

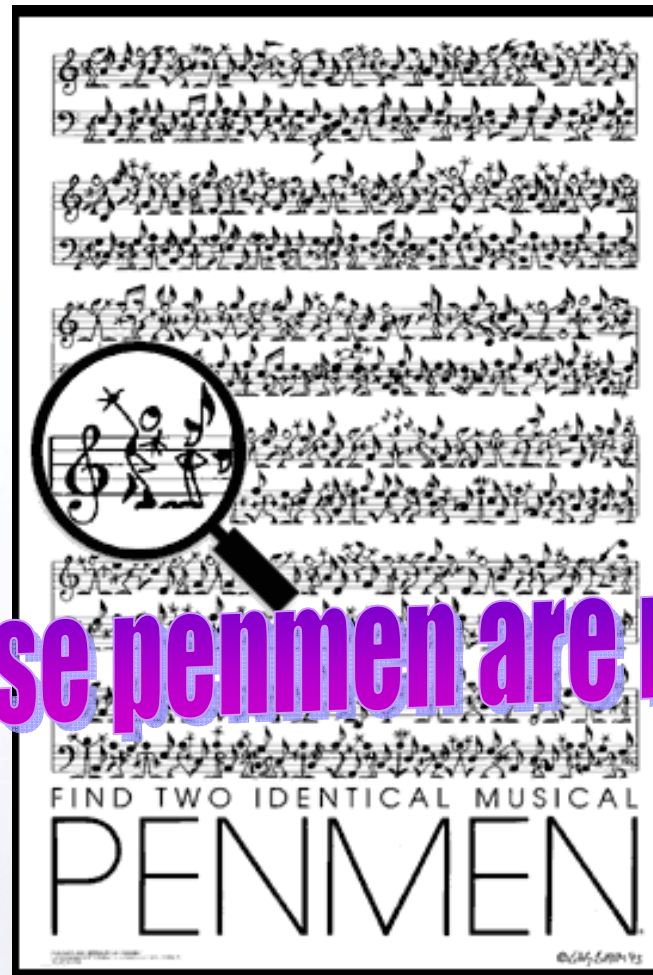
HTTP://WWW...



# Detecting Similar Software Applications

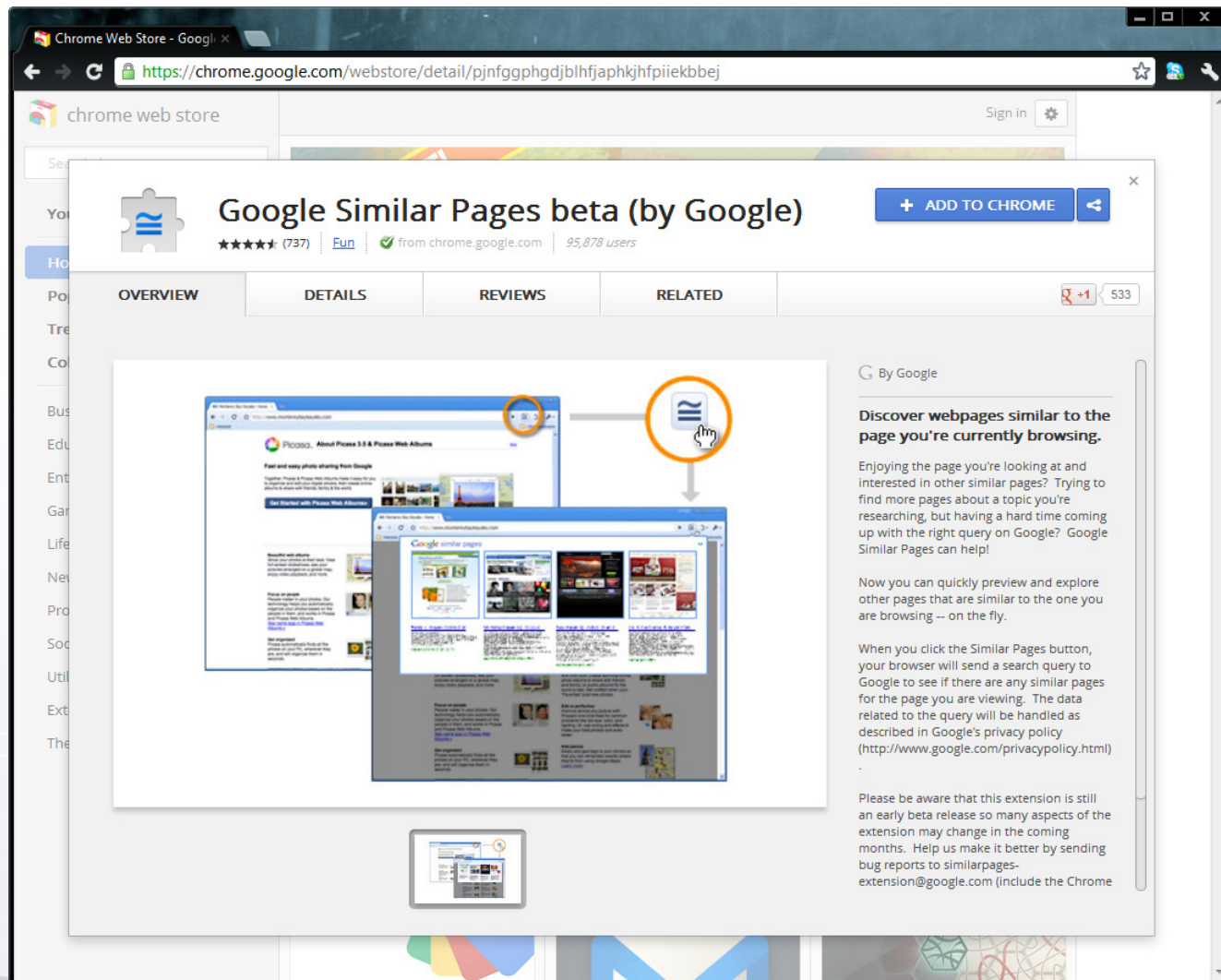
Collin McMillan, Mark Grechanik, and Denys Poshyvanyk  
The College of William and Mary and  
The University of Illinois at Chicago

# Find Identical Penmen



What if these penmen are not identical?

# Finding Similar Web Pages





# Similar Web Pages For ACM SigSoft

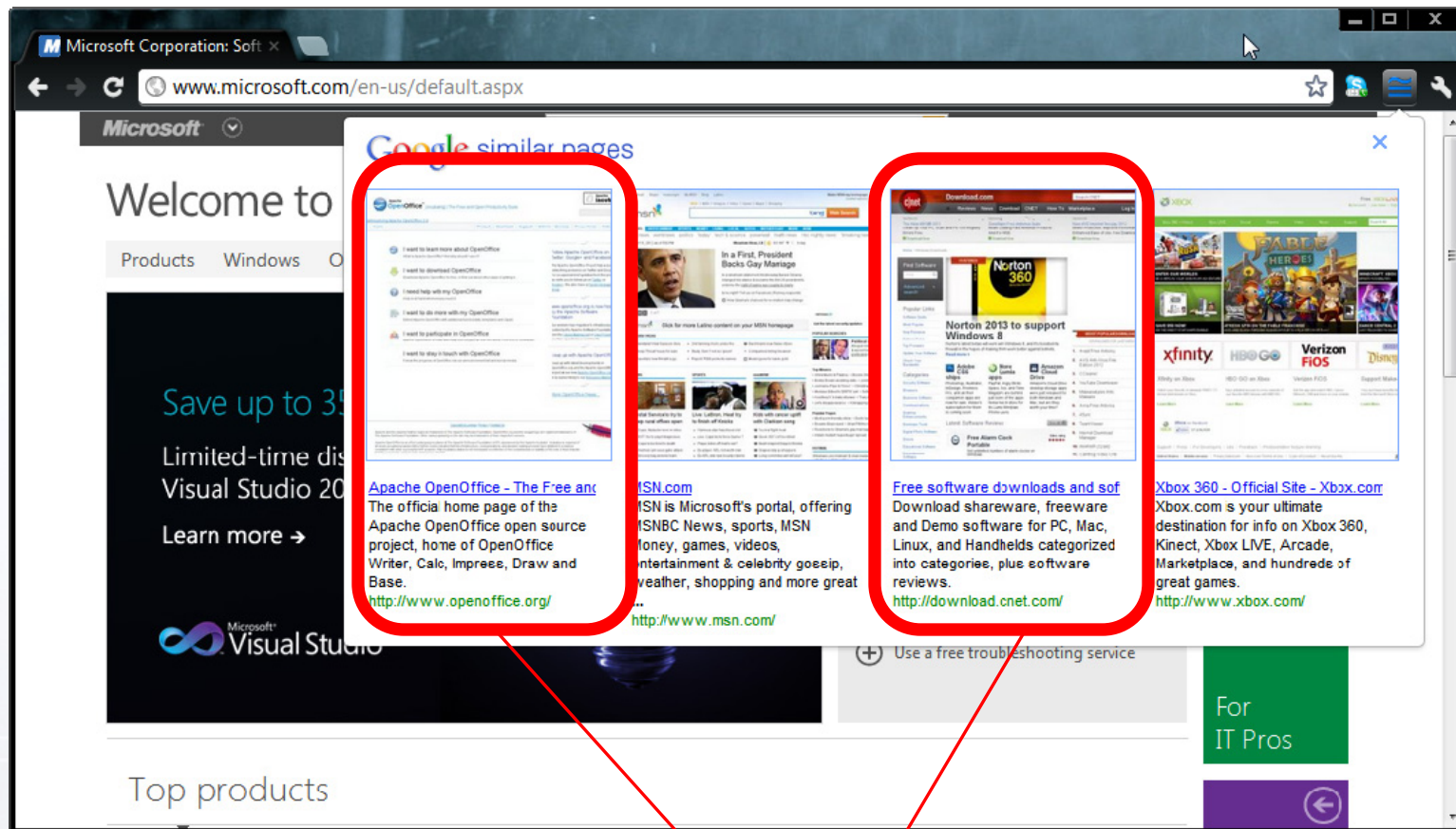
The screenshot shows a web browser window with the address bar displaying "www.sigsoft.org". The main content area of the browser shows the SIGSOFT Homepage. On the left side of the page, there is a navigation menu with links: Home, About Us, Publications, Highlights, Awards, Conferences & Workshops, The Impact Project, Dissertations, Resources, Software Eng. Notes - SEN. The main content area features a large banner with the text "SIGSOFT SPECIAL INTEREST GROUP" and a "Join Us" button. Below this, there is a section titled "SIGSOFT sponsors some very popular events in the area of Software Engineering. An up to date list of those events can be found [here](#). Consult the [ACM Calendar of Events](#) for more information about dates and deadlines. More information for event organizers that are interested in cooperating with SIGSOFT can be found [here](#)."

Overlaid on the right side of the browser window is a "Google similar pages" panel. This panel displays four similar web pages with their respective thumbnails and titles:

- IEEE Computer Society - Premier Of**  
The IEEE Computer Society serves computing professionals' information needs with technical publications, conferences, books, proceedings, certifications, and ...  
<http://www.computer.org/>
- Software Engineering Institute SEI Home Page.**  
<http://www.sei.cmu.edu/>
- ESEC/FSE 2009**  
It is our great pleasure to welcome you to ESEC/FSE'2009 in Amsterdam. This year's conference continues its tradition of being a premier forum for researchers, ...  
<http://www.esec-fse-2009.ewi.tuc>
- Welcome — Association for Computing Machinery**  
ACM, the world's largest educational and scientific computing society, delivers resources that advance computing as a science and a profession. ACM provides ...  
<http://www.acm.org/>

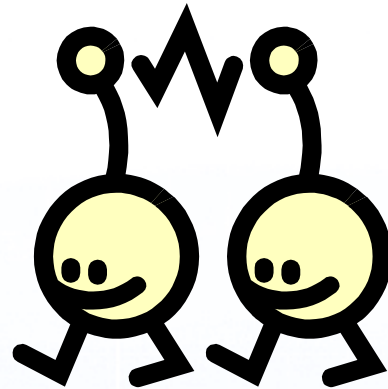
At the bottom of the similar pages panel, there is a section titled "Check out SIGSOFT on" with social media links for Facebook, Twitter, and LinkedIn.

# Similar Web Pages For Microsoft



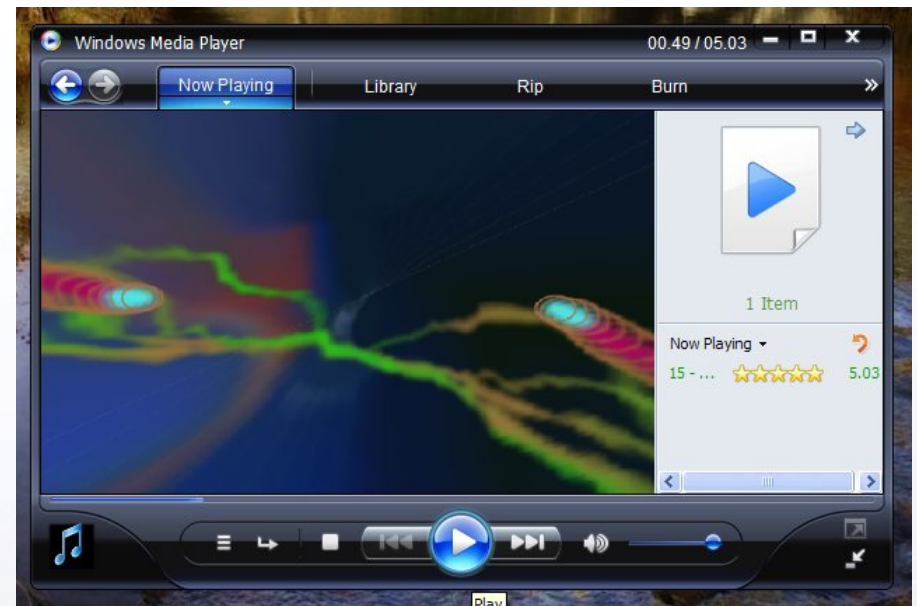
Open-source free software!

# Similar Applications



**Software applications are similar if they implement related semantic requirements**

# Example: RealPlayer and Windows Player





# Why Detect Similar Applications?

Given an entire application,  
detect what other applications  
are similar to it so that we can

- assess reusability of these applications;
- improve understanding of source code;
- rapidly prototype new applications, and
- discover code theft and plagiarism.

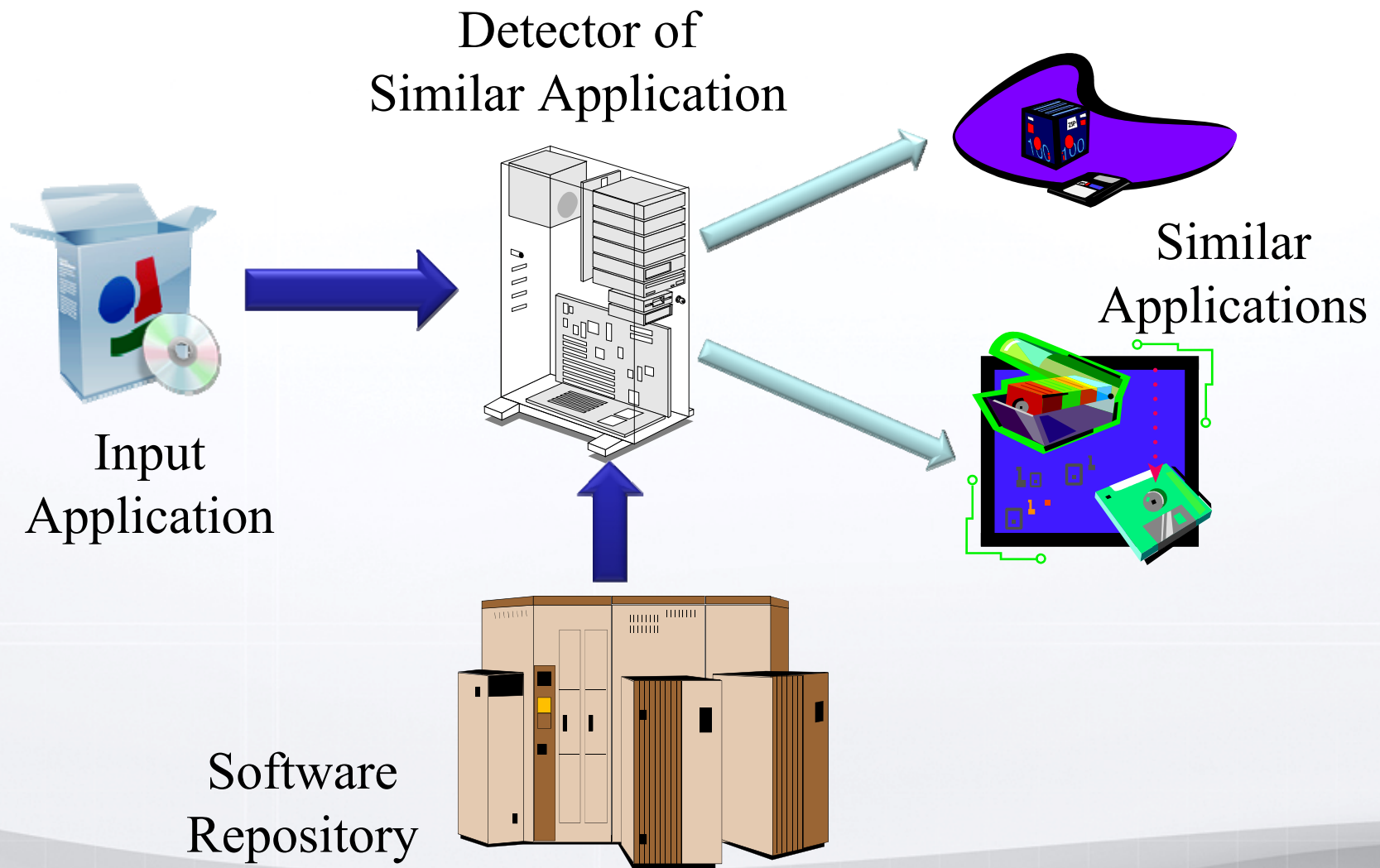


# Economic Importance of Detecting Similar Applications

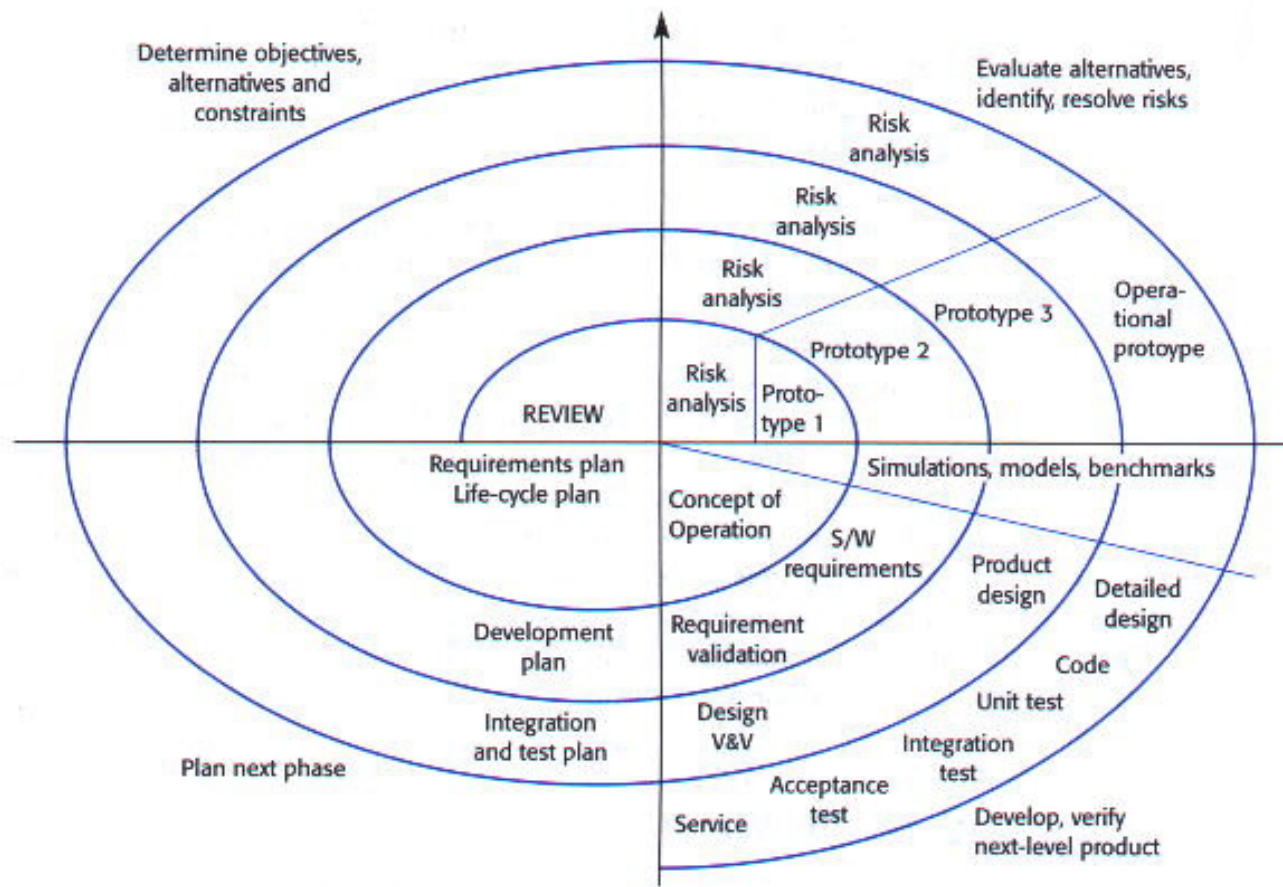
- Consulting companies accumulated tens of thousands of software applications in their repositories that they have built for the past 50 years!
- These applications constitute a knowledge treasure for reusing it from successfully delivered applications in the past.
- Detecting similar applications and reusing their components will save time and resources and increase chances of winning future bids.



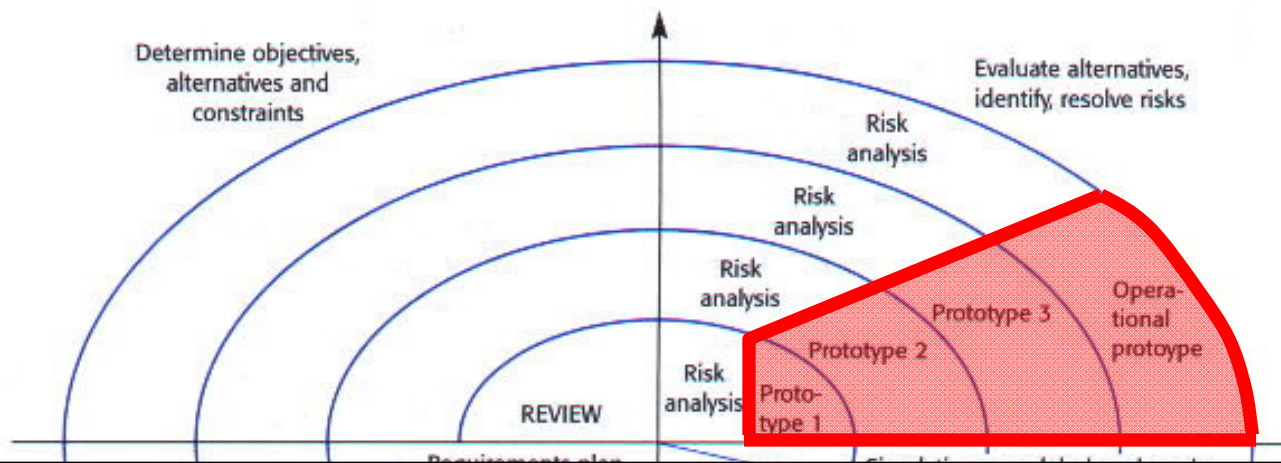
# An Overview of the Process



# Spiral Model, Bidding, and Prototyping



# Spiral Model, Bidding, and Prototyping



Building prototypes repeatedly from scratch is expensive since these prototypes are often discarded after receiving feedback from stakeholders

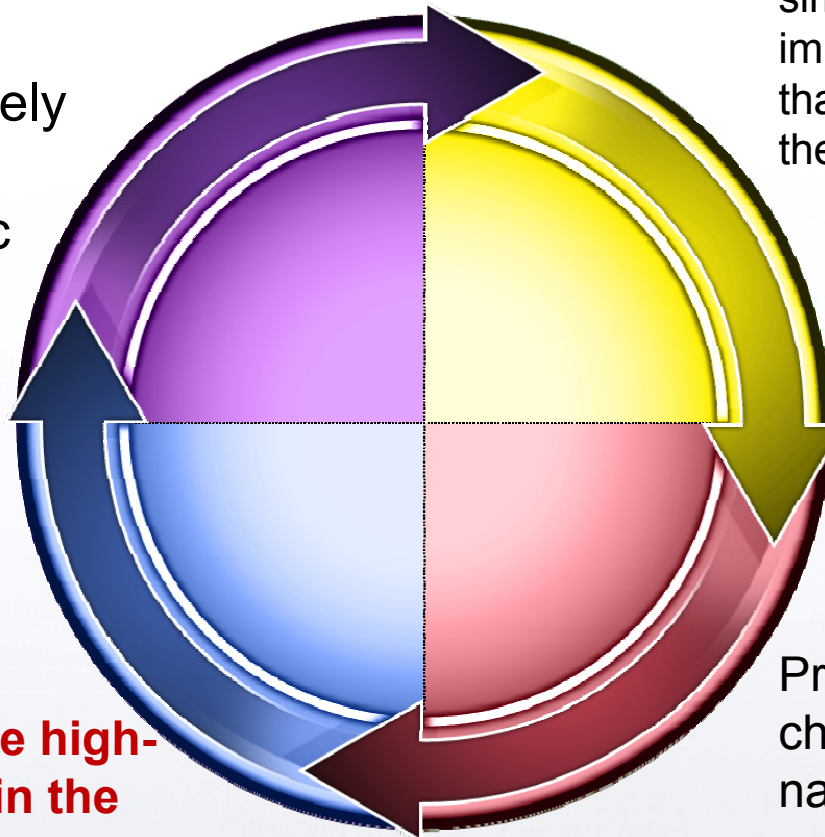


Since prototypes are *approximations* of desired resulting applications, similar applications from software repositories can serve as *prototypes* because they are relevant to your requirements

# Problem

Detecting similar applications in a timely manner can lead to significant economic benefits.

Two applications are similar to each other if they implement some features that are described by the same abstraction.



Programmers rarely choose meaningful names that reflect correctly the concepts or abstractions that they implement

**Mismatch between the high-level intent reflected in the descriptions of these applications and low-level implementation details.**

# Detecting Similar Applications Is Very Difficult



**Currently, detecting similar applications is like looking for a needle in a stack of hay!**

# Mizzaro's Conceptual Framework

Similar documents are relevant to one another if they share some common concepts.

- Once these concepts are known, a corpus of documents can be clustered by how documents are relevant to these concepts.

Semantic anchors are elements of documents that precisely define the documents' semantic characteristics.

- Without semantic anchors, documents are considered as bags of words with no semantics.
- In *syntagmatic associations*, documents are considered similar when terms (i.e., words) in these documents occur together.



# Our Hypothesis

In paradigmatic associations, documents are considered similar if they contain terms with high semantic similarities.

- These terms can be API calls that comes from well-known libraries.

Using paradigmatic associations it is possible to compute similarities between documents with a higher degree of accuracy when compared to documents that have syntagmatic associations (only similar words).

- We deal with the problem that different programmers can use the same words to describe different requirements (i.e., the synonymy problem) and they can use different words to describe the same requirements (i.e., the polysemy problem).

# Closely reLated ApplicationNs

Find proper weights for  
these semantic anchors

Detect co-occurrences of  
semantic anchors that  
form patterns of  
implementing  
different requirements.



Find reliable semantic  
anchors

How to do that?

# Closely reLated ApplicationNs (CLAN) – [www.javaclan.net](http://www.javaclan.net)

Find proper weights for  
these semantic anchors

Detect co-occurrences of  
semantic anchors that  
form patterns of  
implementing  
different requirements.



Find reliable semantic  
anchors

# CLAN!!!

# Latent Semantic Analysis (LSA)

LSA reduces the dimensionality of the similarity space while simultaneously revealing latent concepts that are implemented in the underlying corpus of documents.

- The JDK contains close to 115,000 API calls that are exported by a little more than 13,000 classes and interfaces that are contained in 721 packages.

In LSA, terms are elevated to an abstract space, and terms that are used in similar contexts are considered similar even if they are spelled differently

- LSI automatically makes embedded concepts explicit using Singular Value Decomposition (SVD), which is a form of factor analysis used to reduce dimensionality of the space to capture most essential semantic information.

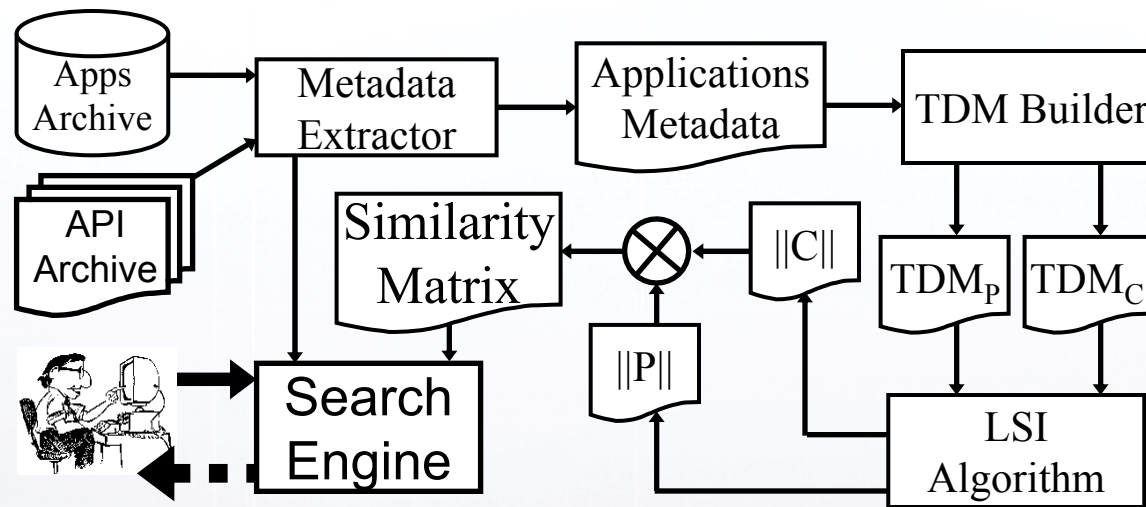


# Latent Semantic Analysis (LSA)



$$\begin{array}{|c|c|} \hline & \text{document} \\ \hline \text{term} & m \times n \\ \hline \end{array} = \begin{array}{|c|c|} \hline & \text{dims} \\ \hline \text{term} & m \times r \\ \hline \end{array} \times \begin{array}{|c|c|} \hline \text{dims} & \text{dims} \\ \hline \text{dims} & r \times r \\ \hline \end{array} \times \begin{array}{|c|c|} \hline & \text{document} \\ \hline \text{dims} & r \times n \\ \hline \end{array}$$

# The Architecture of CLAN



# CLAN UI

**Both Applications**

**Only MidiQuickFix**

**Only mbox**

- com::sun::media::sound
  - com::sun::media::sound::MixerSequencer
  - com::sun::media::sound::MixerSequencer::RecordingTrack
  - com::sun::media::sound::AbstractMidiDevice
    - doClose()
  - com::sun::media::sound::FastSysexMessage
- com::sun::org::apache::xml::internal::utils
- java::math
- javax::sound::midi
- javax::swing

# Empirical Evaluation

- Goal: to compare **CLAN** with **MUDABlue** and **Combined**
- MUDABlue [Kawaguchi'06] provides automatic categorization of applications using underlying words in source code
- Implemented a feature of MUDABlue for computing similarities among apps using ALL IDENTIFIERS from source code
- Combined CLAN + MUDABlue = Combined (Words + APIs)
- Instantiated CLAN, MUDABlue and Combined on the same repository of 8,310 Java applications



# Empirical Evaluation

- A user study with 33 Java student programmers from the University of Illinois in Chicago
  - 21 graduate students;
  - 12 upper-level undergraduate students;
  - 15 participants reported between 1-3 years of Java programming experience
  - 11 participants reported more than 3 years of Java programming experience
  - 16 participants reported prior experience with search engines
  - 8 reported that they never used code search engines

# Cross-Validation Design

Experiment	Group	Approach	Task Set
1	A	CLAN	T1
	B	MUDABlue	T2
	C	Combined	T3
2	A	Combined	T2
	B	CLAN	T3
	C	MUDABlue	T1
3	A	MUDABlue	T3
	B	Combined	T1
	C	CLAN	T2

## Large Case Studies are Rare

“First, it is very difficult to scale human experiments to get quantitative, significant measures of usefulness; this type of large-scale human study is very rare.

Second, comparing different recommenders using human evaluators would involve carefully designed, time-consuming experiments; this is also extremely rare.”

Saul, Filkov, Devanbu, Bird  
Recommending Random Walks, ESEC/FSE'07



# Participants' Role

- 1) Receive Task and search for Apps using the Search Engine

**Recording music data into a MIDI file**

- 2) Translate Task to Query, Enter into Search Engine

Don't know the name of your application? Try describing it!

- 3) Identify the relevant **source** App
- 4) Find **target** applications using a similarity Engine

**(CLASS/PACKAGE VERSION)**

Enter the name of any Java application that you find in Sourceforge, for example, *midiquickfix*, *pimnet*, *gateway*, or *firebird*.



# Likert Scale - Confidence



1) Completely irrelevant – there is absolutely nothing that the participant can use from this retrieved code fragments, nothing in it is related to keywords that the participant chose based on the descriptions of the tasks.



2) Mostly irrelevant – a retrieved code fragment is only remotely relevant to a given task; it is unclear how to reuse it.



3) Mostly relevant – a retrieved code fragment is relevant to a given task and participant can understand with some modest effort how to reuse it to solve a given task.



4) Highly relevant – The participant is highly confident that code fragment can be reused and s/he clearly see how to use it.

# Analysis of the Results

Metrics:

Confidence (C)

Precision (P)

Similarity Engine	Apps Entered	Apps Rated
CLAN	33	304
MUDABlue	33	322
Combined	33	322

# Hypotheses

**Null hypothesis ( $H_0$ ):** There is no difference in the values of confidence level and precision per task between participants who use MUDABlue, Combined, and CLAN.

**Alternative hypothesis ( $H_1$ ):** There is statistically significant difference in the values of confidence level and precision between participants who use MUDABlue, Combined, and CLAN.

# Hypotheses Tested

$H_1$ : Confidence of CLAN vs. MUDABlue

$H_2$ : Precision of CLAN vs. MUDABlue

$H_3$ : Confidence of CLAN vs. Combined

$H_4$ : Precision of CLAN vs. Combined

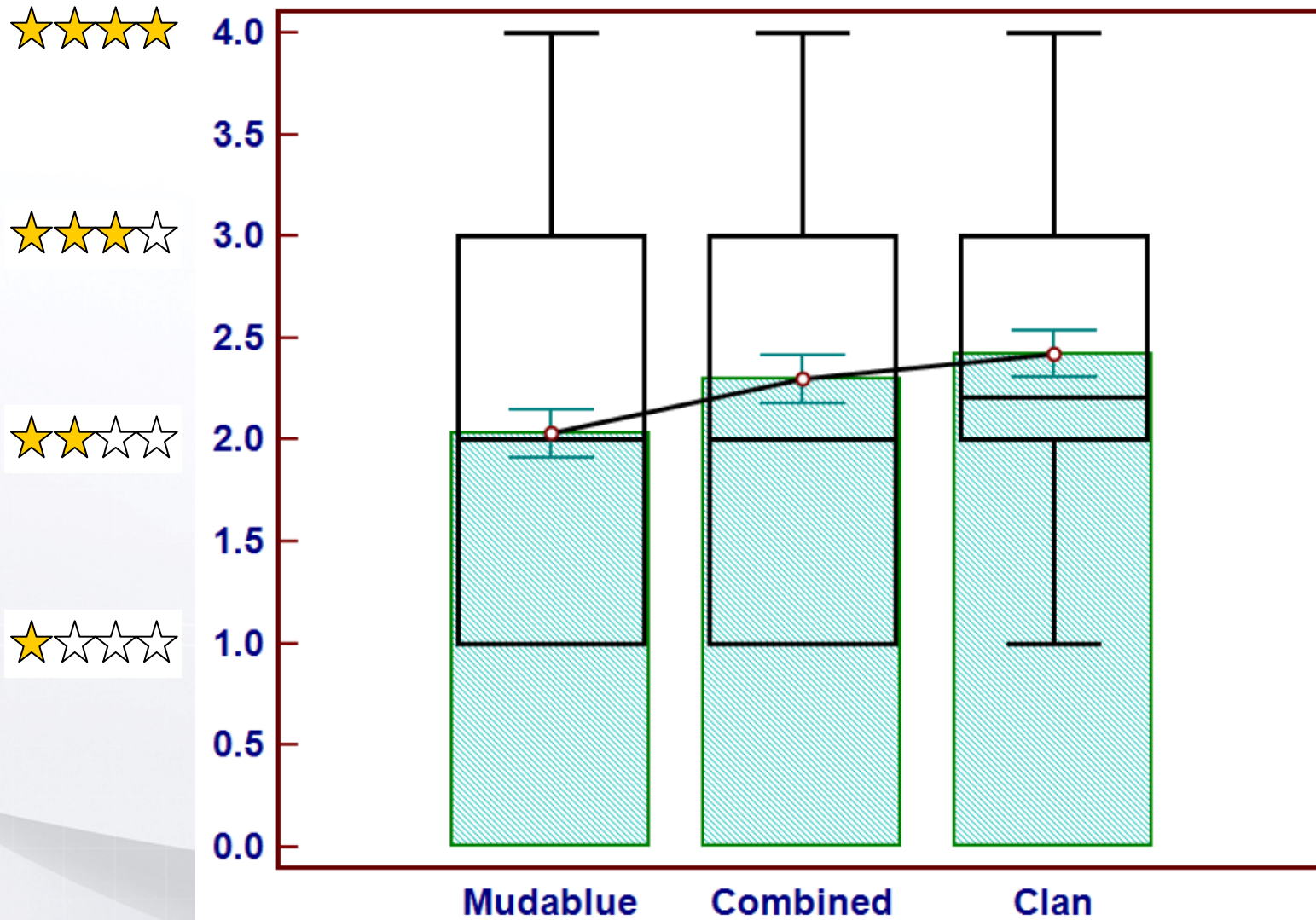
$H_5$ : Confidence of MUDABlue vs. Combined

$H_6$ : Precision of MUDABlue vs. Combined

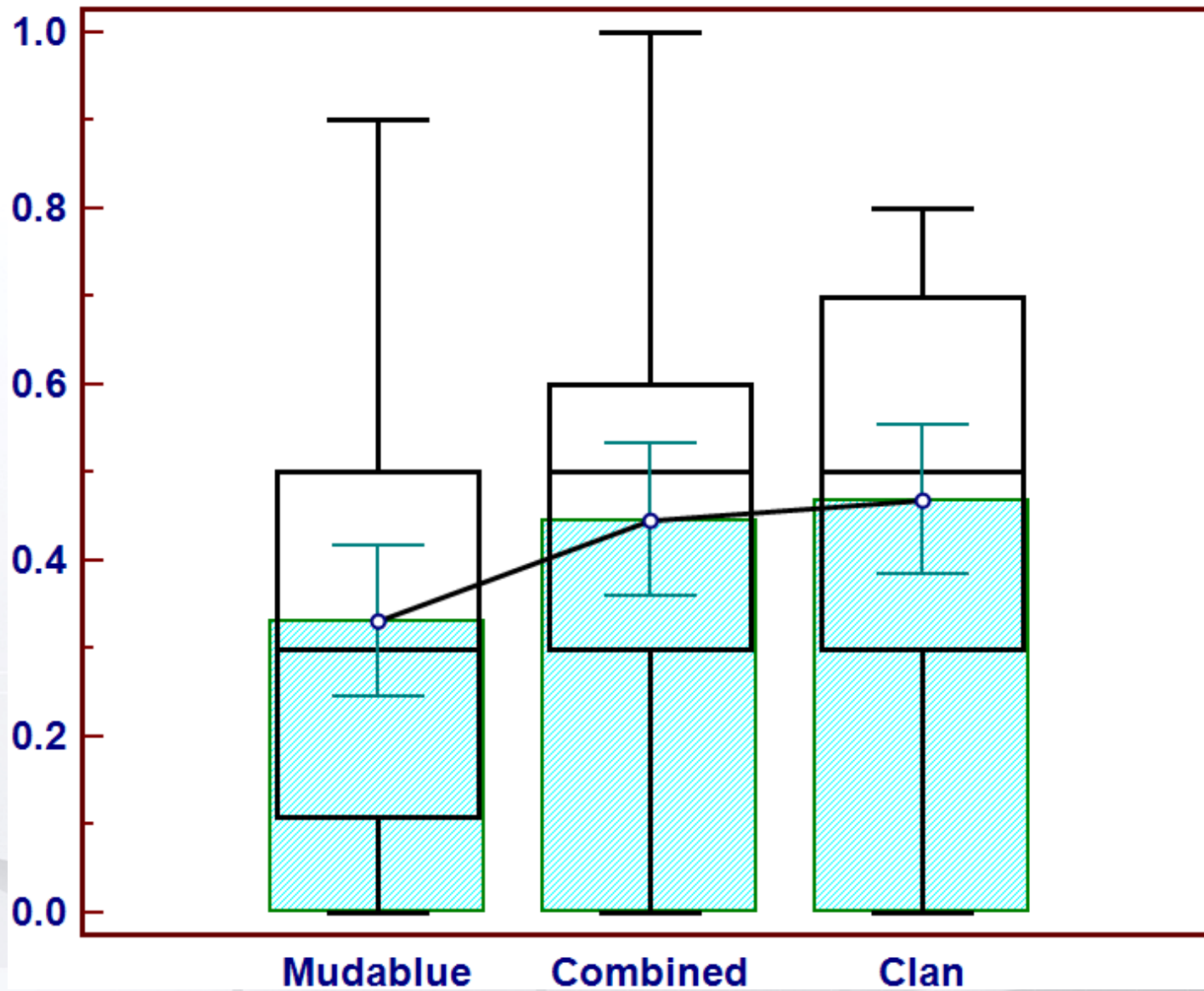


# Results – Confidence

p	$< 4.4 \cdot 10^{-7}$
F	5.02
F <sub>crit</sub>	1.97



## Results – Precision



p	< 0.02
F	2.43
F <sub>crit</sub>	2.04

# Accepted and Rejected Alternative Hypotheses

$H_1$ : Confidence of CLAN vs. MUDABlue



$H_2$ : Precision of CLAN vs. MUDABlue



$H_3$ : Confidence of CLAN vs. Combined



$H_4$ : Precision of CLAN vs. Combined



$H_5$ : Confidence of MUDABlue vs. Combined



$H_6$ : Precision of MUDABlue vs. Combined



## Responses from Programmers

“This search engine is better than MUDABlue because of the extra information provided within the results.”

“I think this is a helpful tool in finding the code one is looking for, but it can be very hit or miss. The hits were very relevant (4's) and the misses were completely irrelevant (1's or 2's).”

“Good comparison of API calls.”

“By using API calls I was able to compare the applications very easily.”



## Suggestions from Programmers


“However, it would be nice to see within the results the actual code, which made calls to function X or used library X”

“While this search engine finds apps which use relevant libraries it does not make it easy to find relevant sections within those projects. It would be helpful if there was functionality to better analyze the results”

“Rank API calls, ignore less significant API calls to return better relevant search results.”



# Threats to Validity

- Participants: proficiency in Java, development experience, and motivation
  - Selecting tasks for the experiment: too general or too specific?
  - On the use of Java SDK APIs
- 

# Ongoing Improvements

All Engines are Publicly Available

CLAN: <http://www.javaclan.net/>

MUDABlue: <http://www.mudablue.net/>

Combined: <http://clancombined.net/>

Case Study Tasks and Responses are available:

<http://www.cs.wm.edu/semeru/clan/>

Improving User Interface

Comparison of API calls, show source code

Generate explanations on why apps are similar

# Conclusions

We created a novel search system for finding Closely reLated applications (CLAN) that helps users find similar or related applications.

Our main contribution is in using a framework for relevance to design a novel approach that computes similarity scores between Java applications.



