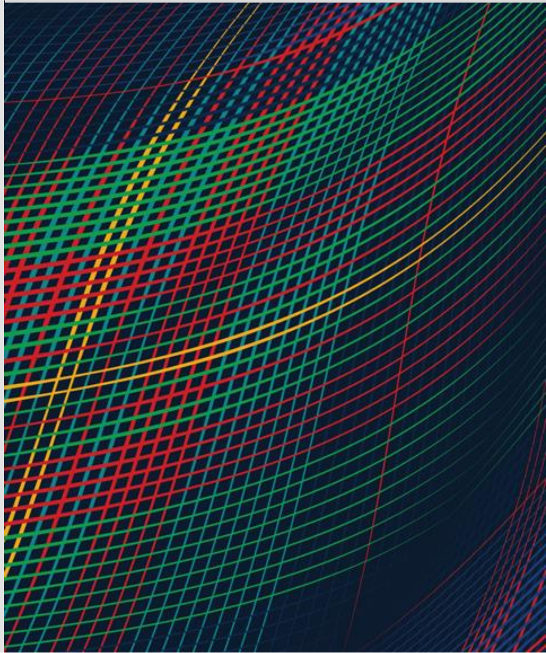
Carnegie Mellon University Software Engineering Institute

Role of Automation in Reducing Software Refactoring Costs



DECEMBER 18-20, 2023

Mario Benitez Preciado

James Ivers

Document Markings

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

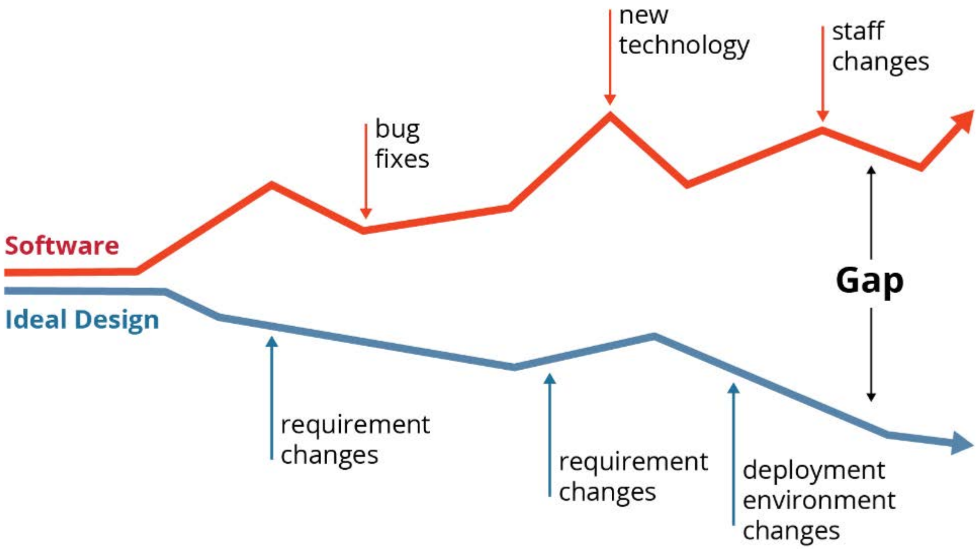
The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM23-2183

Software Is Never Done



Change is inevitable

- Requirements change
- Business priorities change
- Programming languages change
- Deployment environments change
- Technologies and platforms change
- Interacting systems change
- ...

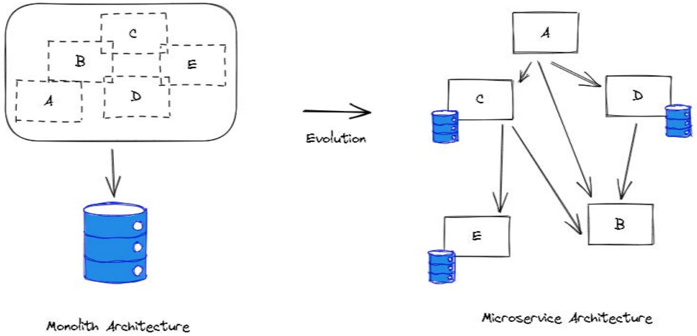
Periodic Refactoring Is Key to Keeping Code Healthy

Software must be delivered on time and on budget.

Messy software slows down development teams' ability to deliver new features or address existing issues.

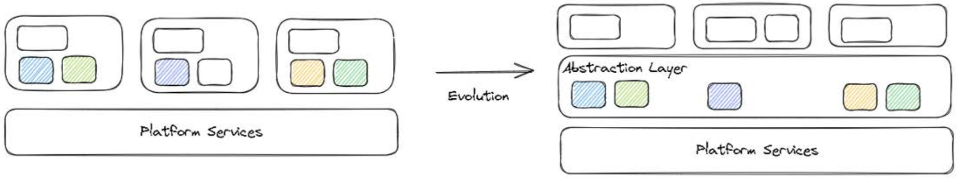
Mission-critical software must evolve over time in response to mission needs.

Examples of Refactoring Goals that Enable Evolution



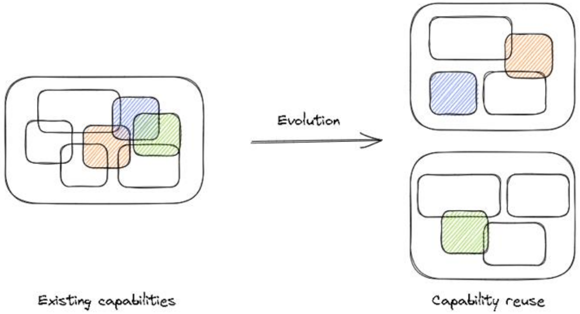
Microservice Architecture

Replicating software capabilities is a popular technique to scale software (e.g., via a microservice architecture).



Abstraction Layer

Isolating capabilities so that they can later be replaced with a better option.



Software Library

Reusing software to increase software quality, reduce development times, and save money (e.g., modular monolith).

Refactoring Gets Harder at Scale

Large-Scale Refactoring

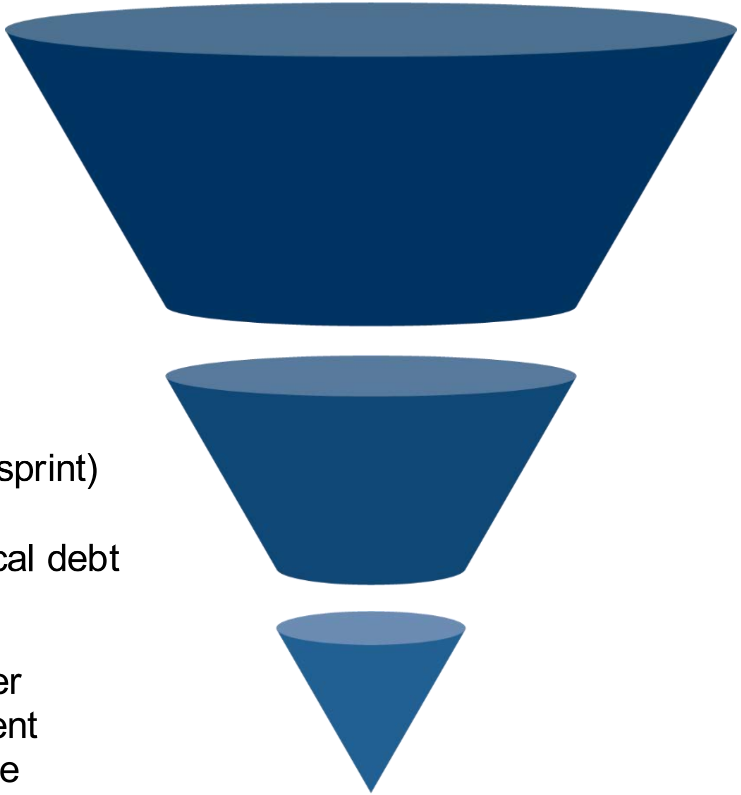
- Changes require substantial effort and coordination among multiple teams of developers
- Measured in staff months to years
- Architecture changes and non-local effects

Refactoring Sprints

- Changes made by a single team
- Often time-boxed (e.g., a two-week sprint)
- Effects limited to a single service
- E.g., 20% reserve to remove technical debt

“Floss Refactoring”

- Changes made by a single developer
- Intermingled with feature development
- Measured in minutes to hours of time
- Local effects



As scale increases,

↑ cross-team coordination increases

↑ technical risk increases

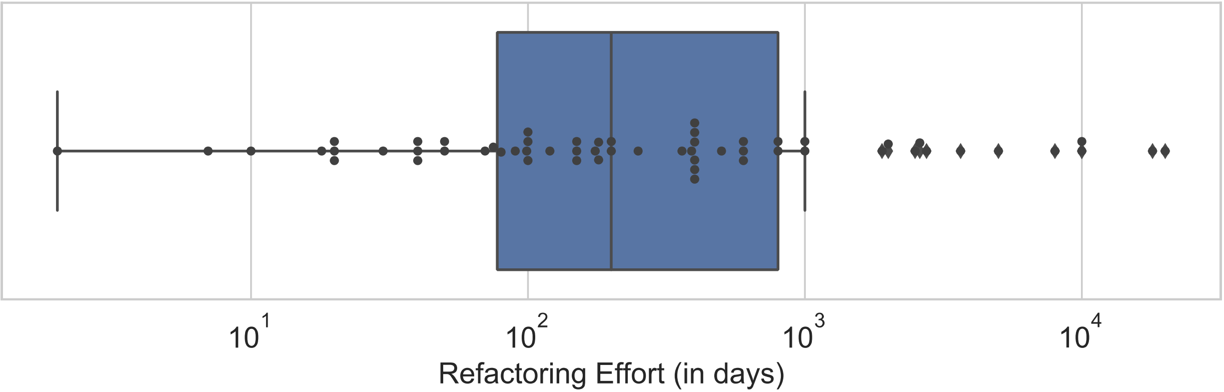
↑ cost and schedule impacts increase

↓ likelihood of securing funding *decreases*

Large-Scale Refactoring (LSR) in Industry

- Most respondents had performed LSR multiple times
- Most systems on which they had performed LSR had undergone LSR multiple times
- Mean of 1,500 staff days to perform LSR

We surveyed 107 industry practitioners to understand the state of the practice.

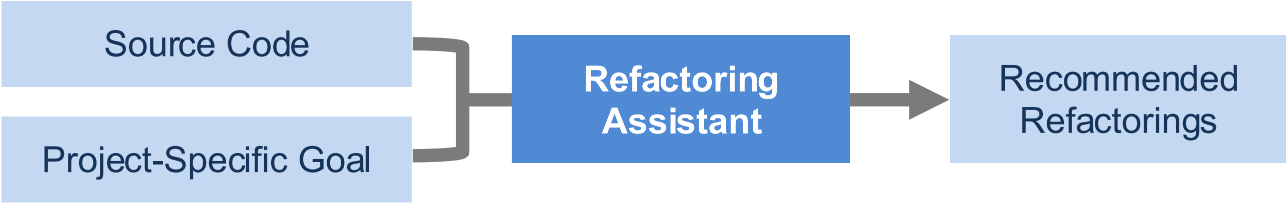


J. Ivers, R. Nord, I. Ozkaya, C. Seifried, C. Timperley, M. Kessentini. **Industry Experiences with Large-Scale Refactoring.** *Foundations of Software Engineering: Software Engineering in Practice* (ESEC/FSE). November 2022.

J. Ivers, R. Nord, I. Ozkaya, C. Seifried, C. Timperley, M. Kessentini. **Industry's Cry for Tools That Support Large-Scale Refactoring.** *Intl. Conference on Software Engineering: Software Engineering in Practice* (ICSE-SEIP). May 2022.

Untangling the Knot

The SEI has developed an automated Refactoring Assistant for developers reduce refactoring costs using a semi-automated approach.

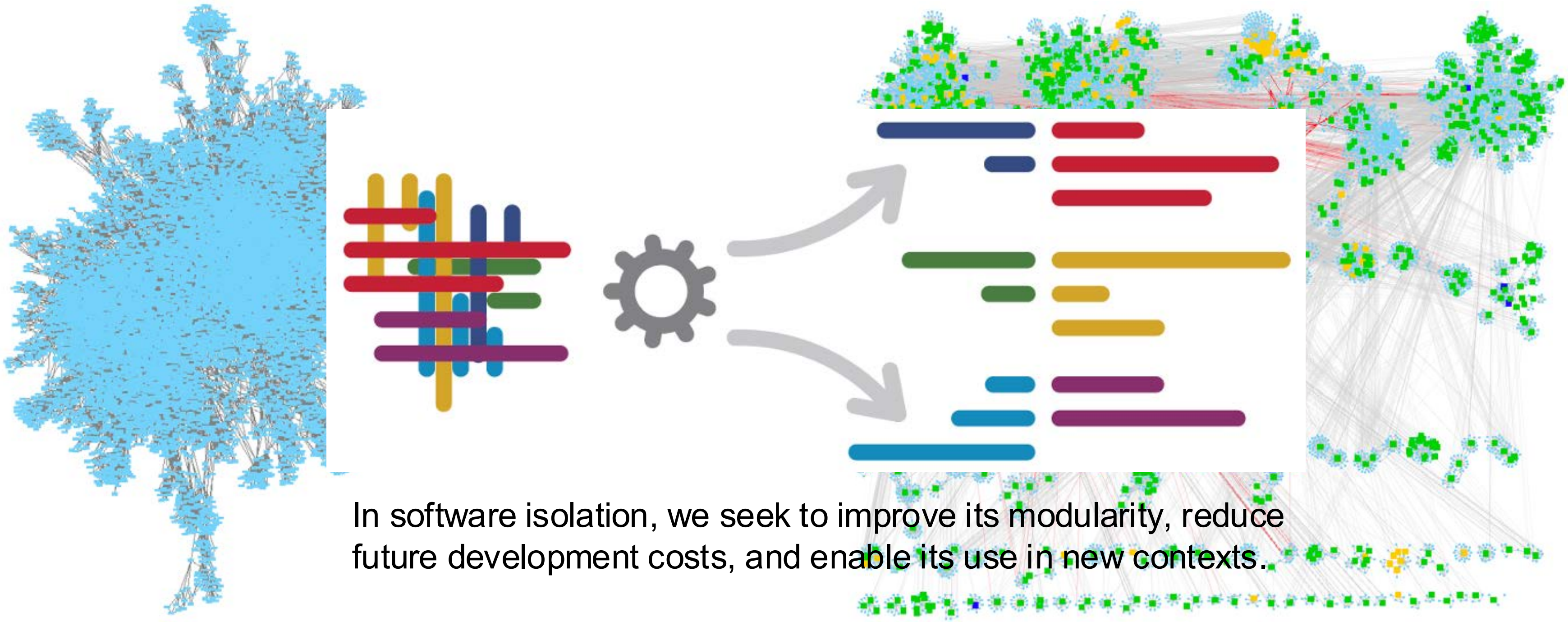


J. Ivers, C. Seifried, I. Ozkaya. **Untangling the Knot: Enabling Architecture Evolution with Search-Based Refactoring**. *19th IEEE International Conference on Software Architecture (ICSA 2023).* 2023.

Our Refactoring Assistant helps modularize specific capabilities, isolating their implementations from surrounding code.

It works with **C#** and **Java** code bases today, with C/C++ support in the works.

Software Modularization Is a Recurring Challenge



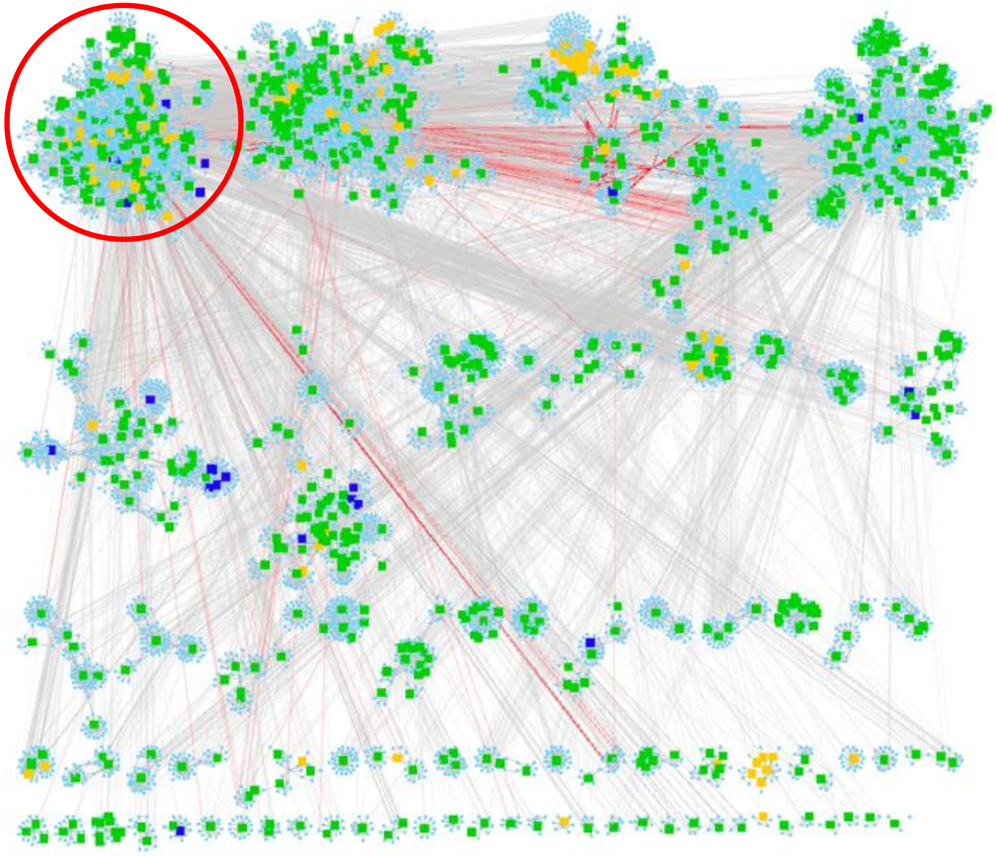
In software isolation, we seek to improve its modularity, reduce future development costs, and enable its use in new contexts.

A “simple” view of only 68K LOC.

Examples include
- strategic reuse
- rehosting on new platforms
- moving to the cloud

There is structure in this data, but that structure doesn't always let us do what we need to do.

Key Concept – Problematic Couplings



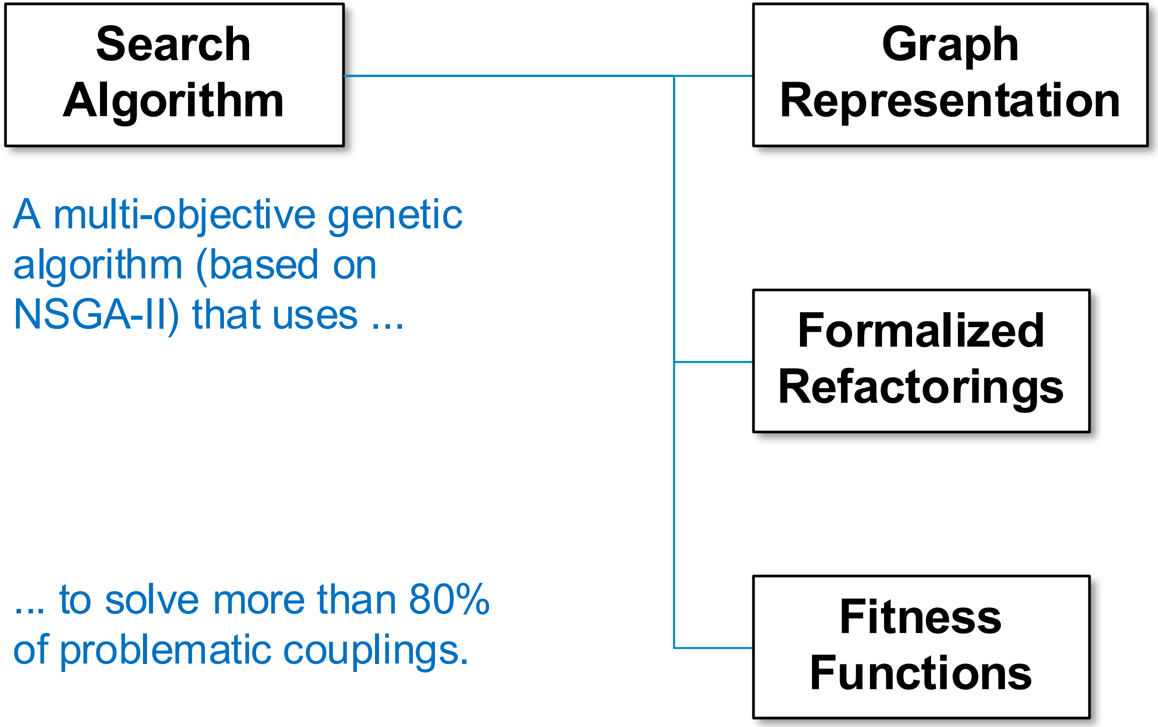
Only certain software dependencies interfere with any particular goal.

For example, if we want to harvest a feature:

- The core problem is dependencies (red lines) from software being harvested to software that is being left behind.
- All other dependencies are irrelevant to the goal, allowing us to focus our analysis and search for solutions.

This insight enables us to apply **search-based software engineering** techniques and treat this as an **optimization problem**.

SEI’s Automated Refactoring Assistant



Search Algorithm

A multi-objective genetic algorithm (based on NSGA-II) that uses ...

Graph Representation

static code analysis to generate an intermediate representation,

Formalized Refactorings

Fowler-style refactorings that have been formalized in terms of the graph, and

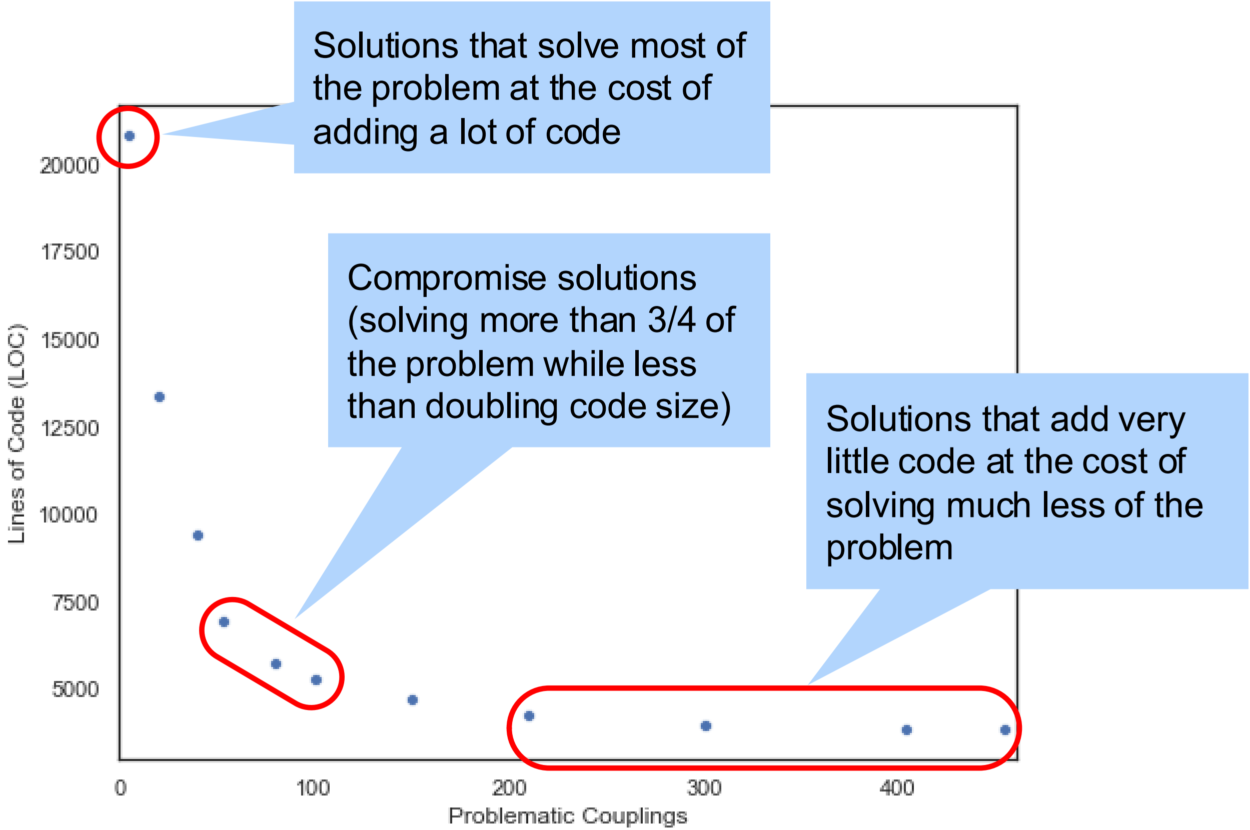
Fitness Functions

a collection of measures of the "goodness" of solutions for different objectives

... to solve more than 80% of problematic couplings.

K. Deb, A. Pratap, S. Agarwal, T. Meyarivan. **A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II.** *IEEE Transactions on Evolutionary Computation*. 2002.

Multi-objective Optimization



Solutions that solve most of the problem at the cost of adding a lot of code

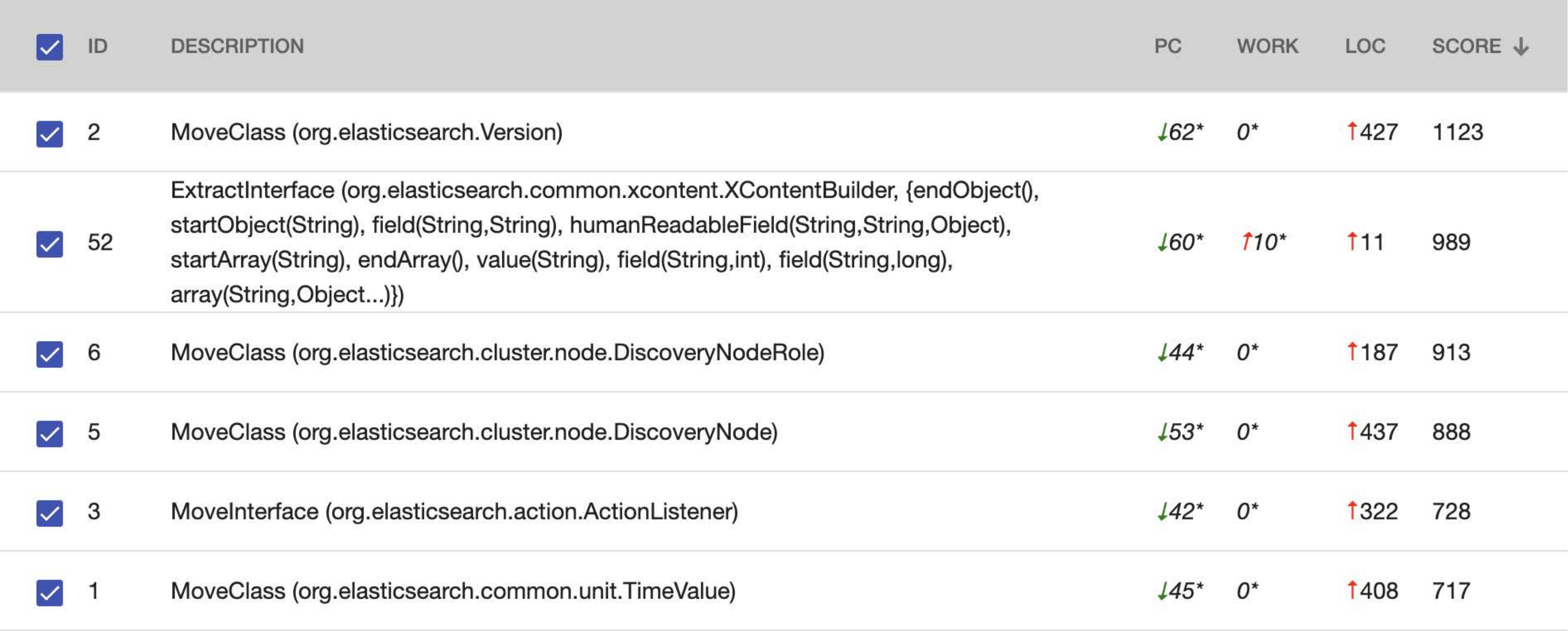
Compromise solutions (solving more than 3/4 of the problem while less than doubling code size)

Solutions that add very little code at the cost of solving much less of the problem

When optimizing for multiple objectives, there is no single best answer; instead, we generate options that represent trade-offs among competing objectives.

This allows developers to choose the trade-offs that best match their needs.

Specific Refactoring Recommendations



☑	ID	DESCRIPTION	PC	WORK	LOC	SCORE ↓
☑	2	MoveClass (org.elasticsearch.Version)	↓62*	0*	↑427	1123
☑	52	ExtractInterface (org.elasticsearch.common.xcontent.XContentBuilder, {endObject(), startObject(String), field(String,String), humanReadableField(String,String,Object), startArray(String), endArray(), value(String), field(String,int), field(String,long), array(String,Object...)})	↓60*	↑10*	↑11	989
☑	6	MoveClass (org.elasticsearch.cluster.node.DiscoveryNodeRole)	↓44*	0*	↑187	913
☑	5	MoveClass (org.elasticsearch.cluster.node.DiscoveryNode)	↓53*	0*	↑437	888
☑	3	MoveInterface (org.elasticsearch.action.ActionListener)	↓42*	0*	↑322	728
☑	1	MoveClass (org.elasticsearch.common.unit.TimeValue)	↓45*	0*	↑408	717

Step by step instructions, many of which can be automated by modern IDEs.

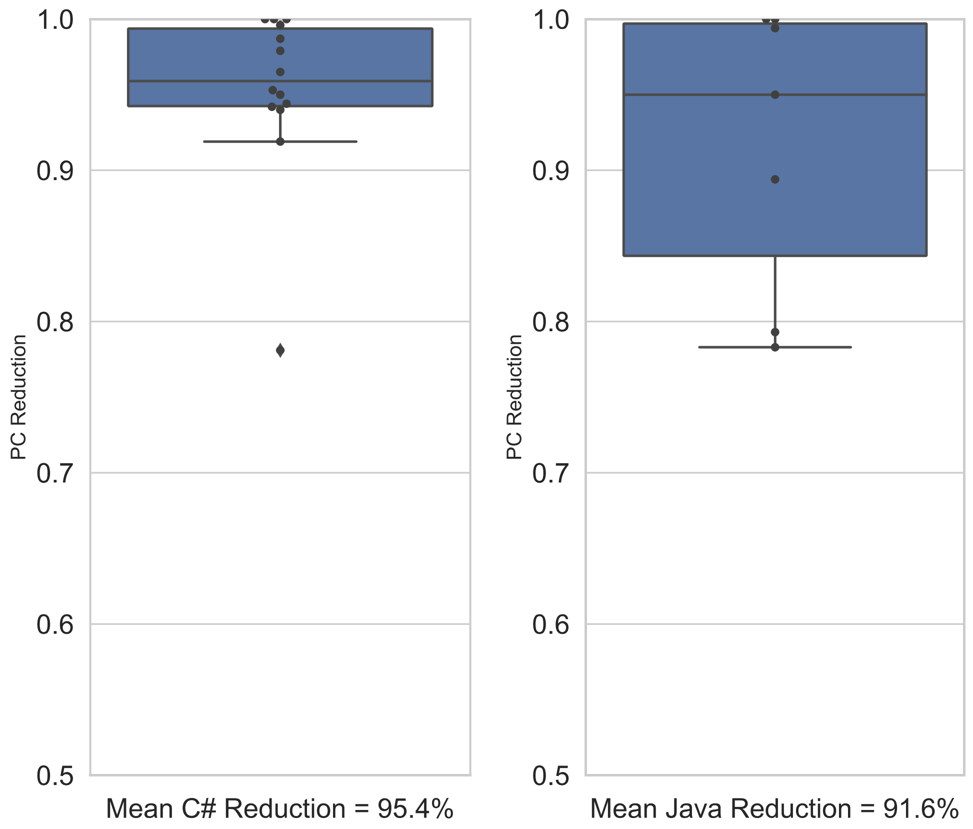
Current Capabilities

Programming languages supported

- Today: Java and C#
- In progress: C/C++

Our refactoring assistant

- scales to at least 2M SLOC
- generates recommendations that solve the majority of each software isolation problem

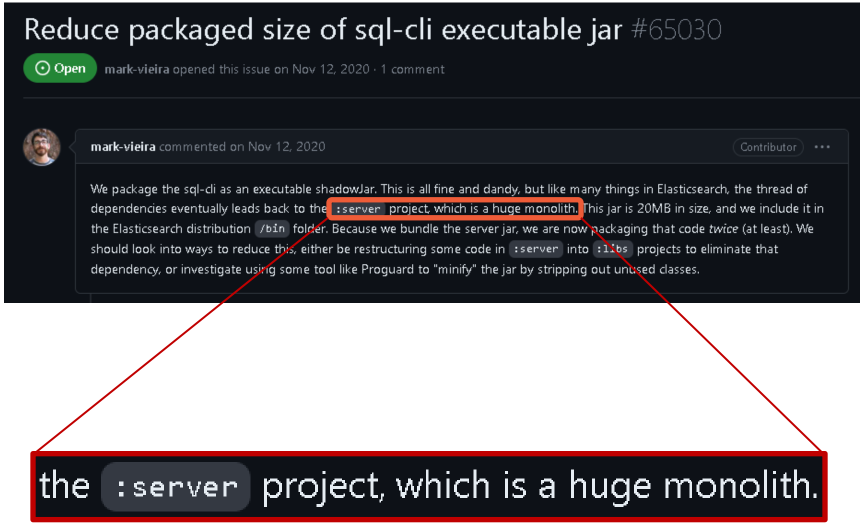


Case Study

Carnegie Mellon University Software Engineering Institute

Elasticsearch Case Study

- Utilize a real-world scenario as use case: Elasticsearch
- Go end-to-end:
  - apply the tool to the problem
  - implement refactorings
  - confirm success through test



Reduce packaged size of sql-cli executable jar #65030

Open mark-vieira opened this issue on Nov 12, 2020 · 1 comment

mark-vieira commented on Nov 12, 2020 Contributor

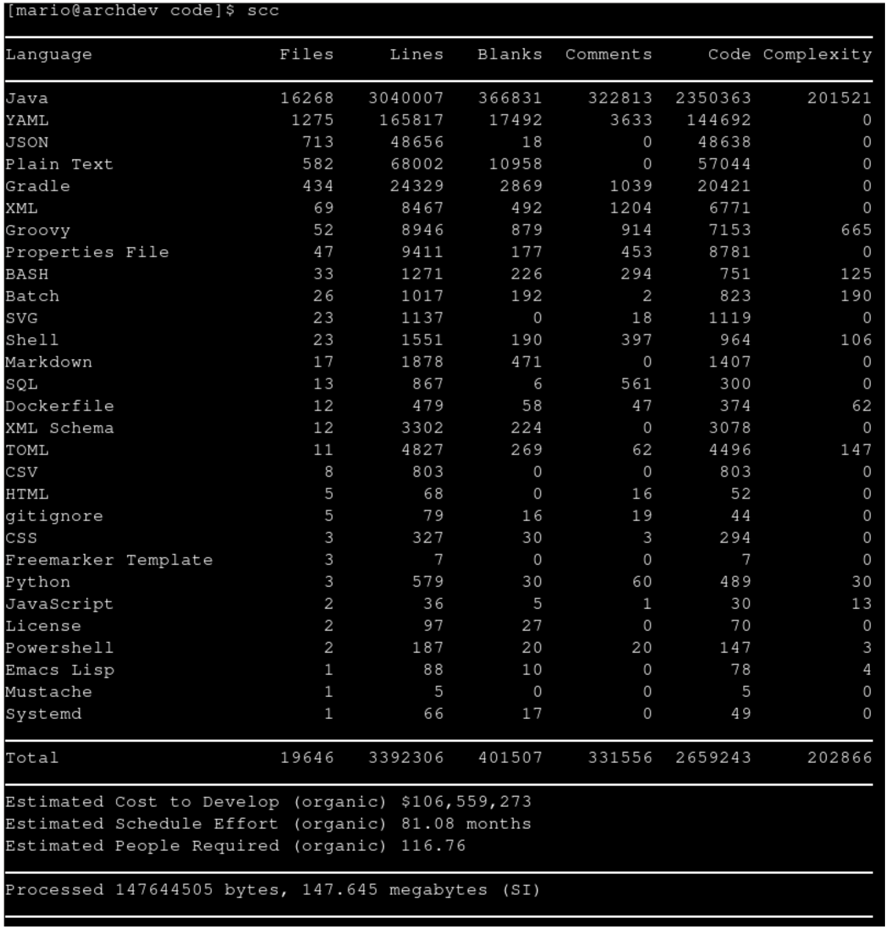
We package the sql-cli as an executable shadowJar. This is all fine and dandy, but like many things in Elasticsearch, the thread of dependencies eventually leads back to the `:server` project, which is a huge monolith. This jar is 20MB in size, and we include it in the Elasticsearch distribution `/bin` folder. Because we bundle the server jar, we are now packaging that code *twice* (at least). We should look into ways to reduce this, either be restructuring some code in `:server` into `:libs` projects to eliminate that dependency, or investigate using some tool like Proguard to "minify" the jar by stripping out unused classes.

the `:server` project, which is a huge monolith.

https://github.com/elastic/elasticsearch/issues/65030

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Elasticsearch Info



```
[mario@archdev code]$ scc
───────────────────────────────────────────────────────────────────────────────
Language                 Files     Lines   Blanks  Comments     Code Complexity
───────────────────────────────────────────────────────────────────────────────
Java                     16268   3040007   366831    322813  2350363     201521
YAML                      1275    165817    17492      3633   144692          0
JSON                       713     48656       18         0    48638          0
Plain Text                 582     68002    10958         0    57044          0
Gradle                     434     24329     2869      1039    20421          0
XML                         69      8467      492      1204     6771          0
Groovy                      52      8946      879       914     7153        665
Properties File             47      9411      177       453     8781          0
BASH                        33      1271      226       294      751        125
Batch                       26      1017      192         2      823        190
SVG                         23      1137        0        18     1119          0
Shell                       23      1551      190       397      964        106
Markdown                    17      1878      471         0     1407          0
SQL                         13       867        6       561      300          0
Dockerfile                  12       479       58        47      374         62
XML Schema                  12      3302      224         0     3078          0
TOML                        11      4827      269        62     4496        147
CSV                          8       803        0         0      803          0
HTML                         5        68        0        16       52          0
gitignore                    5        79       16        19       44          0
CSS                          3       327       30         3      294          0
Freemarker Template          3         7        0         0        7          0
Python                       3       579       30        60      489         30
JavaScript                   2        36        5         1       30         13
License                      2        97       27         0       70          0
Powershell                   2       187       20        20      147          3
Emacs Lisp                   1        88       10         0       78          4
Mustache                     1         5        0         0        5          0
Systemd                      1        66       17         0       49          0
───────────────────────────────────────────────────────────────────────────────
Total                    19646   3392306   401507    331556  2659243     202866
───────────────────────────────────────────────────────────────────────────────
Estimated Cost to Develop (organic) $106,559,273
Estimated Schedule Effort (organic) 81.08 months
Estimated People Required (organic) 116.76
───────────────────────────────────────────────────────────────────────────────
Processed 147644505 bytes, 147.645 megabytes (SI)
───────────────────────────────────────────────────────────────────────────────
```

- Elasticsearch is a distributed, RESTful search and analytics engine
- Over 2M lines of code
- Written in Java
- Over 13K tests available
- Build & test time ~ 2hr

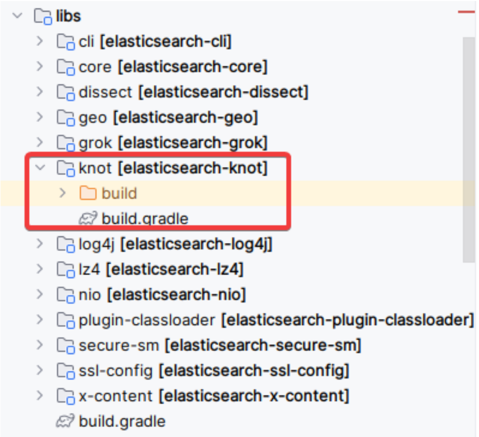
Anticipated Benefits

Modular code

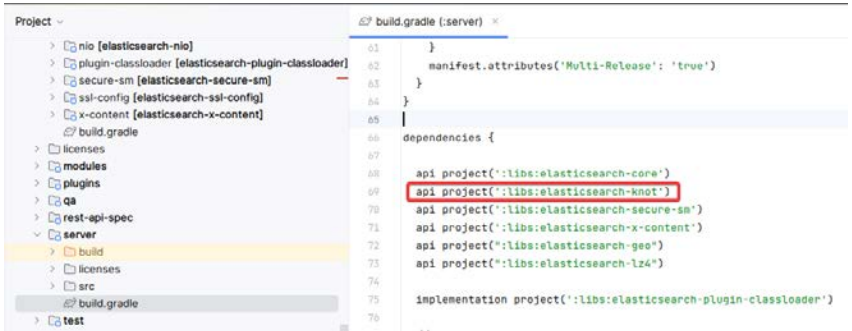
- Allows for work to be performed more independently
- Reduces build and test times
- Improves developer productivity (e.g., IDEs load and perform better with smaller code bases)
- Enables agile development and future improvements

Plan of Attack

- Extract a subset of functionality into an independent library (break the monolith!)
- Use the refactoring assistant to generate recommendations
  - Identify the target (scenario selection)
  - Run the tool (took ~21 minutes to generate solutions)
  - Pick a solution (solution selection)
- Follow the refactoring recommendations
- Complete the refactoring (PCs not solved by the tool)
- Run the tests and confirm that they pass



```
libs
  cli [elasticsearch-cli]
  core [elasticsearch-core]
  dissect [elasticsearch-dissect]
  geo [elasticsearch-geo]
  grok [elasticsearch-grok]
  knot [elasticsearch-knot]
    build
    build.gradle
  log4j [elasticsearch-log4j]
  lz4 [elasticsearch-lz4]
  nio [elasticsearch-nio]
  plugin-classloader [elasticsearch-plugin-classloader]
  secure-sm [elasticsearch-secure-sm]
  ssl-config [elasticsearch-ssl-config]
  x-content [elasticsearch-x-content]
  build.gradle
```

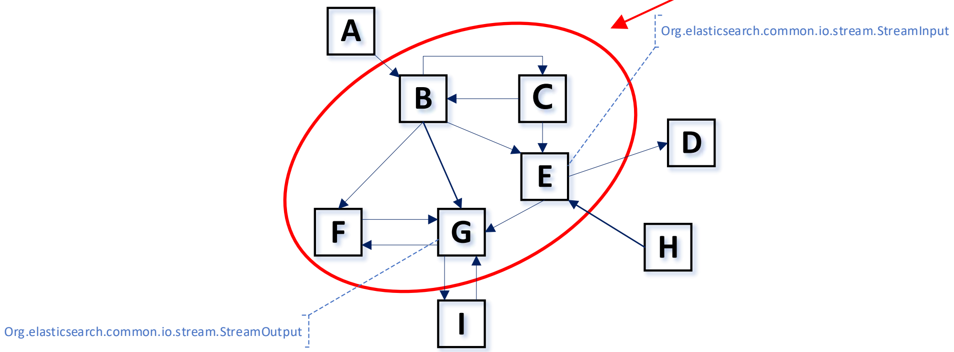


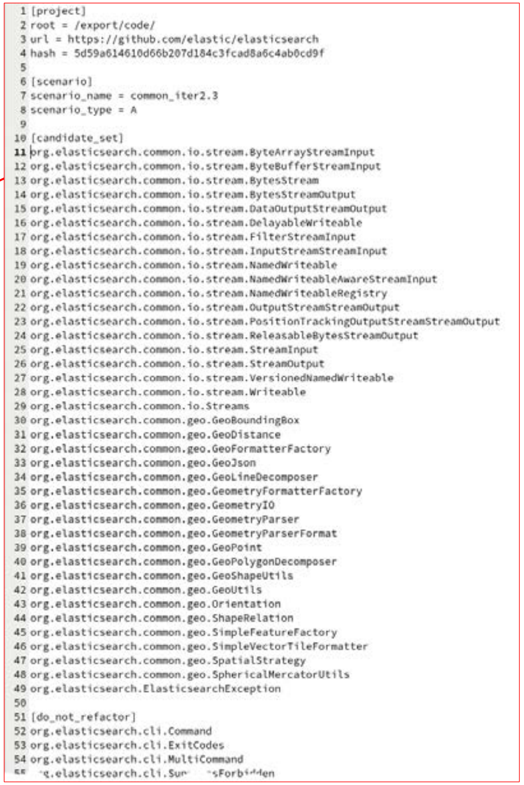
```
    nio [elasticsearch-nio]
    plugin-classloader [elasticsearch-plugin-classloader]
    secure-sm [elasticsearch-secure-sm]
    ssl-config [elasticsearch-ssl-config]
    x-content [elasticsearch-x-content]
    build.gradle
  licenses
  modules
  plugins
  qa
  rest-api-spec
  server
    build
    licenses
    src
    build.gradle
  test
```

```
    }
    manifest.attributes('Multi-Release': 'true')
  }
}

dependencies {

  api project(':libs:elasticsearch-core')
  api project(':libs:elasticsearch-knot')
  api project(':libs:elasticsearch-secure-sm')
  api project(':libs:elasticsearch-x-content')
  api project(":libs:elasticsearch-geo")
  api project(":libs:elasticsearch-lz4")

  implementation project(':libs:elasticsearch-plugin-classloader')
```

Scenario Selection

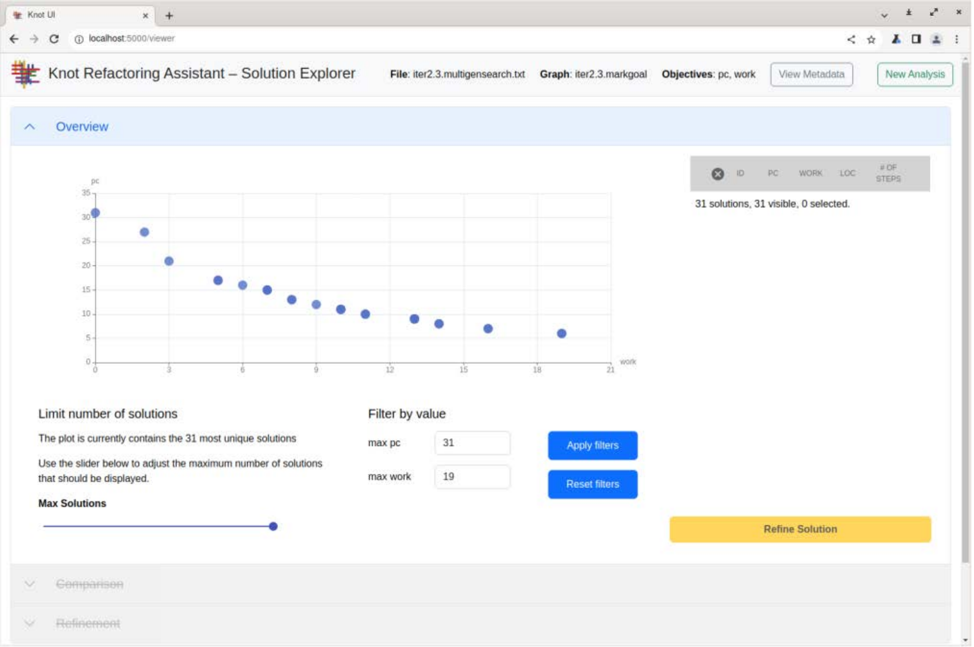
- Scenario selection has a huge impact on refactoring activities
- Start with lowest common denominator
- Selected **<u>StreamI/O</u>** and **<u>GeoUtils</u>** to be harvested from Server


A
B
C
D
E
F
G
H
I
Org.elasticsearch.common.io.stream.StreamInput
Org.elasticsearch.common.io.stream.StreamOutput



```
[project]
root = /export/code/
url = https://github.com/elastic/elasticsearch
hash = 5d59a614610d66b207d184c3fcad8a6c4ab0cd9f

[scenario]
scenario_name = common_iter2.3
scenario_type = A

[candidate_set]
org.elasticsearch.common.io.stream.ByteArrayStreamInput
org.elasticsearch.common.io.stream.ByteBufferStreamInput
org.elasticsearch.common.io.stream.BytesStream
org.elasticsearch.common.io.stream.BytesStreamOutput
org.elasticsearch.common.io.stream.DataOutputStreamOutput
org.elasticsearch.common.io.stream.DelayableWriteable
org.elasticsearch.common.io.stream.FilterStreamInput
org.elasticsearch.common.io.stream.InputStreamStreamInput
org.elasticsearch.common.io.stream.NamedWriteable
org.elasticsearch.common.io.stream.NamedWriteableAwareStreamInput
org.elasticsearch.common.io.stream.NamedWriteableRegistry
org.elasticsearch.common.io.stream.OutputStreamStreamOutput
org.elasticsearch.common.io.stream.PositionTrackingOutputStreamStreamOutput
org.elasticsearch.common.io.stream.ReleasableBytesStreamOutput
org.elasticsearch.common.io.stream.StreamInput
org.elasticsearch.common.io.stream.StreamOutput
org.elasticsearch.common.io.stream.VersionedNamedWriteable
org.elasticsearch.common.io.stream.Writeable
org.elasticsearch.common.io.Streams
org.elasticsearch.common.geo.GeoBoundingBox
org.elasticsearch.common.geo.GeoDistance
org.elasticsearch.common.geo.GeoFormatterFactory
org.elasticsearch.common.geo.GeoJson
org.elasticsearch.common.geo.GeoLineDecomposer
org.elasticsearch.common.geo.GeometryFormatterFactory
org.elasticsearch.common.geo.GeometryIO
org.elasticsearch.common.geo.GeometryParser
org.elasticsearch.common.geo.GeometryParserFormat
org.elasticsearch.common.geo.GeoPoint
org.elasticsearch.common.geo.GeoPolygonDecomposer
org.elasticsearch.common.geo.GeoShapeUtils
org.elasticsearch.common.geo.GeoUtils
org.elasticsearch.common.geo.Orientation
org.elasticsearch.common.geo.ShapeRelation
org.elasticsearch.common.geo.SimpleFeatureFactory
org.elasticsearch.common.geo.SimpleVectorTileFormatter
org.elasticsearch.common.geo.SpatialStrategy
org.elasticsearch.common.geo.SphericalMercatorUtils
org.elasticsearch.ElasticsearchException

[do_not_refactor]
org.elasticsearch.cli.Command
org.elasticsearch.cli.ExitCodes
org.elasticsearch.cli.MultiCommand
org.elasticsearch.cli.SuppressForbidden
```

Solution Selection



File: iter2.3.multigensearch.txt **Graph**: iter2.3.markgoal **Objectives**: pc, work

31 solutions, 31 visible, 0 selected.

Limit number of solutions

The plot is currently contains the 31 most unique solutions

Use the slider below to adjust the maximum number of solutions that should be displayed.

Filter by value

max pc 31

max work 19

- Used the refactoring assistant GUI to inspect potential solutions
  - Easy comparison of solutions (very useful!)
- Compared solutions at the edges of the chart (PCs vs Work)
- Reviewed the types of recommended steps (e.g., MoveClass vs ExtractInterface)
- Selected a solution

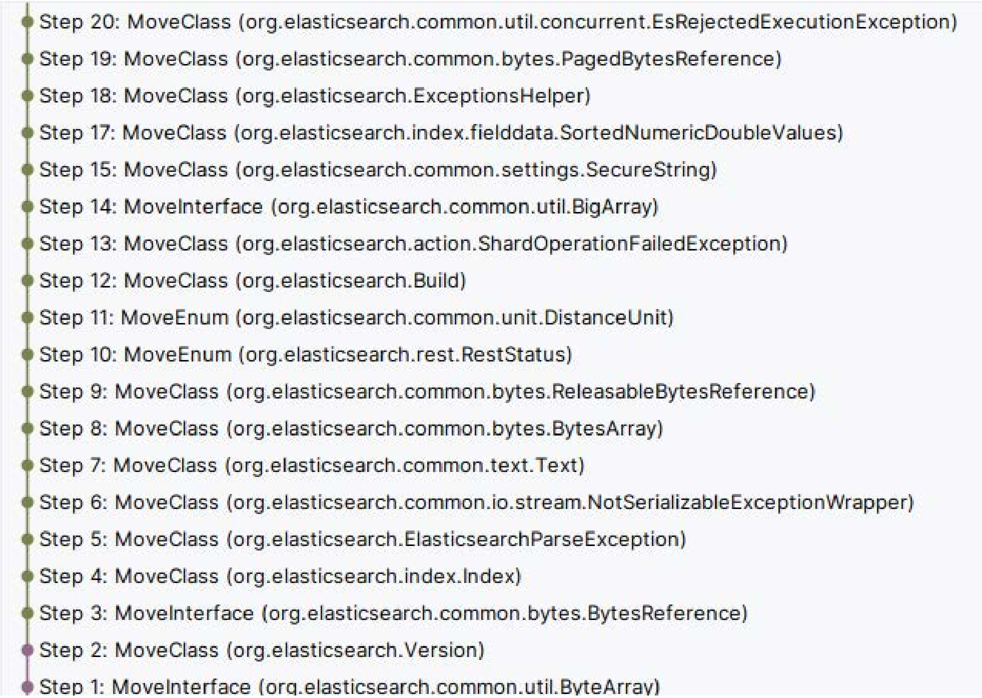
Overall process took ~15 mins

Following Refactoring Steps

Following solution steps generally fell into one of three categories:

1. Followed refactoring guidance
2. Deviated from refactoring guidance
3. Unresolved Problematic Couplings

Followed Refactoring Guidance

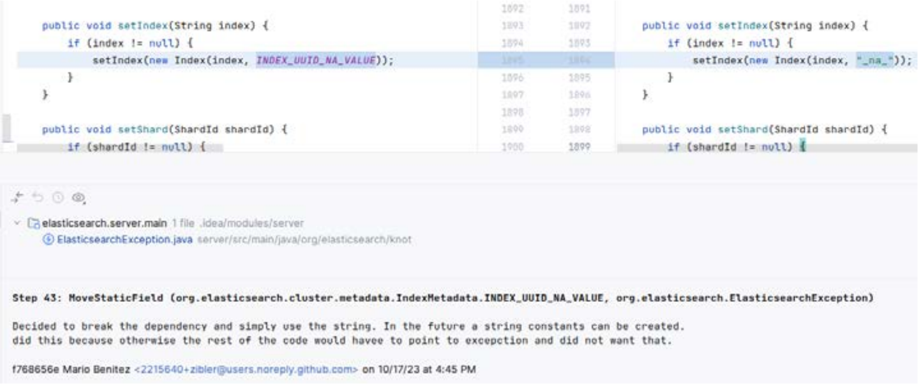


Step 20: MoveClass (org.elasticsearch.common.util.concurrent.EsRejectedExecutionException)
Step 19: MoveClass (org.elasticsearch.common.bytes.PagedBytesReference)
Step 18: MoveClass (org.elasticsearch.ExceptionsHelper)
Step 17: MoveClass (org.elasticsearch.index.fielddata.SortedNumericDoubleValues)
Step 15: MoveClass (org.elasticsearch.common.settings.SecureString)
Step 14: MoveInterface (org.elasticsearch.common.util.BigArray)
Step 13: MoveClass (org.elasticsearch.action.ShardOperationFailedException)
Step 12: MoveClass (org.elasticsearch.Build)
Step 11: MoveEnum (org.elasticsearch.common.unit.DistanceUnit)
Step 10: MoveEnum (org.elasticsearch.rest.RestStatus)
Step 9: MoveClass (org.elasticsearch.common.bytes.ReleasableBytesReference)
Step 8: MoveClass (org.elasticsearch.common.bytes.BytesArray)
Step 7: MoveClass (org.elasticsearch.common.text.Text)
Step 6: MoveClass (org.elasticsearch.common.io.stream.NotSerializableExceptionWrapper)
Step 5: MoveClass (org.elasticsearch.ElasticsearchParseException)
Step 4: MoveClass (org.elasticsearch.index.Index)
Step 3: MoveInterface (org.elasticsearch.common.bytes.BytesReference)
Step 2: MoveClass (org.elasticsearch.Version)
Step 1: MoveInterface (org.elasticsearch.common.util.ByteArray)

- Most solution steps fell in this category (41/46 or 89%)
- Most of these steps could be implemented automatically using the IDE
- Refactoring proceeded quickly (~4.5hrs) even though we were new to the code

Deviated from Refactoring Guidance

- Five steps (11%) were less than ideal or suggested changes the developer did not agree with
- The refactoring assistant allows us to deviate as we see fit
- For example:
  - Some dependencies can be easily broken by calling system methods directly
  - Step suggested creating an interface for static methods; a class is a better option



```
    public void setIndex(String index) {
        if (index != null) {
            setIndex(new Index(index, INDEX_UUID_NA_VALUE));
        }
    }

    public void setShard(ShardId shardId) {
        if (shardId != null) {
```

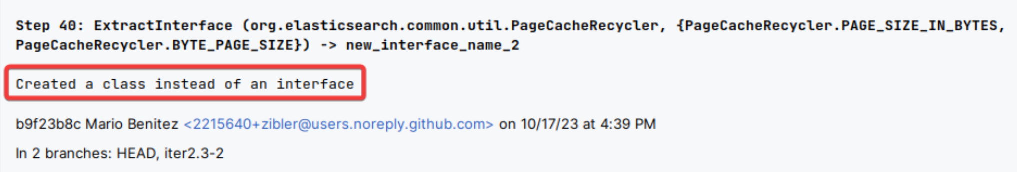
```
    public void setIndex(String index) {
        if (index != null) {
            setIndex(new Index(index, "_na_"));
        }
    }

    public void setShard(ShardId shardId) {
        if (shardId != null) {
```

elasticsearch.server.main 1 file .idea/modules/server

ElasticsearchException.java server/src/main/java/org/elasticsearch/knot

```
Step 43: MoveStaticField (org.elasticsearch.cluster.metadata.IndexMetadata.INDEX_UUID_NA_VALUE, org.elasticsearch.ElasticsearchException)

Decided to break the dependency and simply use the string. In the future a string constants can be created.
did this because otherwise the rest of the code would havee to point to excepction and did not want that.
```

f768656e Mario Benitez <2215640+zibler@users.noreply.github.com> on 10/17/23 at 4:45 PM



```
Step 40: ExtractInterface (org.elasticsearch.common.util.PageCacheRecycler, {PageCacheRecycler.PAGE_SIZE_IN_BYTES,
PageCacheRecycler.BYTE_PAGE_SIZE}) -> new_interface_name_2

Created a class instead of an interface
```

b9f23b8c Mario Benitez <2215640+zibler@users.noreply.github.com> on 10/17/23 at 4:39 PM

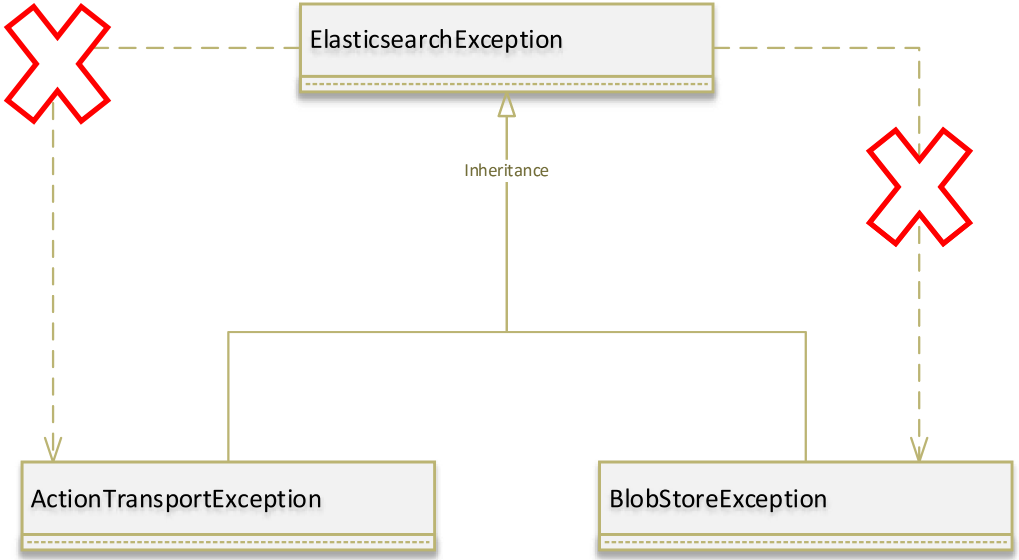
In 2 branches: HEAD, iter2.3-2

[DISTRIBUTION STATEMENT A] Approved for public release and unlimited distribution.

Unresolved Problematic Couplings (Finishing the Job)

- As mentioned before, not all problematic couplings are solved by the tool
- A *very* small percentage (2.8%) was left up to the developer to resolve
- AI augmented software engineering helps (a lot), but it won’t replace developers
- Let’s see an example…

Uncoupling Parent Class from Child Classes



Key: UML

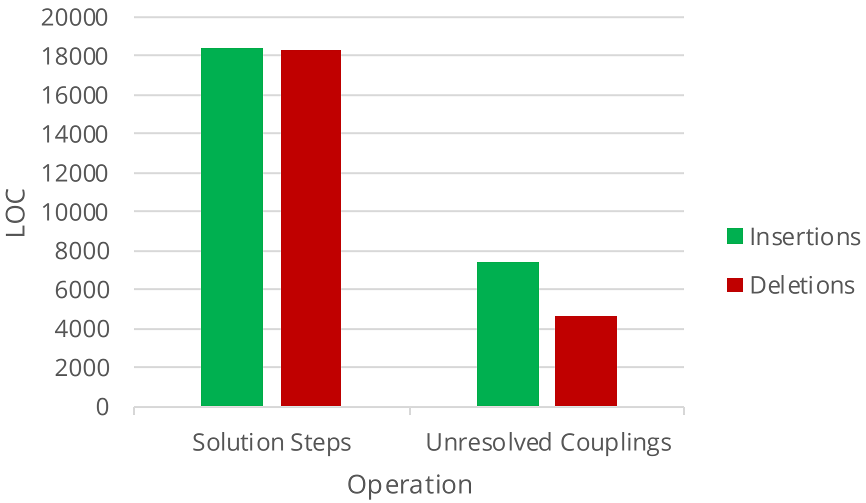
- The parent class ElasticsearchException depended on its children (>170 classes)
- This is less than ideal code (i.e., circular dependencies are bad!)

Solution: decouple responsibilities by removing index initialization maintained for other purposes

Put In Perspective

Followed refactoring guidance

- Moved 80 files, created 5 files
- Resolved 210 PCs **(97%)**
- In ~4.5hrs



Unresolved Problematic Couplings

- Moved 93 files, created 8 files
- Resolved 6 PCs **(3%)**
- In ~160hrs

Imagine if we did it all by hand?

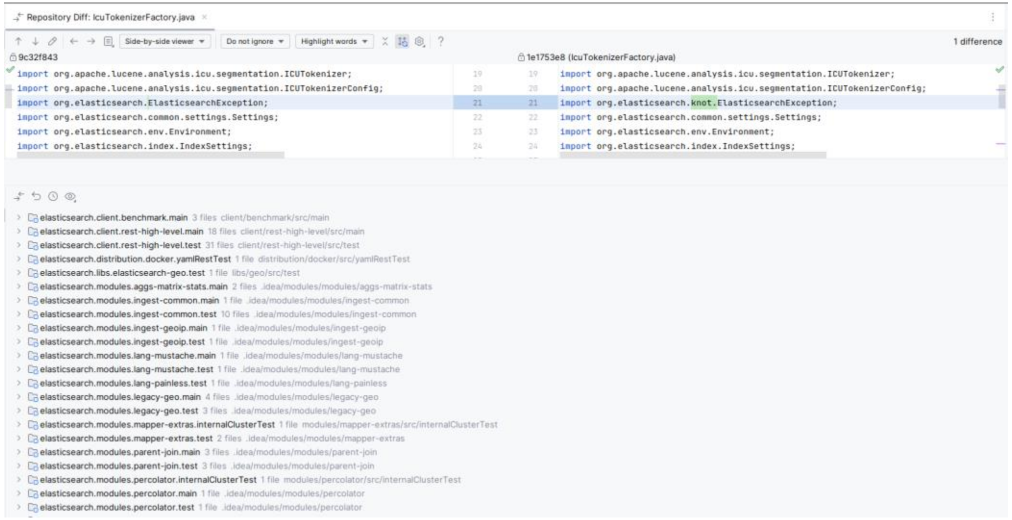
Lessons Learned

Your Refactoring Strategy Matters

- Create a strategy prior to refactoring to make it easier to implement
- Start with lowest common denominator
  - This example involved creating a library with the harvested code
- Suggestions for implementation strategy
  - Move code to a new package first. This helps identify PCs that need to be broken
  - Package as a new build unit last

IDEs Help

- Use IDEs to automatically implement simple code changes (e.g., imports and class references)
- Keep in mind that IDEs may make mistakes (they are still good tools!)
- Compile and test often to catch any errors introduced by the IDE.



Repository Diff: IcuTokenizerFactory.java

Side-by-side viewer | Do not ignore | Highlight words | 1 difference

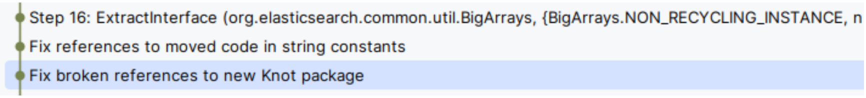
9c32f843

```
import org.apache.lucene.analysis.icu.segmentation.ICUTokenizer;
import org.apache.lucene.analysis.icu.segmentation.ICUTokenizerConfig;
import org.elasticsearch.ElasticsearchException;
import org.elasticsearch.common.settings.Settings;
import org.elasticsearch.env.Environment;
import org.elasticsearch.index.IndexSettings;
```

1e1753e8 (IcuTokenizerFactory.java)

```
import org.apache.lucene.analysis.icu.segmentation.ICUTokenizer;
import org.apache.lucene.analysis.icu.segmentation.ICUTokenizerConfig;
import org.elasticsearch.knot.ElasticsearchException;
import org.elasticsearch.common.settings.Settings;
import org.elasticsearch.env.Environment;
import org.elasticsearch.index.IndexSettings;
```

```
elasticsearch.client.benchmark.main 3 files client/benchmark/src/main
elasticsearch.client.rest-high-level.main 18 files client/rest-high-level/src/main
elasticsearch.client.rest-high-level.test 31 files client/rest-high-level/src/test
elasticsearch.distribution.docker.yamlRestTest 1 file distribution/docker/src/yamlRestTest
elasticsearch.libs.elasticsearch-geo.test 1 file libs/geo/src/test
elasticsearch.modules.aggs-matrix-stats.main 2 files .idea/modules/modules/aggs-matrix-stats
elasticsearch.modules.ingest-common.main 1 file .idea/modules/modules/ingest-common
elasticsearch.modules.ingest-common.test 10 files .idea/modules/modules/ingest-common
elasticsearch.modules.ingest-geoip.main 1 file .idea/modules/modules/ingest-geoip
elasticsearch.modules.ingest-geoip.test 1 file .idea/modules/modules/ingest-geoip
elasticsearch.modules.lang-mustache.main 1 file .idea/modules/modules/lang-mustache
elasticsearch.modules.lang-mustache.test 1 file .idea/modules/modules/lang-mustache
elasticsearch.modules.lang-painless.test 1 file .idea/modules/modules/lang-painless
elasticsearch.modules.legacy-geo.main 4 files .idea/modules/modules/legacy-geo
elasticsearch.modules.legacy-geo.test 3 files .idea/modules/modules/legacy-geo
elasticsearch.modules.mapper-extras.internalClusterTest 1 file modules/mapper-extras/src/internalClusterTest
elasticsearch.modules.mapper-extras.test 2 files .idea/modules/modules/mapper-extras
elasticsearch.modules.parent-join.main 3 files .idea/modules/modules/parent-join
elasticsearch.modules.parent-join.test 3 files .idea/modules/modules/parent-join
elasticsearch.modules.percolator.internalClusterTest 1 file modules/percolator/src/internalClusterTest
elasticsearch.modules.percolator.main 1 file .idea/modules/modules/percolator
elasticsearch.modules.percolator.test 1 file .idea/modules/modules/percolator
```

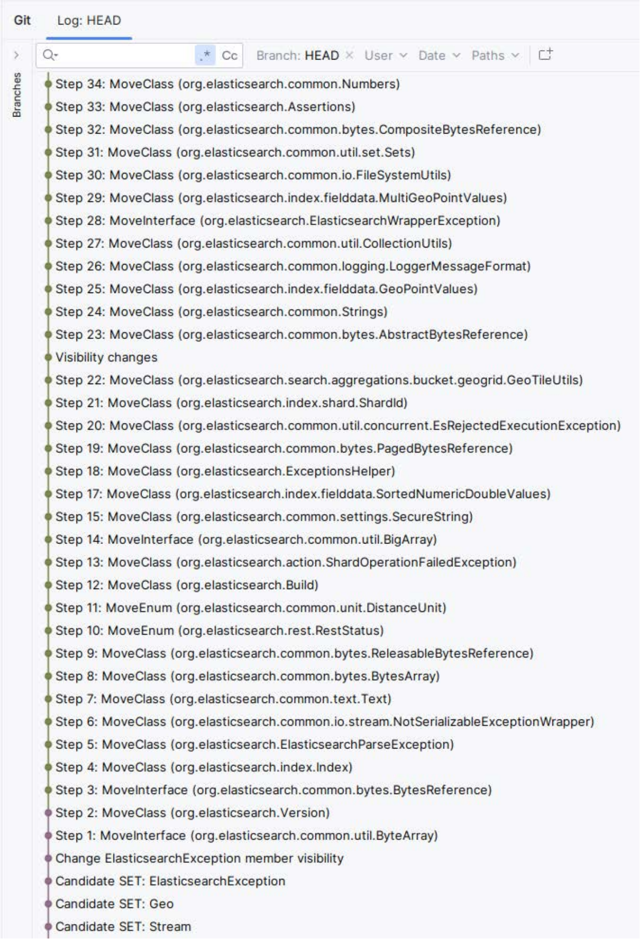


Step 16: ExtractInterface (org.elasticsearch.common.util.BigArrays, {BigArrays.NON_RECYCLING_INSTANCE, n

Fix references to moved code in string constants

Fix broken references to new Knot package

Work Incrementally



```
Step 34: MoveClass (org.elasticsearch.common.Numbers)
Step 33: MoveClass (org.elasticsearch.Assertions)
Step 32: MoveClass (org.elasticsearch.common.bytes.CompositeBytesReference)
Step 31: MoveClass (org.elasticsearch.common.util.set.Sets)
Step 30: MoveClass (org.elasticsearch.common.io.FileSystemUtils)
Step 29: MoveClass (org.elasticsearch.index.fielddata.MultiGeoPointValues)
Step 28: MoveInterface (org.elasticsearch.ElasticsearchWrapperException)
Step 27: MoveClass (org.elasticsearch.common.util.CollectionUtils)
Step 26: MoveClass (org.elasticsearch.common.logging.LoggerMessageFormat)
Step 25: MoveClass (org.elasticsearch.index.fielddata.GeoPointValues)
Step 24: MoveClass (org.elasticsearch.common.Strings)
Step 23: MoveClass (org.elasticsearch.common.bytes.AbstractBytesReference)
Visibility changes
Step 22: MoveClass (org.elasticsearch.search.aggregations.bucket.geogrid.GeoTileUtils)
Step 21: MoveClass (org.elasticsearch.index.shard.ShardId)
Step 20: MoveClass (org.elasticsearch.common.util.concurrent.EsRejectedExecutionException)
Step 19: MoveClass (org.elasticsearch.common.bytes.PagedBytesReference)
Step 18: MoveClass (org.elasticsearch.ExceptionsHelper)
Step 17: MoveClass (org.elasticsearch.index.fielddata.SortedNumericDoubleValues)
Step 15: MoveClass (org.elasticsearch.common.settings.SecureString)
Step 14: MoveInterface (org.elasticsearch.common.util.BigArray)
Step 13: MoveClass (org.elasticsearch.action.ShardOperationFailedException)
Step 12: MoveClass (org.elasticsearch.Build)
Step 11: MoveEnum (org.elasticsearch.common.unit.DistanceUnit)
Step 10: MoveEnum (org.elasticsearch.rest.RestStatus)
Step 9: MoveClass (org.elasticsearch.common.bytes.ReleasableBytesReference)
Step 8: MoveClass (org.elasticsearch.common.bytes.BytesArray)
Step 7: MoveClass (org.elasticsearch.common.text.Text)
Step 6: MoveClass (org.elasticsearch.common.io.stream.NotSerializableExceptionWrapper)
Step 5: MoveClass (org.elasticsearch.ElasticsearchParseException)
Step 4: MoveClass (org.elasticsearch.index.Index)
Step 3: MoveInterface (org.elasticsearch.common.bytes.BytesReference)
Step 2: MoveClass (org.elasticsearch.Version)
Step 1: MoveInterface (org.elasticsearch.common.util.ByteArray)
Change ElasticsearchException member visibility
Candidate SET: ElasticsearchException
Candidate SET: Geo
Candidate SET: Stream
```

- Revision control is your friend
- Create a commit for every step in the refactoring
- This allows for easy backtracking of changes when needed

Conclusions

Carnegie Mellon University Software Engineering Institute

Conclusions

- Refactoring large code bases is beneficial, but it takes time
- This is a first step (*bug's been opened for 3 years!*)
- Automation can speed up this process greatly
  - Refactoring assistant created a plan to solve 97% of PCs in ~21 minutes
  - IDEs automate much of the implementation
- Developers are still needed to review the plan and complete the work

- Working with modular code is better because...
  - 14K lines of code vs ~1M
  - 120 files vs ~6K
  - Fast IDE operations vs hourglass

Learn More

Learn more about the SEI's work in software isolation and large-scale refactoring:

https://sei.cmu.edu/go/knot

Contact us at **sei-knot@sei.cmu.edu**
if you are interested in collaborating.

THANK YOU!