PARSEWeb: A Programmer Assistant for Reusing Open Source Code on the Web

Suresh Thummalapenta and Tao Xie

Department of Computer Science

North Carolina State University

Raleigh, USA

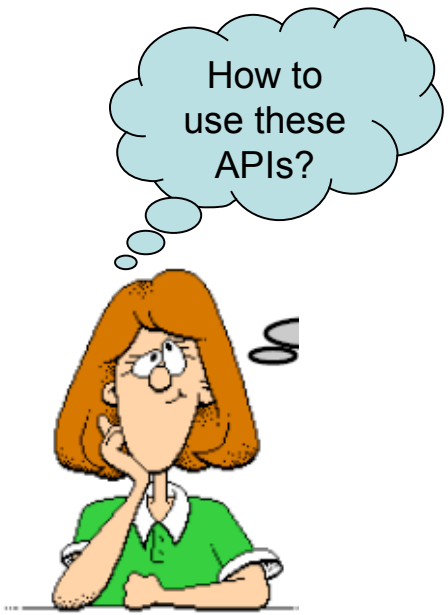
Automated Software Engineering 2007

Motivation

- Programmers commonly reuse APIs of existing frameworks or libraries

  – Advantages: Low cost and high efficiency of development
  – Challenges: Complexity and lack of documentation

Problem



- While reusing APIs of existing open source frameworks or libraries, programmers often
  - know what type of object they need
  - but do not know how to write code for getting that object

Query: “Source →Destination”

Example Task from Eclipse Programming

- Task: How to parse code in a dirty editor?
- Query: IEditorPart -> ICompilationUnit

- PARSEWeb Solution:

```
IEditorPart iep = ...
IEditorInput editorInp = iep.getEditorInput();
IWorkingCopyManager wcm = JavaUI.getWorkingCopyManager();
ICompilationUnit icu = wcm.getWorkingCopy(editorInp);
```

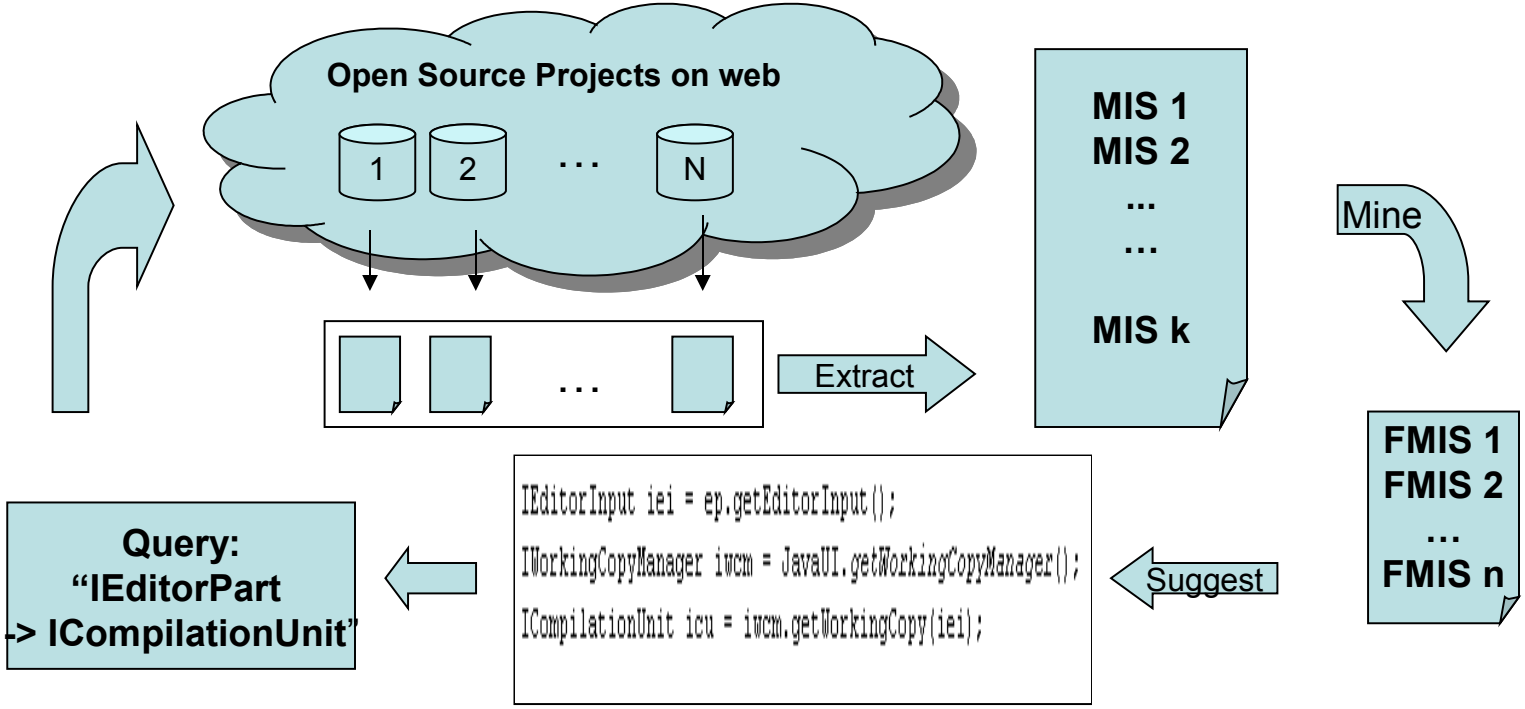
- Difficulties:

  a. Needs an instance of *IWorkingCopyManager*

  b. Needs to invoke a static method of *JavaUI* for getting the preceding instance

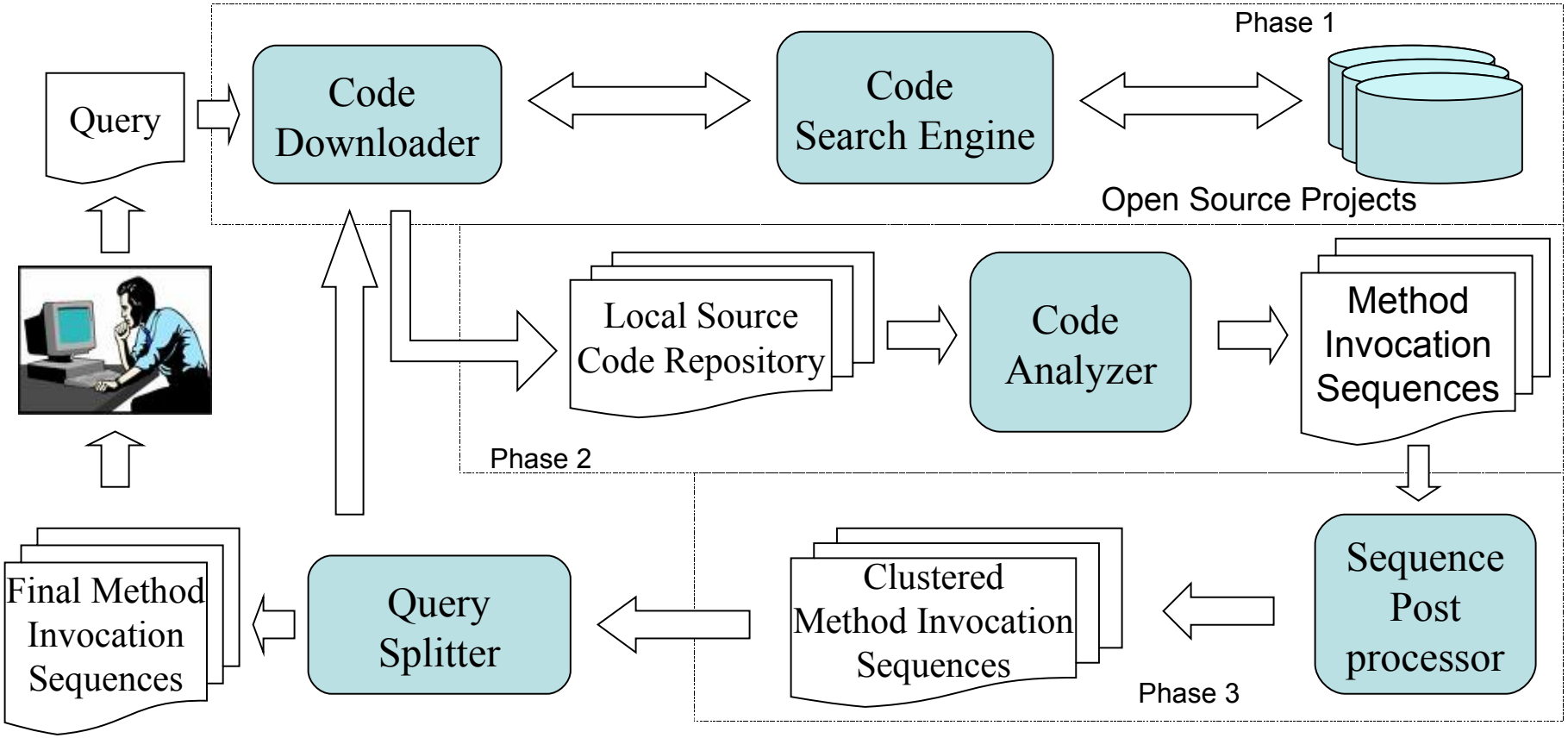
Example Task: High-level Data flow

Task: How to parse code in a dirty editor of Eclipse?

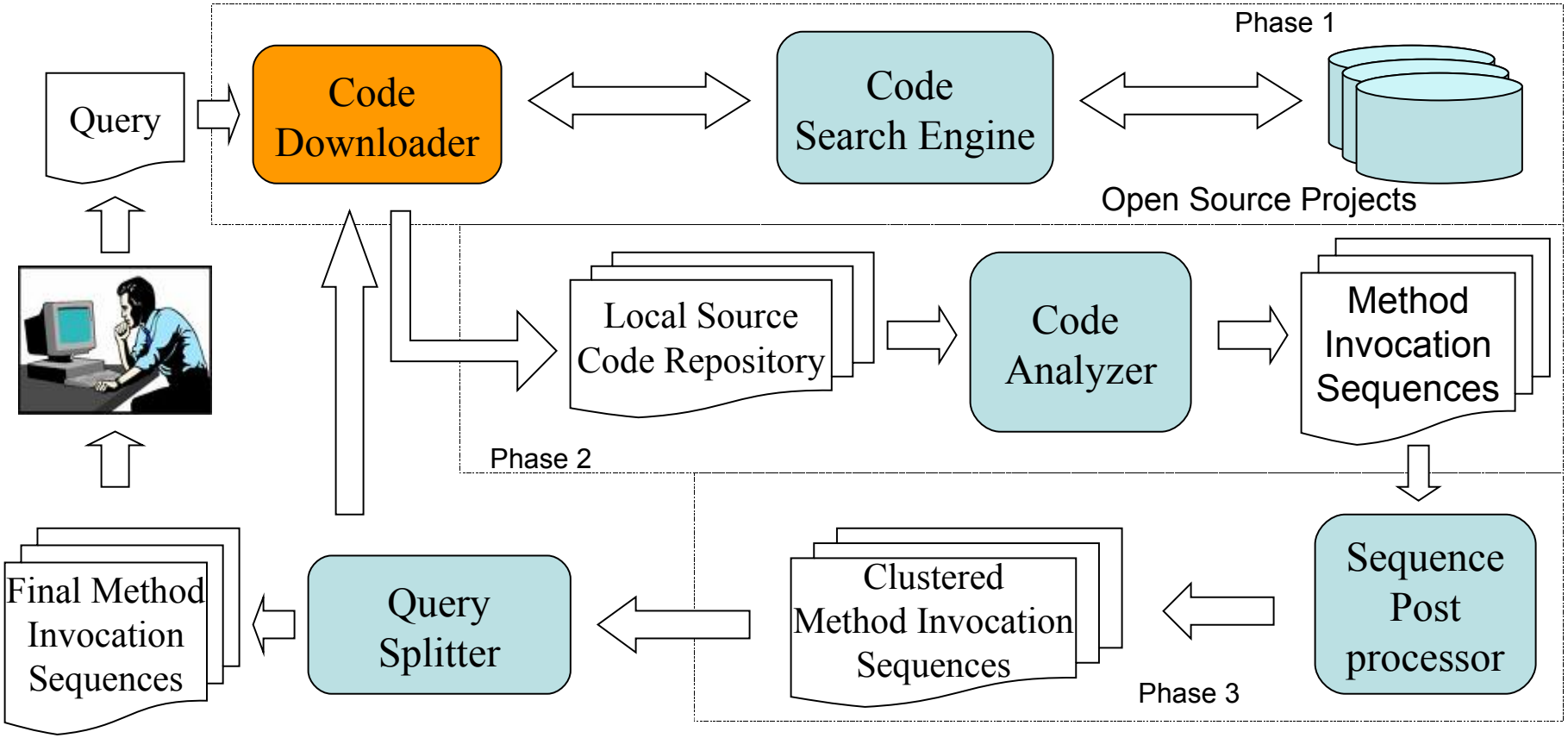


*MIS: Method-Invocation sequence, FMIS: Frequent MIS

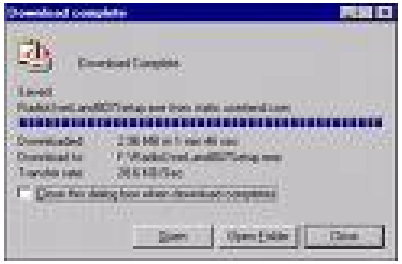
PARSEWeb Overview



PARSEWeb : Code Downloader

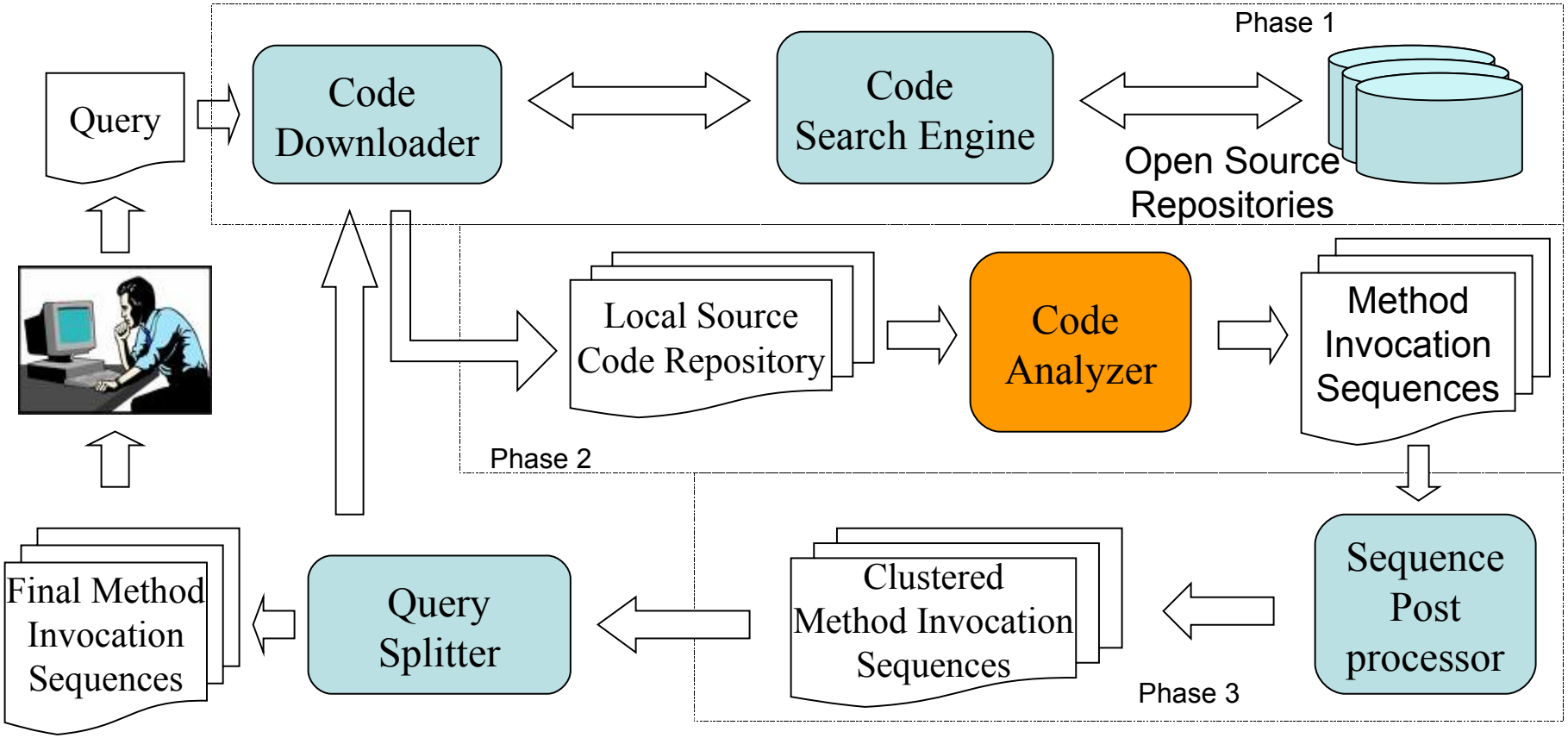


Code Downloader

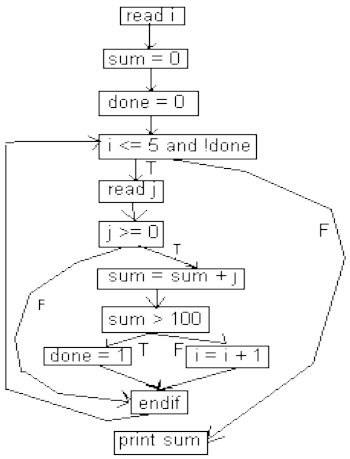


- Accepts queries of form [Source → Destination]
- Interacts with a code search engine to gather related code samples
- Stores gathered code samples (source files) in a repository, referred as “Local Source Code Repository”

PARSEWeb: Code Analyzer



Code Analyzer



- Collect [Source → Destination] method sequences by analyzing code samples statically
  - Deal with local method calls by inlining methods
  - Deal with conditionals/loops by traversing control flow graphs
- Resolve types in code samples
  - Challenges: Gathered code samples are not compilable
  - Solutions: heuristics are developed

Type Heuristics

- How do I get the return type of createQueueSession method?

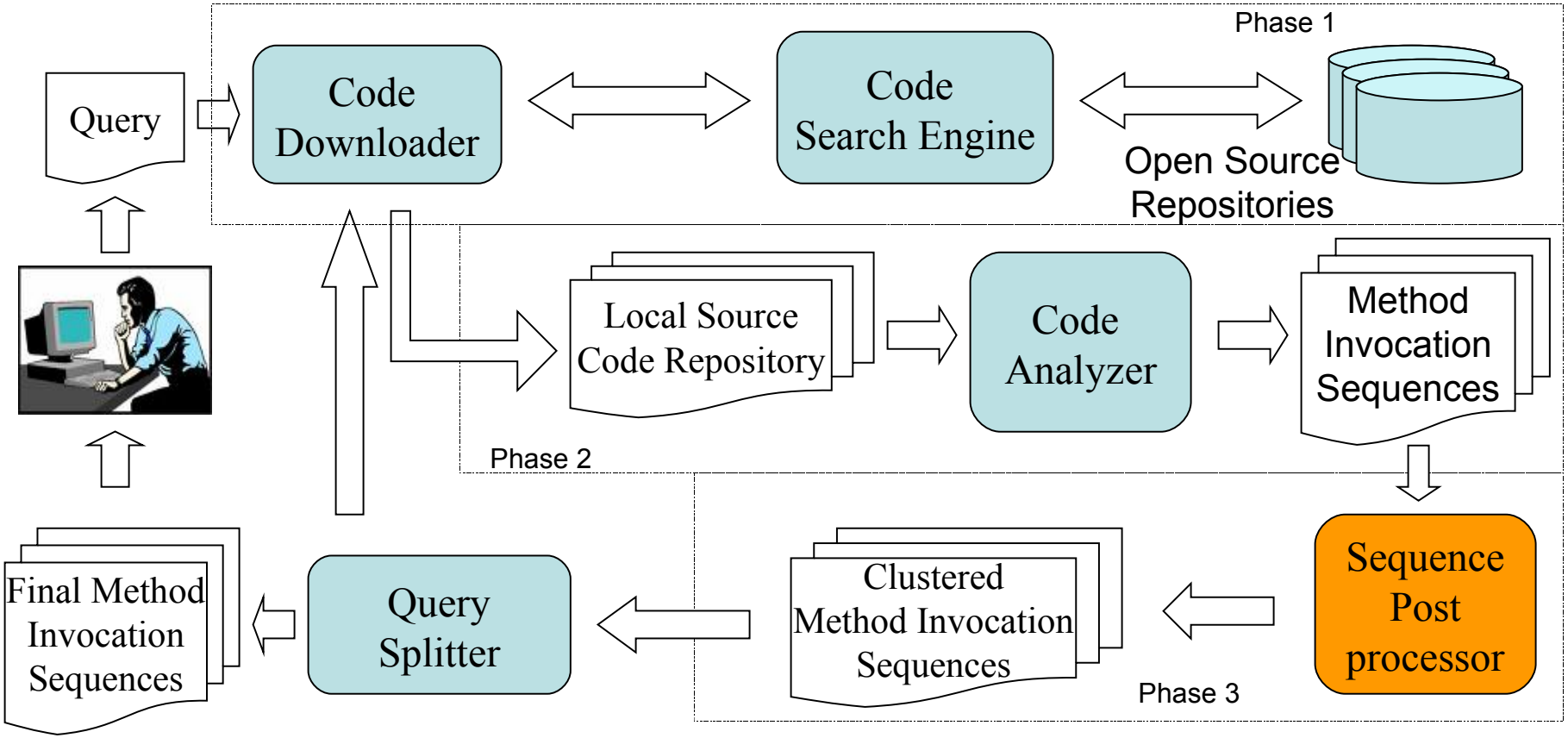
- Example 1:

```
QueueConnection connect;
QueueSession session = connect.createQueueSession(false,int)
```

- Example 2:

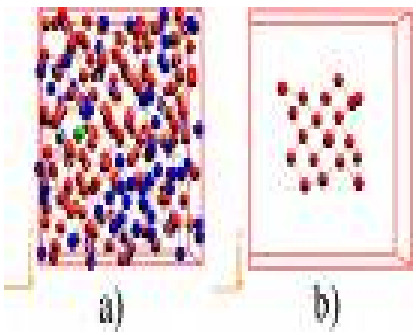
```
public QueueSession test()
{       ...
        return connect.createQueueSession(false,int);
}
```

PARSEWeb: Sequence Postprocessor



Sequence Postprocessor

- Candidate sequences produced by the code analyzer may be too many



Solutions:

- Cluster similar sequences
  - Clustering heuristics are developed
- Rank sequences
  - Ranking heuristics are developed

Clustering Heuristics

- Method-invocation sequences with the same set of statements can be considered similar, although the statements are in different order.

  e.g., "2 3 4 5" and "2 4 3 5 "

- Method-invocation sequences with minor differences measured by an attribute cluster precision value can be considered similar.

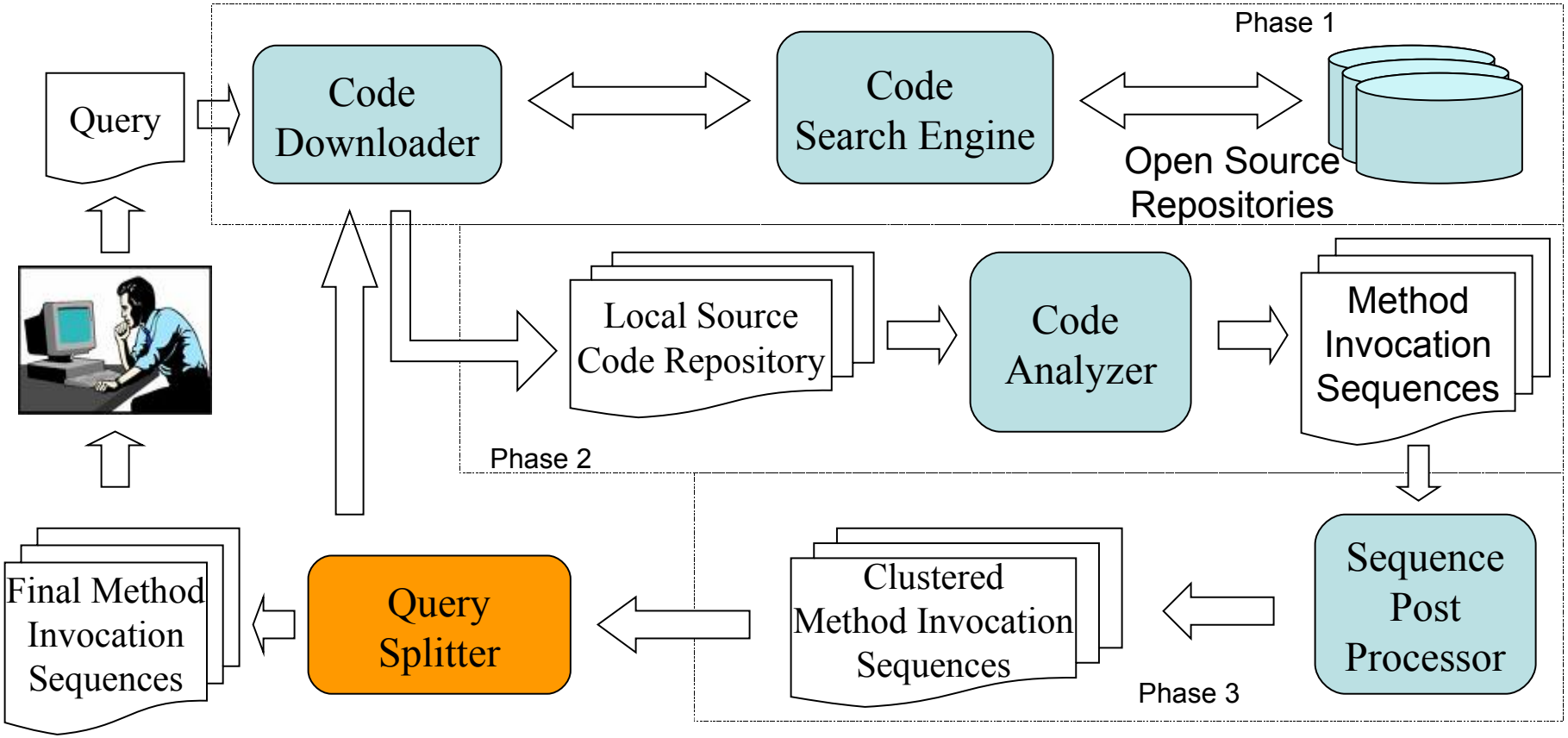
  e.g., "8 9 6 7" and "8 6 10 7 " can be considered similar under cluster precision value one

Ranking Heuristics

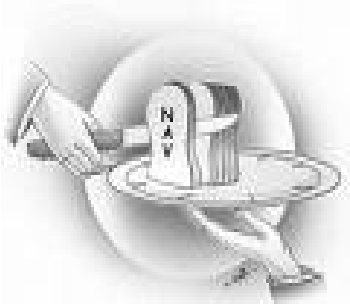
- Heuristic 1: *Higher frequency -> Higher rank*

- Heuristic 2: *Shorter length -> Higher rank*

PARSEWeb: Query Splitter



Query Splitter



- Lack of code samples that give candidate method-invocation sequences in the results of code search engines
  - Required method-invocation sequences are split among different source files
- Solution:
  - Split the user query into multiple queries
  - Compose the results for each split query

Query Splitting Example

1. User query: "org.eclipse.jface.viewers.IStructuredSelection->java.io.ObjectInputStream"

Results: None

2. Query: "java.io.ObjectInputStream"

Results: 3.

Most used immediate sources are: java.io.InputStream, java.io.ByteArrayInputStream, java.io.FileInputStream

3. Three Queries to be fired:

"org.eclipse.jface.viewers.IStructuredSelection-> java.io.InputStream" Results: 1

"org.eclipse.jface.viewers.IStructuredSelection-> java.io.ByteArrayInputStream" Results: 5

"org.eclipse.jface.viewers.IStructuredSelection-> java.io.FileInputStream" Results: None

Eclipse Plugin

0_JavaFileMerger.java

```
// delete Abator generated stuff, and collect imports
astParser.setSource(icu);
CompilationUnit cu = (CompilationUnit) astParser.createAST(null);
AST ast = cu.getAST();

ExistingJavaFileVisitor visitor = new ExistingJavaFileVisitor();

cu.recordModifications();
```

Problems | Javadoc | Declaration | PARSEWeb Search Results

org.eclipse.jdt.core.ICompilationUnit -> org.eclipse.jdt.core.dom.CompilationUnit

File Name and Method Sequences	Rank	Frequency	Length	Confidence	U...
File: 0_JavaFileMerger.java Method: getMergedSource	1	3	2	High	1
org.eclipse.jdt.core.dom.ASTParser,setSource(org.eclipse.jdt.core.ICompilationUnit) ReturnType:void					
org.eclipse.jdt.core.dom.ASTParser,createAST(null) DownCasted_ReturnType:org.eclipse.jdt.core.dom.CompilationUnit					
File: 5_BugResolution.java Method: runInternal	2	1	1	High	1
this,createWorkingCopy(org.eclipse.jdt.core.ICompilationUnit) ReturnType:org.eclipse.jdt.core.dom.CompilationUnit					

Evaluations

- Real Programming Problems: To address problems posted in developer forums

- Real Projects: To show that solutions recommended by PARSEWeb are
  - available in real projects
  - better than solutions recommended by related tools PROSPECTOR, Strathcona, and Google Code Search averagely

Real Programming Problems

Jakarta BCEL user forum, 2001

Problem: "How to disassemble java byte code"
Query: "Code → Instruction"

Solution Sequence:

FileName:2_RepMIStubGenerator.java MethodName: isWriteMethod Rank:1
NumberOfOccurrences:1
Code,getCode() ReturnType:#UNKNOWN#
CONSTRUCTOR,InstructionList(#UNKNOWN#) ReturnType:InstructionList
InstructionList,getInstructions() ReturnType:Instruction

Solution Sample Code:

```
Code code;
InstructionList il = new InstructionList(code.getCode());
Instruction[] ins = il.getInstructions();
```

Real Programming Problems

Dev 2 Dev Newsgroups, 2006

Problem: “how to connect db by sessionBean”

Query: javax.naming.InitialContext → java.sql.Connection

Solution Sequence:

FileName:3 AddressBean.java MethodName:getNextUniqueKey Rank:1
NumberOfOccurrences:34
javax.naming.InitialContext,lookup(java.lang.String)
ReturnType:javax.sql.DataSource
javax.sql.DataSource,getConnection()
ReturnType:java.sql.Connection

Real Project: Logic

- Source File: LogicEditor.java

Query		PARSE		PROS		Strath		GCSE
Source	Destination	No	Ra	No	Ra	No	Ra	
IPageSite	IActionBars	1	1	3	1	10	7	1
ActionRegistry	IAction	3	1	4	1	10	3	2
ActionRegistry	ContextMenu Provider	Nil	Nil	2	2	10	3	NA
IPageSite	ISelection Provider	1	1	12	1	10	Nil	5
IPageSite	IToolBar Manager	2	1	12	1	10	6	9
String	ImageDescriptor	10	6	12	Nil	10	Nil	28
Composite	Control	10	2	12	Nil	10	Nil	72
Composite	Canvas	10	5	12	Nil	10	Nil	28
GraphicalViewer Thumbnail	Scrollable	2	1	12	8	10	7	2
GraphicalViewer	IFigure	1	Nil	12	Nil	10	Nil	NA

PARSE: PARSEWeb, PROS: Prospector, Strath: Strathcona
No: Number, Ra: Rank

SUMMARY-> PARSEWeb: 8/10, Prospector: 6/10, Strathcona: 5/10

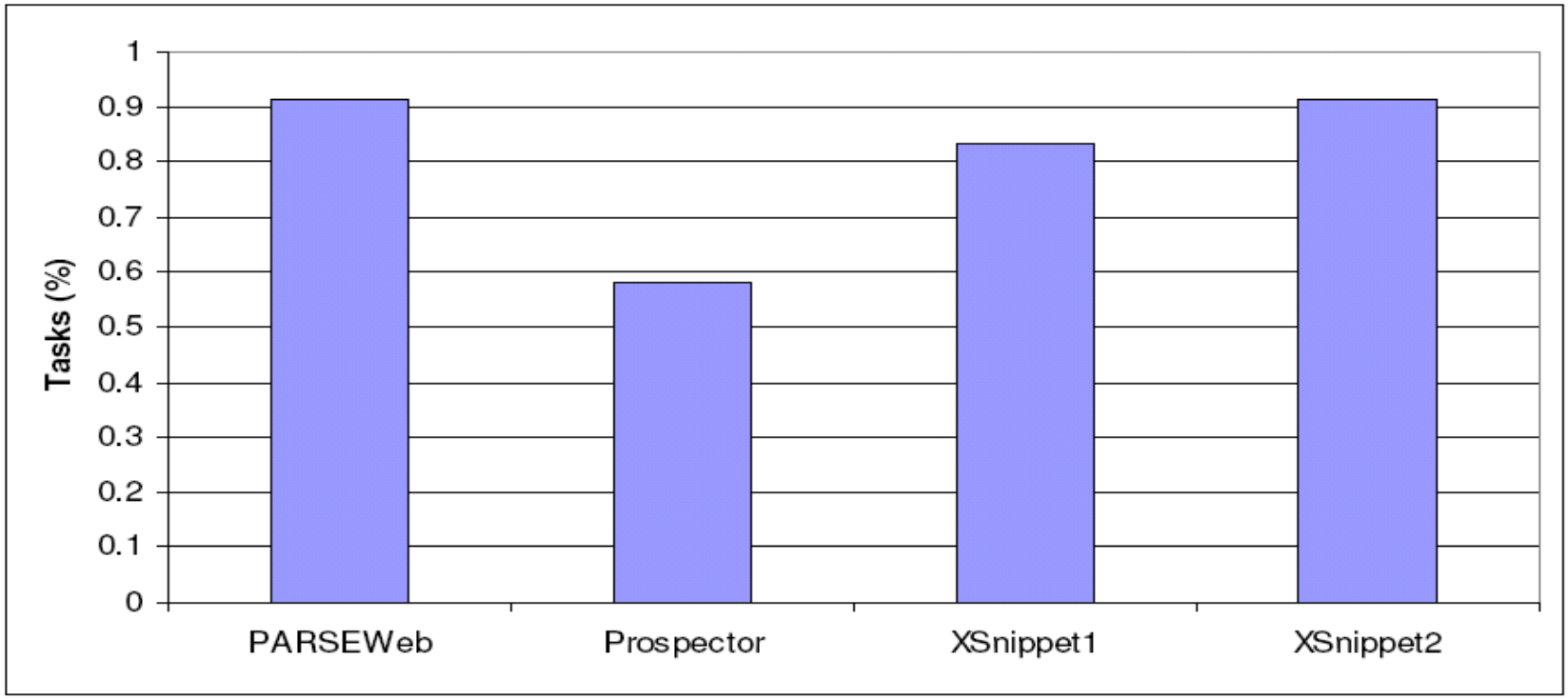
Comparison with Prospector

- 12 specific programming tasks taken from XSnippet approach.

Query		PARSE	PROS
Source	Destination		
ISelection	ICompilationUnit	Yes	No
IStructuredSelection	ICompilationUnit	Yes	Yes
ElementChangedEvent	ICompilationUnit	Yes	Yes
IEditorPart	ICompilationUnit	Yes	Yes
IEditorPart	IEditorInput	Yes	Yes
ViewPart	ISelectionService	Yes	Yes
TextEditorAction	ITextEditor	Yes	No
TextEditorAction	ITextSelection	Yes	No
ITextEditor	ITextSelection	Yes	Yes
AbstractDecorated TextEditor	ProjectViewer	No	No
TextEditor	IDocument	Yes	No
TextEditor	ITextSelection	Yes	Yes

SUMMARY-> PARSEWeb: 11/12, Prospector: 7/12

Comparison with Other Tools



Percentage of tasks successfully completed by PARSEWeb, Prospector, and XSnippet

Significance of Internal Techniques

Query		Simple	Method Inline	Post Process	Query Split
Source	Destination				
TableViewer	TableColumn	21	23	2	2
IWorkbench	IEditorPart	13	17	8	8
IWorkBench Page	IStructured Selection	5	6	1	1
Composite	Control	26	29	24	24
IEditorSite	ISelectionService	Nil	Nil	Nil	2

*Legend:
Method inline: Method inlining
Post Process: Sequence Post Processor
Query Split: Query Splitter

Related Work

- Jungloid mining: Helping to navigate the API jungle [Mandelin et al. PLDI 05]
- Using structural context to recommend source code examples [Holmes and Murphy ICSE 05]
- XSnippet: Mining for sample code [Sahavechaphan and Claypool OOPSLA 06]
- MAPO: Mining API usages from open source repositories [Xie and Pei MSR 06]

Future Work

- Interacts with a single code search engine, i.e., Google code search
  - Plan to interact with other code search engines
- Requires both Source and Destination objects
  - Plan to infer Source from the code context
- Type heuristics cannot infer types in a few scenarios

  Eg:

```
QueueSession session = factory.createQueueConnection().
                                createQueueSession(false, Session.AUTO);
```

→ Cannot infer the *return* type of createQueueConnection and the *receiver* type of createQueueSession

Conclusion

- An approach that tries to address problems faced by programmers in reusing APIs of existing frameworks or libraries by accepting queries of the form

  "Source → Destination "

  and by suggesting frequent method-invocation sequences as solutions

- Addressed two problems posted in developer forums and compared with existing related tools Prospector and Strathcona

Questions?