Untangling the Knot
Enabling Rapid Software Evolution

Problem
To quickly deliver new capabilities and take advantage of new technologies, DoD needs the ability to efficiently restructure software for common scenarios like:

• migrating a capability to the cloud
• harvesting software for reuse
• containerizing software

One recent anecdote estimates the effort to isolate a capability from the platform at 14,000 staff hours just for development.

Solution
Create an automated assistant that rapidly refactors software to support software isolation goals that enable software evolution.

• Allows users to specify project-specific goals.
• Uses genetic algorithms to recommend refactorings.
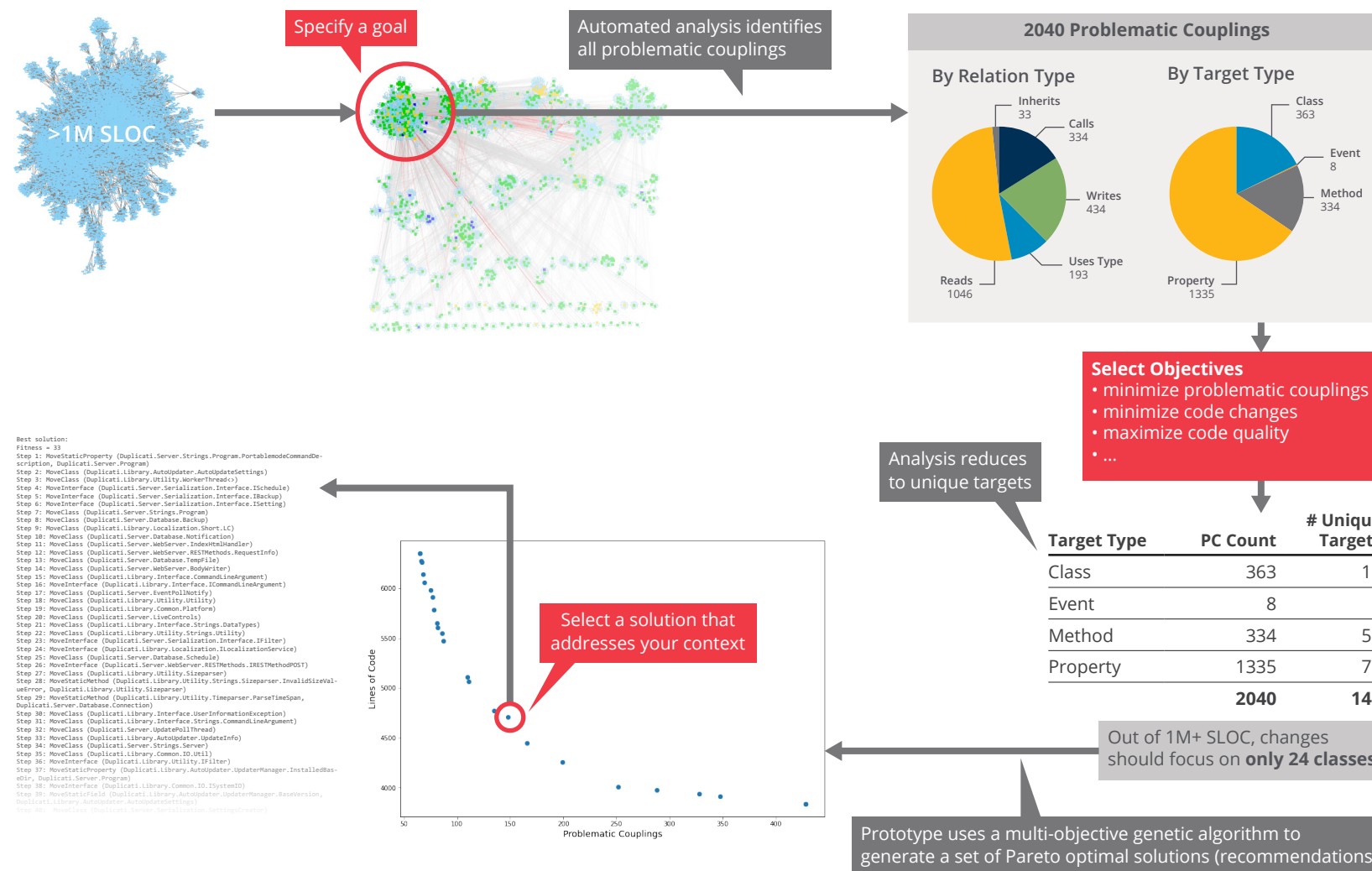• Navigates multiple, competing objectives.

Intended Outcomes (FY19–21)
• Refactoring recommendations outperform those based only on quality metrics, reducing problematic couplings by at least 75%.
• Our automation reduces the time to restructure software to 1/3 of the time compared to manual effort.

Read more about our vision:
J. Ivers, I. Ozkaya, R. L. Nord, C. Seifried, **Next Generation Automated Software Evolution: Refactoring at Scale**. 2020. *28th Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '20). ACM, Virtual Event, USA.*

Automated refactoring can improve the structure of existing software in **1/3 of the time** it takes to manually refactor.

Specify a goal

Automated analysis identifies all problematic couplings

>1M SLOC

2040 Problematic Couplings

By Relation Type
- Inherits 33
- Calls 334
- Writes 434
- Uses Type 193
- Reads 1046

By Target Type
- Class 363
- Event 8
- Method 334
- Property 1335

Select Objectives
• minimize problematic couplings
• minimize code changes
• maximize code quality
• ...

Analysis reduces to unique targets

Target Type	PC Count	# Unique Targets
Class	363	15
Event	8	1
Method	334	51
Property	1335	77
	2040	**144**

Out of 1M+ SLOC, changes should focus on **only 24 classes**

Select a solution that addresses your context

```
Best solution:
Fitness = 33
Step 1: MoveStaticProperty (Duplicati.Server.Strings.Program.PortablemodeCommandDe-
scription, Duplicati.Server.Program)
Step 2: MoveClass (Duplicati.Library.AutoUpdater.AutoUpdateSettings)
Step 3: MoveClass (Duplicati.Library.Utility.WorkerThread<>)
Step 4: MoveInterface (Duplicati.Server.Serialization.Interface.ISchedule)
Step 5: MoveInterface (Duplicati.Server.Serialization.Interface.IBackup)
Step 6: MoveInterface (Duplicati.Server.Serialization.Interface.ISetting)
Step 7: MoveClass (Duplicati.Server.Strings.Program)
Step 8: MoveClass (Duplicati.Server.Database.Backup)
Step 9: MoveClass (Duplicati.Library.Localization.Short.LC)
Step 10: MoveClass (Duplicati.Server.Database.Notification)
Step 11: MoveClass (Duplicati.Server.WebServer.IndexHtmlHandler)
Step 12: MoveClass (Duplicati.Server.WebServer.RESTMethods.RequestInfo)
Step 13: MoveClass (Duplicati.Server.Database.TempFile)
Step 14: MoveClass (Duplicati.Server.WebServer.BodyWriter)
Step 15: MoveClass (Duplicati.Library.Interface.CommandLineArgument)
Step 16: MoveInterface (Duplicati.Library.Interface.ICommandLineArgument)
Step 17: MoveClass (Duplicati.Server.EventPollNotify)
Step 18: MoveClass (Duplicati.Library.Utility.Utility)
Step 19: MoveClass (Duplicati.Library.Common.Platform)
Step 20: MoveClass (Duplicati.Server.LiveControls)
Step 21: MoveClass (Duplicati.Library.Interface.Strings.DataTypes)
Step 22: MoveClass (Duplicati.Library.Utility.Strings.Utility)
Step 23: MoveInterface (Duplicati.Server.Serialization.Interface.IFilter)
Step 24: MoveInterface (Duplicati.Library.Localization.ILocalizationService)
Step 25: MoveClass (Duplicati.Server.Database.Schedule)
Step 26: MoveInterface (Duplicati.Server.WebServer.RESTMethods.IRESTMethodPOST)
Step 27: MoveClass (Duplicati.Library.Utility.Sizeparser)
Step 28: MoveStaticMethod (Duplicati.Library.Utility.Strings.Sizeparser.InvalidSizeVal-
ueError, Duplicati.Library.Utility.Sizeparser)
Step 29: MoveStaticMethod (Duplicati.Library.Utility.Timeparser.ParseTimeSpan,
Duplicati.Server.Database.Connection)
Step 30: MoveClass (Duplicati.Library.Interface.UserInformationException)
Step 31: MoveClass (Duplicati.Library.Interface.Strings.CommandLineArgument)
Step 32: MoveClass (Duplicati.Server.UpdatePollThread)
Step 33: MoveClass (Duplicati.Library.AutoUpdater.UpdateInfo)
Step 34: MoveClass (Duplicati.Server.Strings.Server>)
Step 35: MoveClass (Duplicati.Library.Common.IO.Util)
Step 36: MoveInterface (Duplicati.Library.Utility.IFilter)
Step 37: MoveStaticProperty (Duplicati.Library.AutoUpdater.UpdaterManager.InstalledBas-
eDir, Duplicati.Server.Program)
Step 38: MoveInterface (Duplicati.Library.Common.IO.ISystemIO)
Step 39: MoveStaticField (Duplicati.Library.AutoUpdater.UpdaterManager.BaseVersion,
Duplicati.Library.AutoUpdater.AutoUpdateSettings)
Step 40: MoveClass (Duplicati.Server.Serialization.SettingsCreator)
```


Lines of Code / Problematic Couplings

Prototype uses a multi-objective genetic algorithm to generate a set of Pareto optimal solutions (recommendations)

Our prototype can help with common evolution scenarios:

Scenario
Gather data to assess the difficulty associated with project-specific goals as input to funding decisions.

Maturity
Available now (TRL 4)

Expected Results
Enumeration of problematic couplings, their locations, and types potentially impacted by proposed change as data to inform cost estimates.

Scenario
Compare the difficulty of different refactoring approaches.

Maturity
Available now (TRL 4)

Expected Results
Enumeration of problematic couplings, their locations, and types potentially impacted by proposed change as data to inform cost estimates.

Scenario
Automatically refactor software to isolate software and speed its evolution.

Maturity
Ready for pilot application in 3–6 months.

Expected Results
Recommended refactorings that enable the proposed change address multiple criteria.

Contact us at **info@sei.cmu.edu** if you are interested in partnering with us.

James Ivers, Ipek Ozkaya, Robert Nord, Chris Seifried
Mario Benitez, Jared Frank, Carly Jones, Andrew Kotov, Reed Little, Craig Mazzotta, Scott Pavetti, Jeff Yackley

James Ivers, Ipek Ozkaya, Robert Nord, Chris Seifried
Mario Benitez, Jared Frank, Carly Jones, Andrew Kotov, Reed Little, Craig Mazzotta, Scott Pavetti, Jeff Yackley