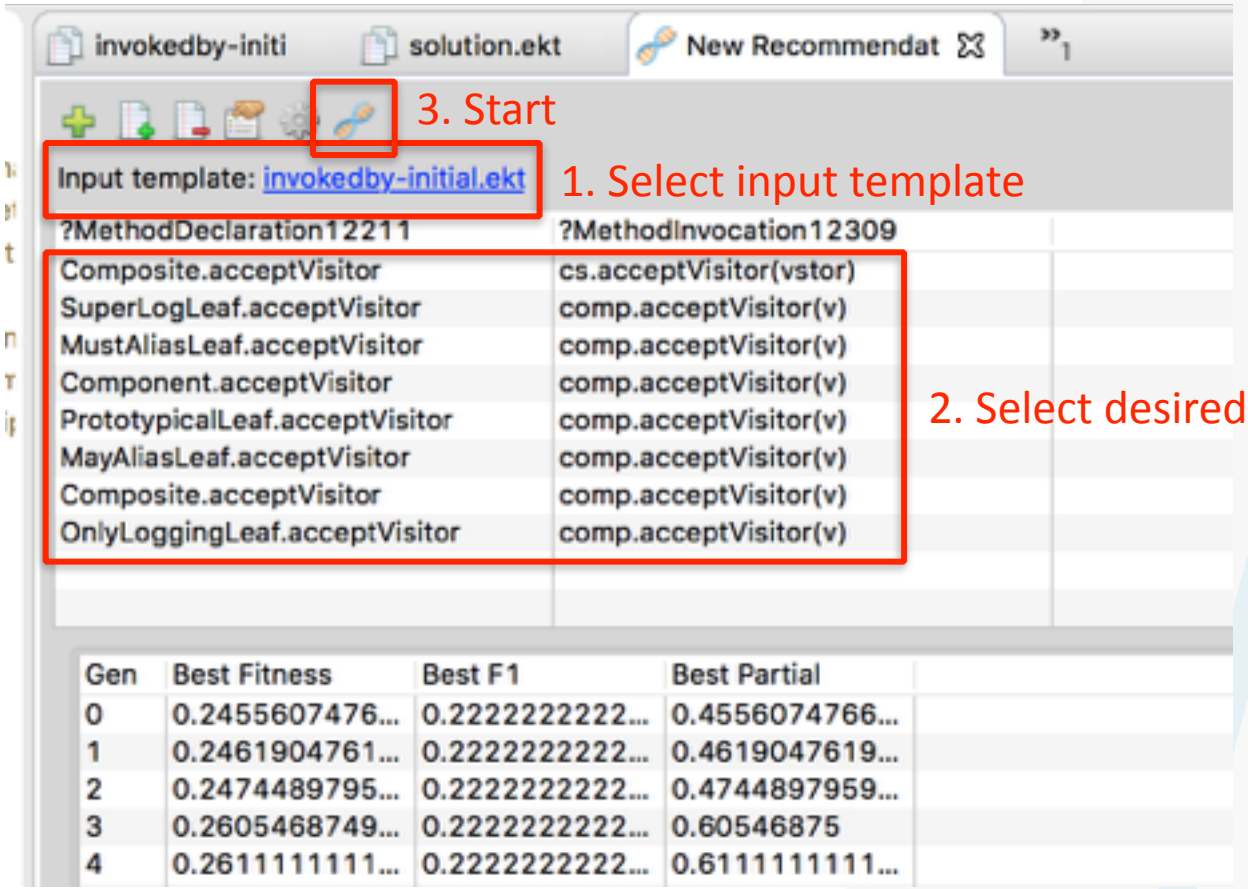
Recommender system



Recommender system



Recommender system



Demo

Summary

- Recommender system to assist Cha-Qeko/X users
- Future work:
  - Generalize instances of a systematic change
  - Migrate e.g. a bug fix between different versions of the same software system

Automated Generalization and Refinement of Source Code Templates Using Mutation Operators
Tim Molderez and Coen De Roover
23rd IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER'16), Tool track