KI-gestützte Java Migrationen

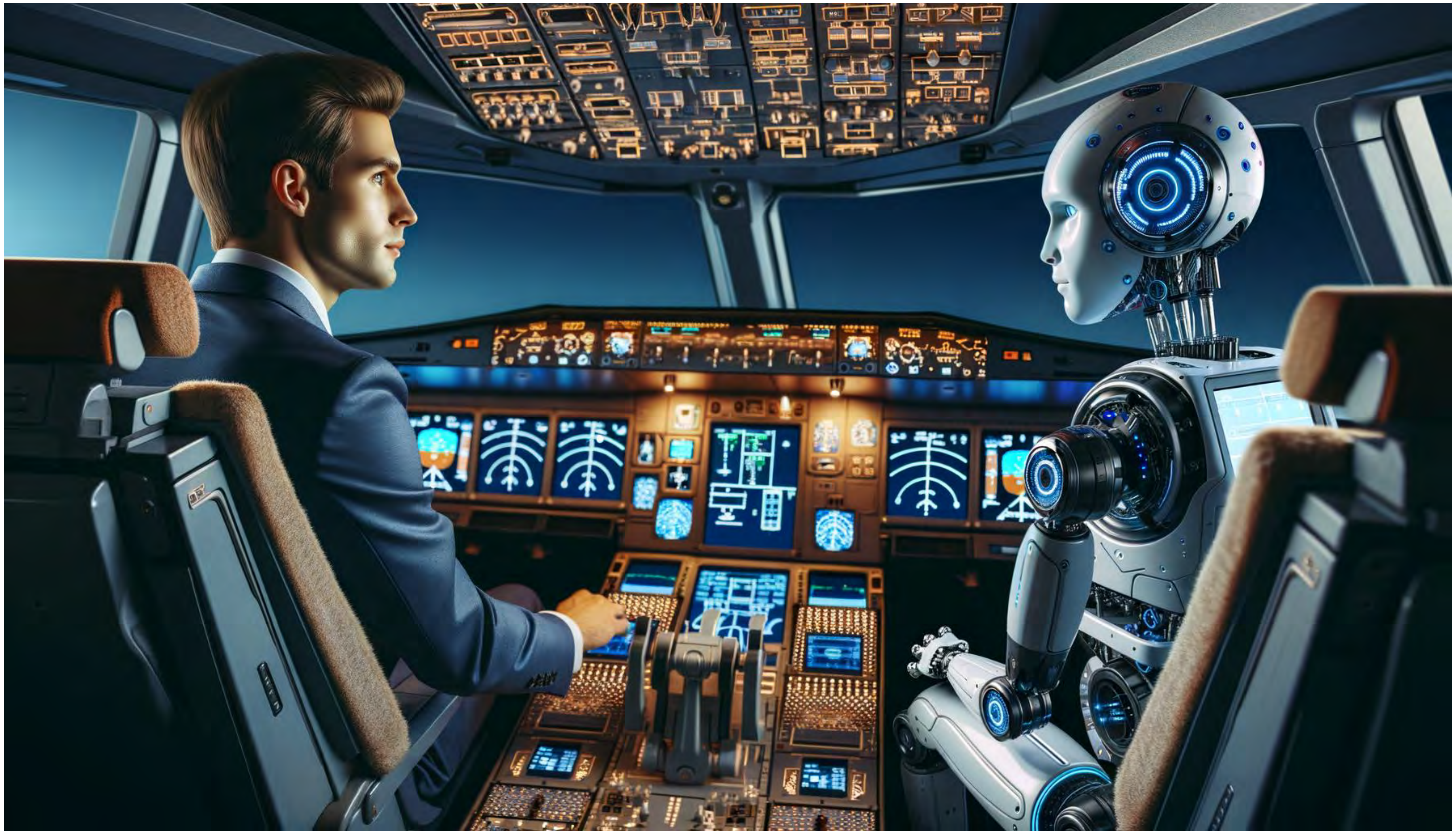
Michael Pisula
Marijn van Geest

2025-07-10

Stuttgart

From Copilot to Autopilot

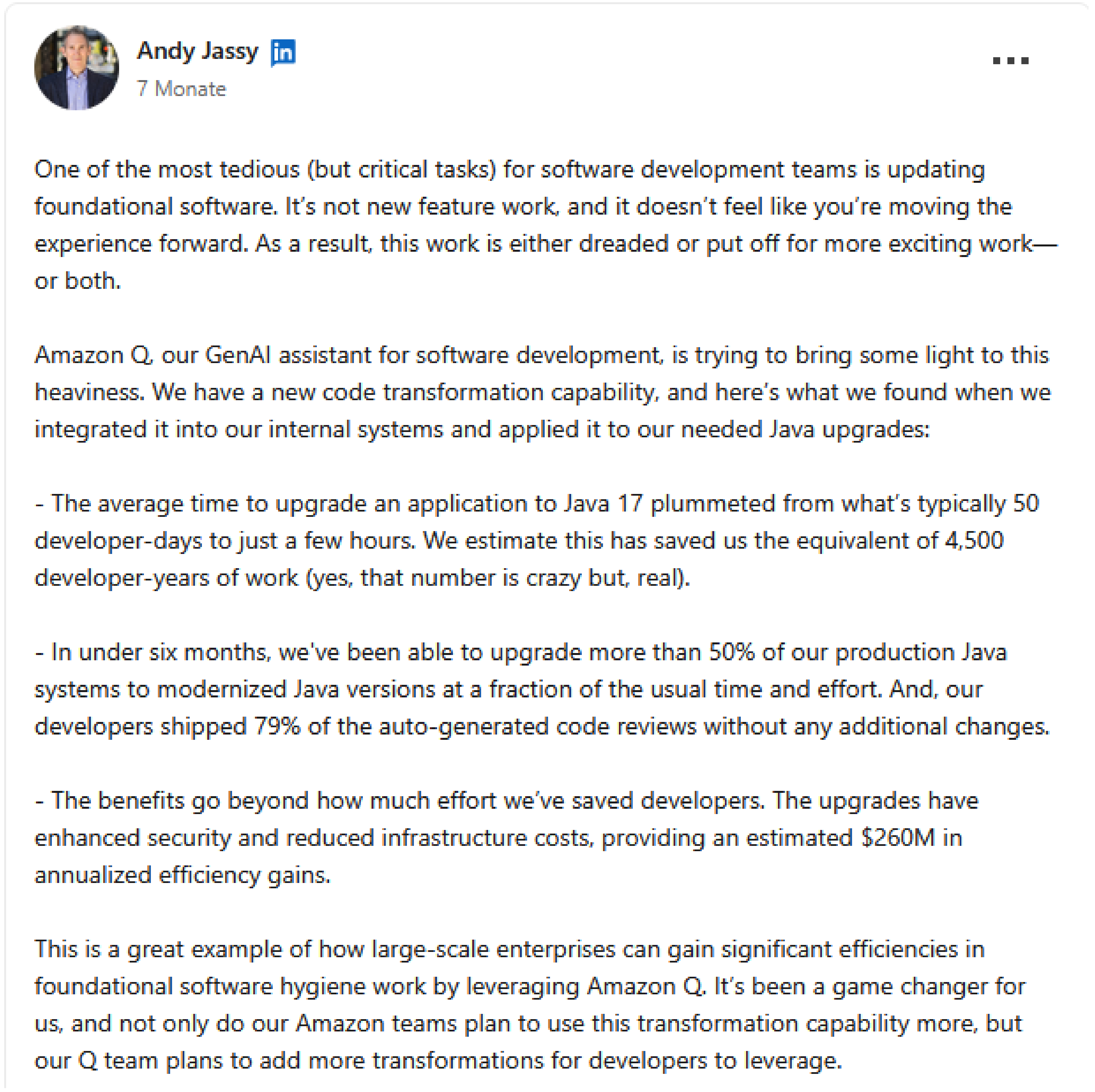
TNG TECHNOLOGY CONSULTING



TNG TECHNOLOGY CONSULTING

1 | Code Migration

AI Code Migration



Andy Jassy
7 Monate

One of the most tedious (but critical tasks) for software development teams is updating foundational software. It's not new feature work, and it doesn't feel like you're moving the experience forward. As a result, this work is either dreaded or put off for more exciting work—or both.

Amazon Q, our GenAI assistant for software development, is trying to bring some light to this heaviness. We have a new code transformation capability, and here's what we found when we integrated it into our internal systems and applied it to our needed Java upgrades:

- The average time to upgrade an application to Java 17 plummeted from what's typically 50 developer-days to just a few hours. We estimate this has saved us the equivalent of 4,500 developer-years of work (yes, that number is crazy but, real).

- In under six months, we've been able to upgrade more than 50% of our production Java systems to modernized Java versions at a fraction of the usual time and effort. And, our developers shipped 79% of the auto-generated code reviews without any additional changes.

- The benefits go beyond how much effort we've saved developers. The upgrades have enhanced security and reduced infrastructure costs, providing an estimated $260M in annualized efficiency gains.

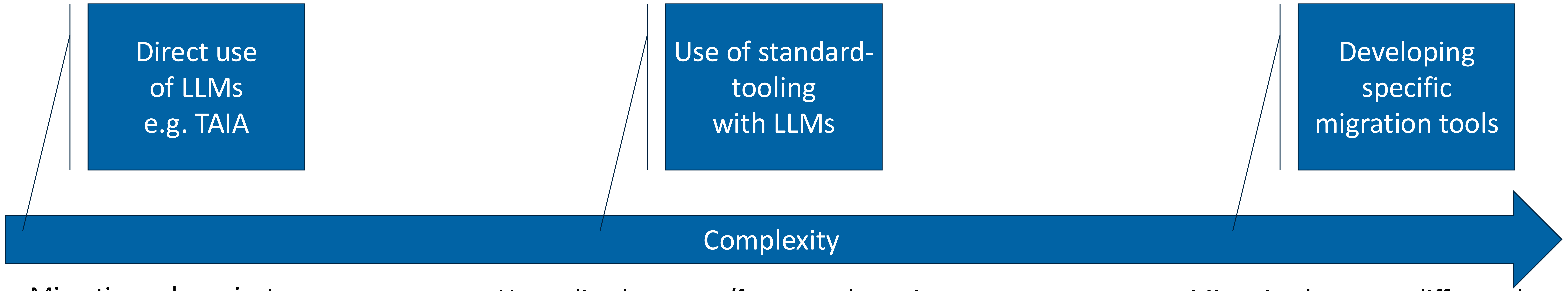
This is a great example of how large-scale enterprises can gain significant efficiencies in foundational software hygiene work by leveraging Amazon Q. It's been a game changer for us, and not only do our Amazon teams plan to use this transformation capability more, but our Q team plans to add more transformations for developers to leverage.

AI Code Migration

The average time to upgrade an application to Java 17 plummeted from what's typically 50 developer-days to just a few hours. We estimate this has saved us the equivalent of 4,500 developer-years of work (yes, that number is crazy but, real).

AI Code Migration

Different Approaches


Direct use of LLMs e.g. TAIA
Use of standard-tooling with LLMs
Developing specific migration tools
Complexity

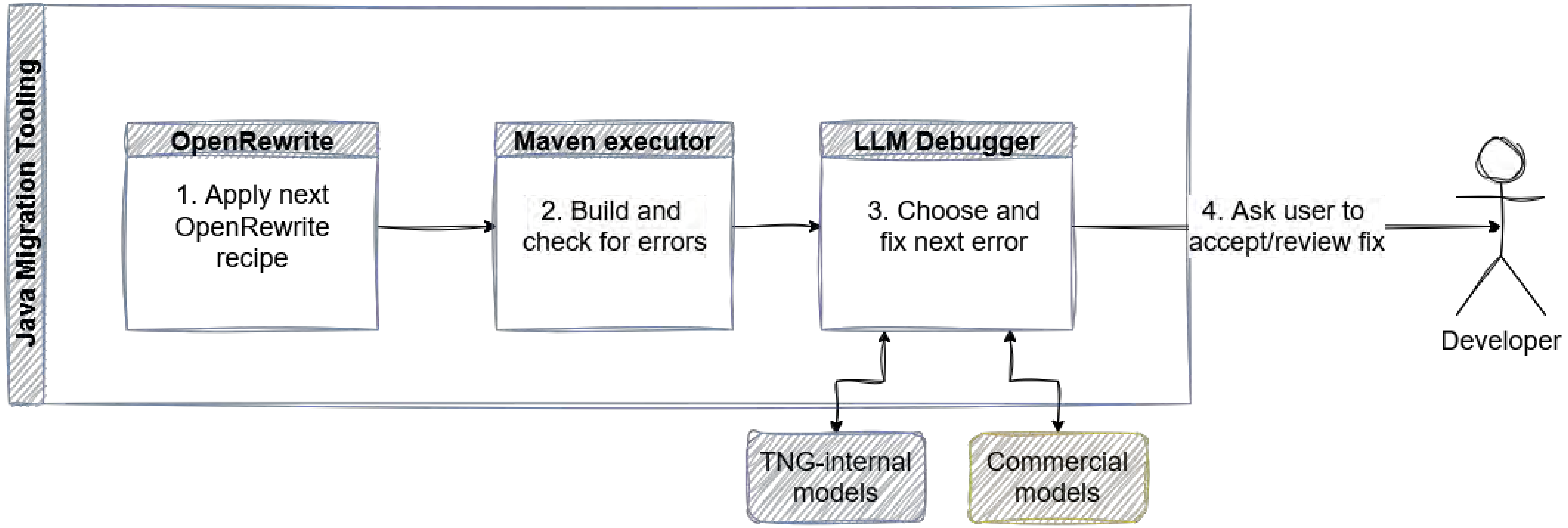
- Migrating sub-projects
- Evaluating LLM migration

- Upgrading language/framework versions
- Migrating mid-size to large projects
- Migrating modern programming languages, e.g. *Java 8* → *Java 17*

- Migrating between different languages
- Migrating of legacy languages, e.g. *Natural* → *Python*
- Specific target architectures

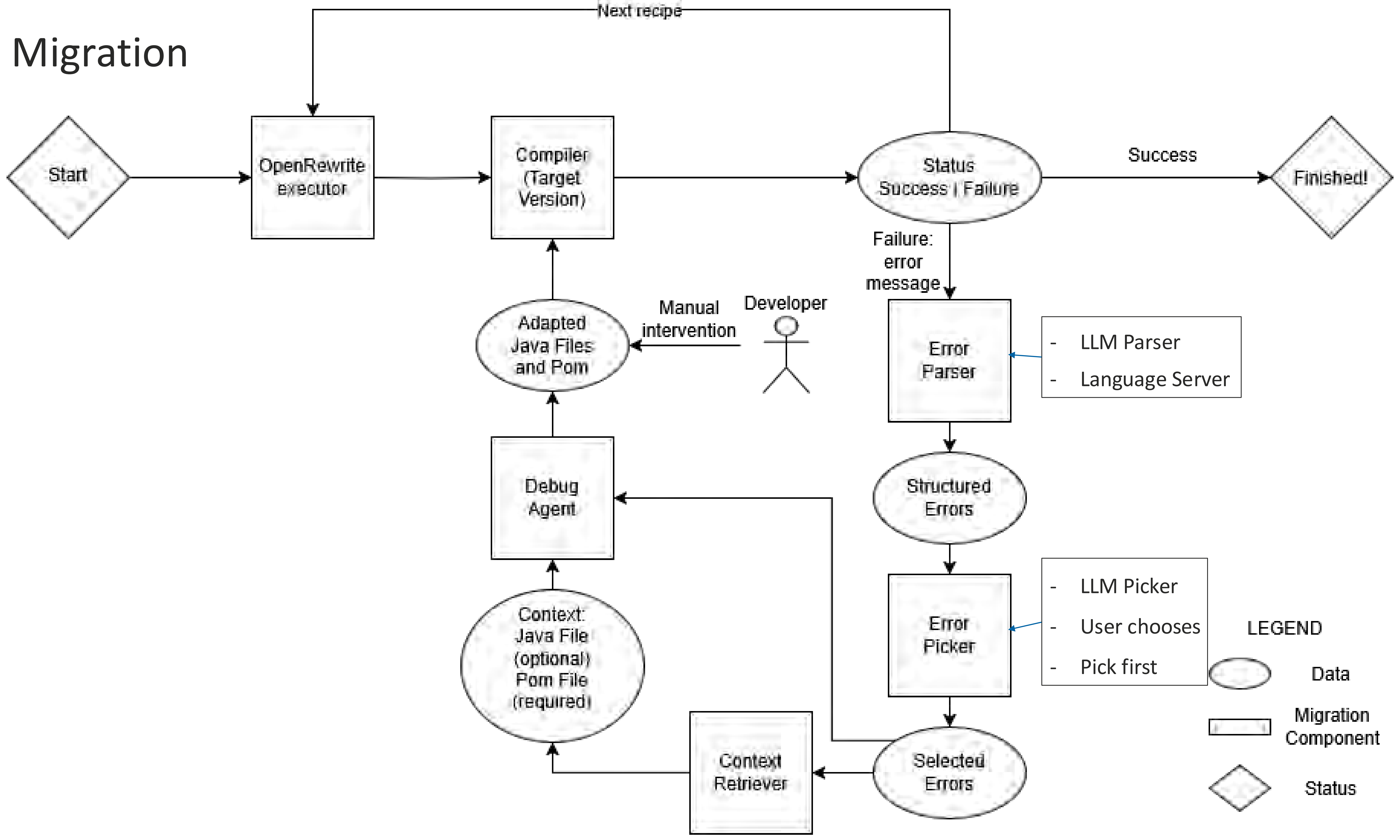
AI Code Migration

Java Migration


Java Migration Tooling
OpenRewrite
1. Apply next OpenRewrite recipe
Maven executor
2. Build and check for errors
LLM Debugger
3. Choose and fix next error
4. Ask user to accept/review fix
Developer
TNG-internal models
Commercial models

AI Code Migration

Java Migration



Java Migration

```
@Override
  public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
      if (object == null) {
         return null;
      }
      if (object instanceof Customer) {
         Customer o = (Customer) object;
         return getStringKey(o.getCustomerId());
      } else {
         return null;
      }
  }
```

OpenRewrite

```
@Override
public String getAsString(FacesContext facesContext, UIComponent
component, Object object) {
   if (object == null) {
      return null;
   }
   if (object instanceof Customer o) {
      return getStringKey(o.getCustomerId());
   } else {
      return null;
   }
}
```

Why OpenRewrite + AI?

```
public String base64decode(String text) {
   try {
      return new String(dec.decodeBuffer(text), DEFAULT_ENCODING);
   } catch (IOException e) {
      return null;
   }
}
```

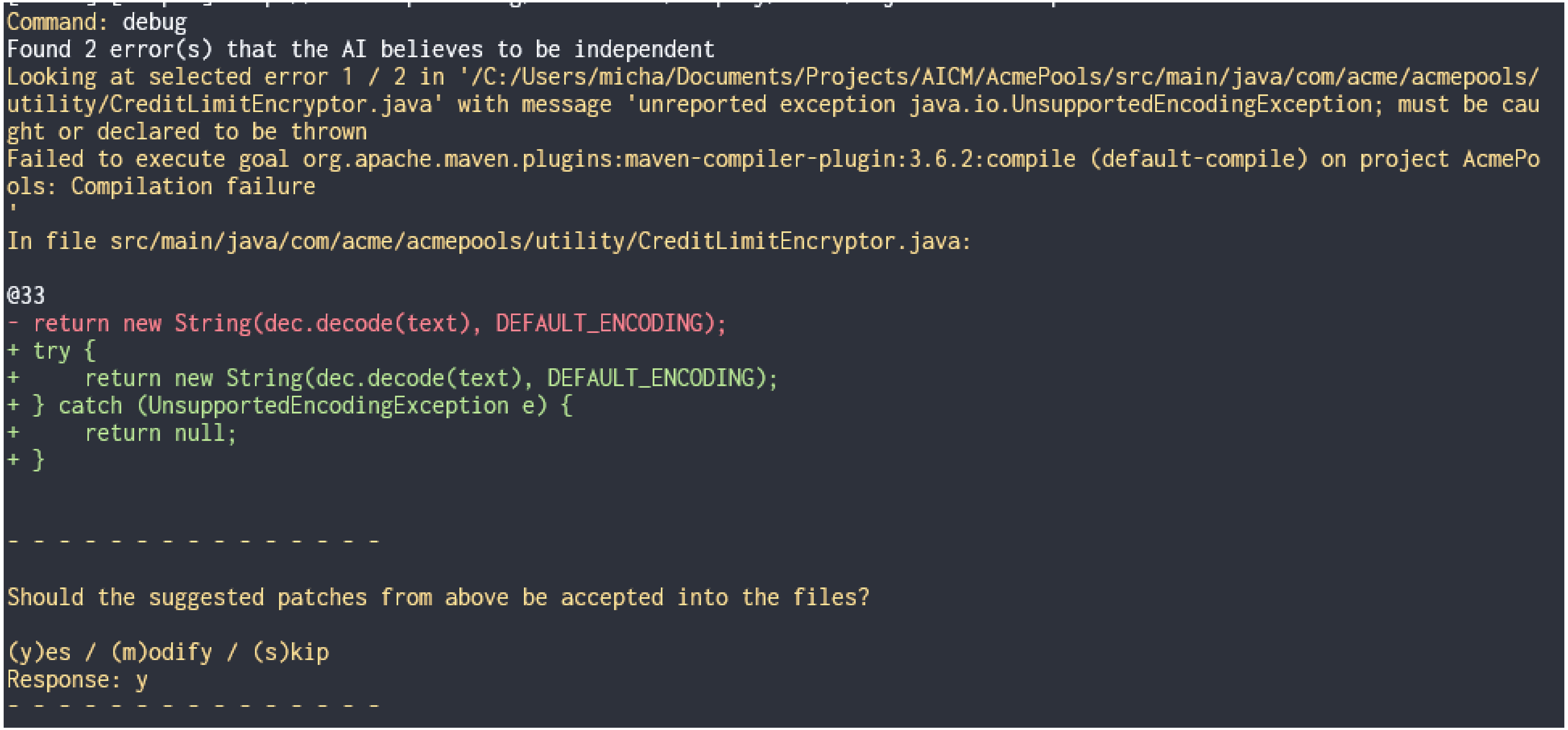
OpenRewrite

```
public String base64decode(String text) {
   return new String(dec.decode(text), DEFAULT_ENCODING);
}
```

Why OpenRewrite + AI?

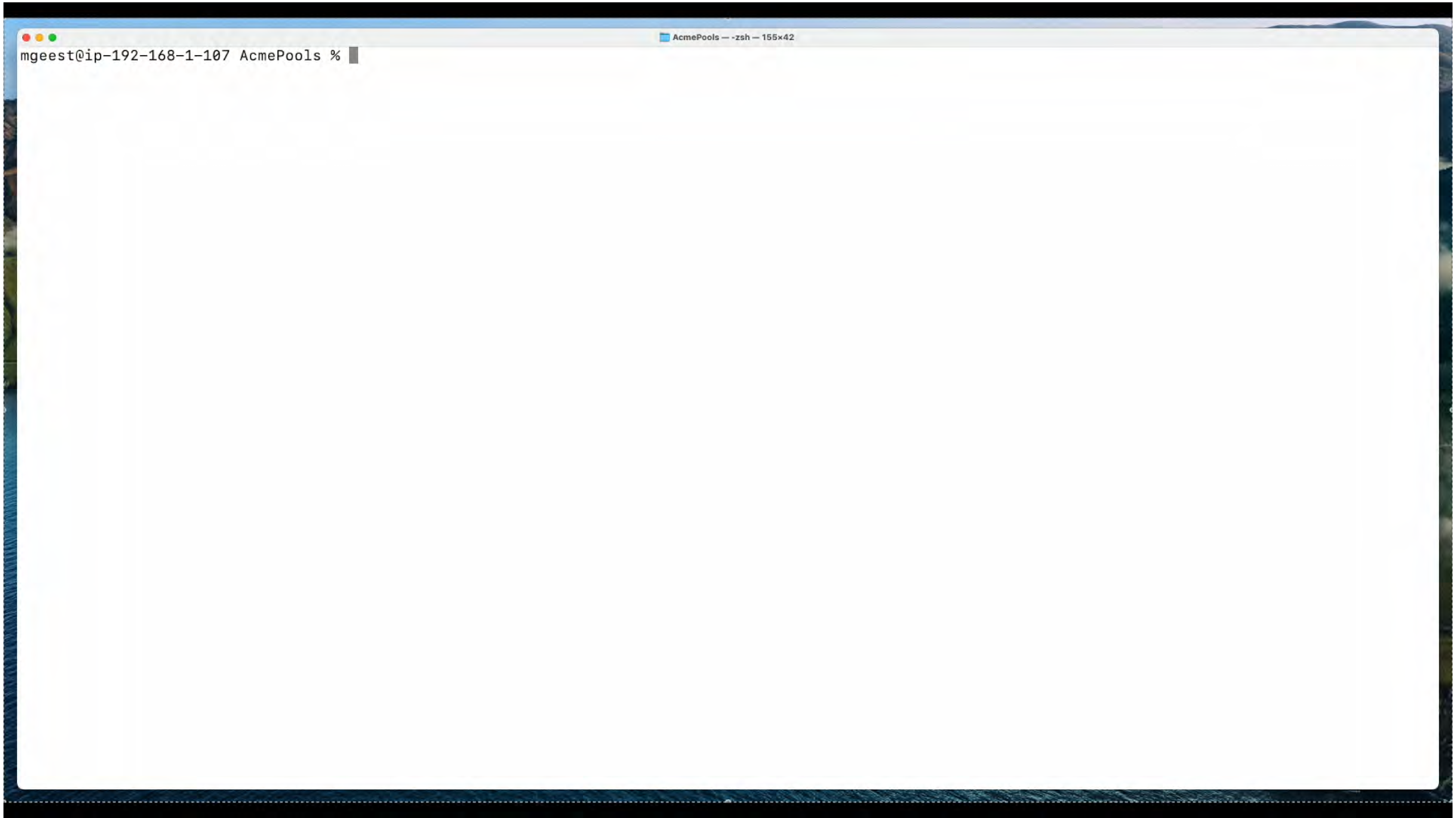
TNG TECHNOLOGY CONSULTING

LLM Fix



```
Command: debug
Found 2 error(s) that the AI believes to be independent
Looking at selected error 1 / 2 in '/C:/Users/micha/Documents/Projects/AICM/AcmePools/src/main/java/com/acme/acmepools/
utility/CreditLimitEncryptor.java' with message 'unreported exception java.io.UnsupportedEncodingException; must be cau
ght or declared to be thrown
Failed to execute goal org.apache.maven.plugins:maven-compiler-plugin:3.6.2:compile (default-compile) on project AcmePo
ols: Compilation failure
'
In file src/main/java/com/acme/acmepools/utility/CreditLimitEncryptor.java:

@33
- return new String(dec.decode(text), DEFAULT_ENCODING);
+ try {
+     return new String(dec.decode(text), DEFAULT_ENCODING);
+ } catch (UnsupportedEncodingException e) {
+     return null;
+ }



- - - - - - - - - - - - - - -


Should the suggested patches from above be accepted into the files?

(y)es / (m)odify / (s)kip
Response: y
- - - - - - - - - - - - - - -
```

Demo



```
mgeest@ip-192-168-1-107 AcmePools %
```

Improving Debug Context

Context Retrieval is further Expanded

- User-AI interaction allows context-aware chatting about an error
  - **User** can guide the AI into the right direction
  - **The AI** can explain in more detail what is wrong
- RAG based retrieval of known issues and how to solve them

- And more ideas...
  - Let the LLM decide what it wants to know
  - Dynamically search the internet for solutions

AI Rule Application

TNG TECHNOLOGY CONSULTING

Define Custome Migration Rules

- Feature to complement the rule-based approach with custom ai-based rules

```
{
  "id": 0,
  "detect": "\\bprint"
  "instruction": "Use logging instead of printing everywhere"
}
```

- AI uses language server to iteratively build its own context
- Rules will be applied concurrently

AI Rule Application

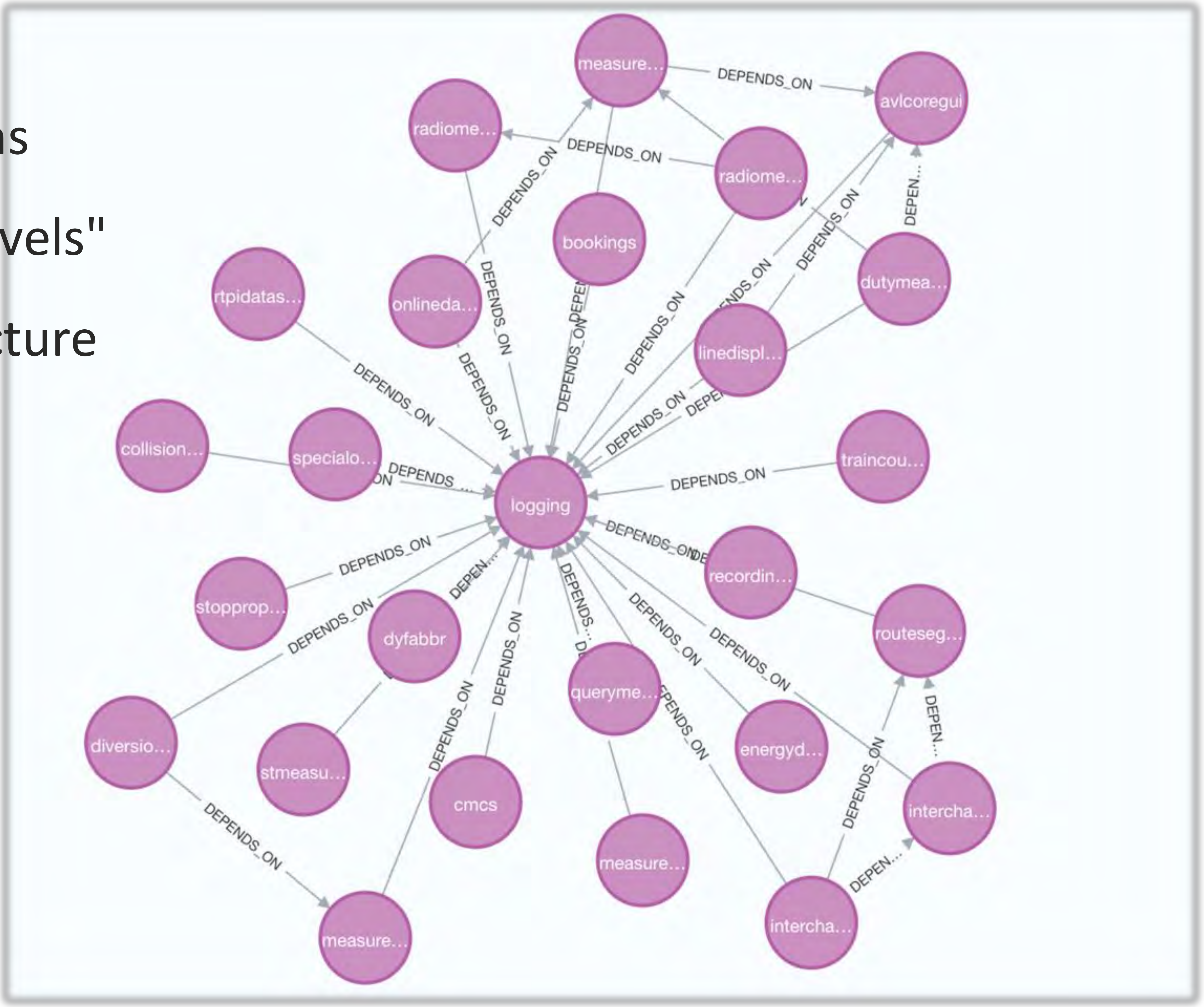
```java

import javax.annotation.PostConstruct;
import javax.validation.constraints.Max;
import javax.validation.constraints.Min;
import java.util.List;

@RestController("/foos")
public class FooController {

    @PostConstruct
    public void init(){
        System.out.println("test");
    }

    @Autowired
    private FooRepository repo;

    // API - read

    @GetMapping("/foos/{id}")
    @ResponseBody
    @Validated
    public Foo findById(@PathVariable @Min(0) final long id) {
        return repo.findById(id)
            .orElse(null);
```

```java
import javax.validation.constraints.Min;
import java.util.List;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@RestController("/foos")
public class FooController {

    private static final Logger logger = LoggerFactory.getLogger(FooController.class);

    @PostConstruct
    public void init(){
        logger.info("test");
    }

    @Autowired
    private FooRepository repo;

    // API - read

    @GetMapping("/foos/{id}")
    @ResponseBody
    @Validated
    public Foo findById(@PathVariable @Min(0) final long id) {
```

Migration Planning

In Multi-Module Projects

- Feature to plan correct sequence of module migrations
- Automatically sorts modules together in migration "levels"
- Based on graphical representation of the project structure



Migration Results

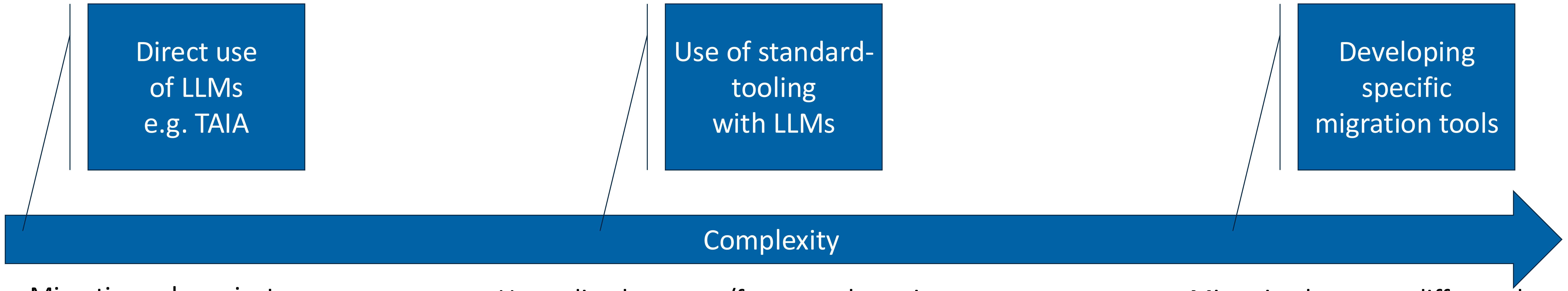
TNG TECHNOLOGY CONSULTING

Migration Project

- 600 Modules
- Estimated manual migration 2 years
- Two months, 1 developer
- Average migration: 30 minutes per module

AI Code Migration

Different Approaches


Direct use of LLMs e.g. TAIA
Use of standard-tooling with LLMs
Developing specific migration tools
Complexity

- Migrating sub-projects
- Evaluating LLM migration

- Upgrading language/framework versions
- Migrating mid-size to large projects
- Migrating modern programming languages, e.g. *Java 8* → *Java 17*

- Migrating between different languages
- Migrating of legacy languages, e.g. *Natural* → *Python*
- Specific target architectures

Complex Migrations

Migration PL/I → Java

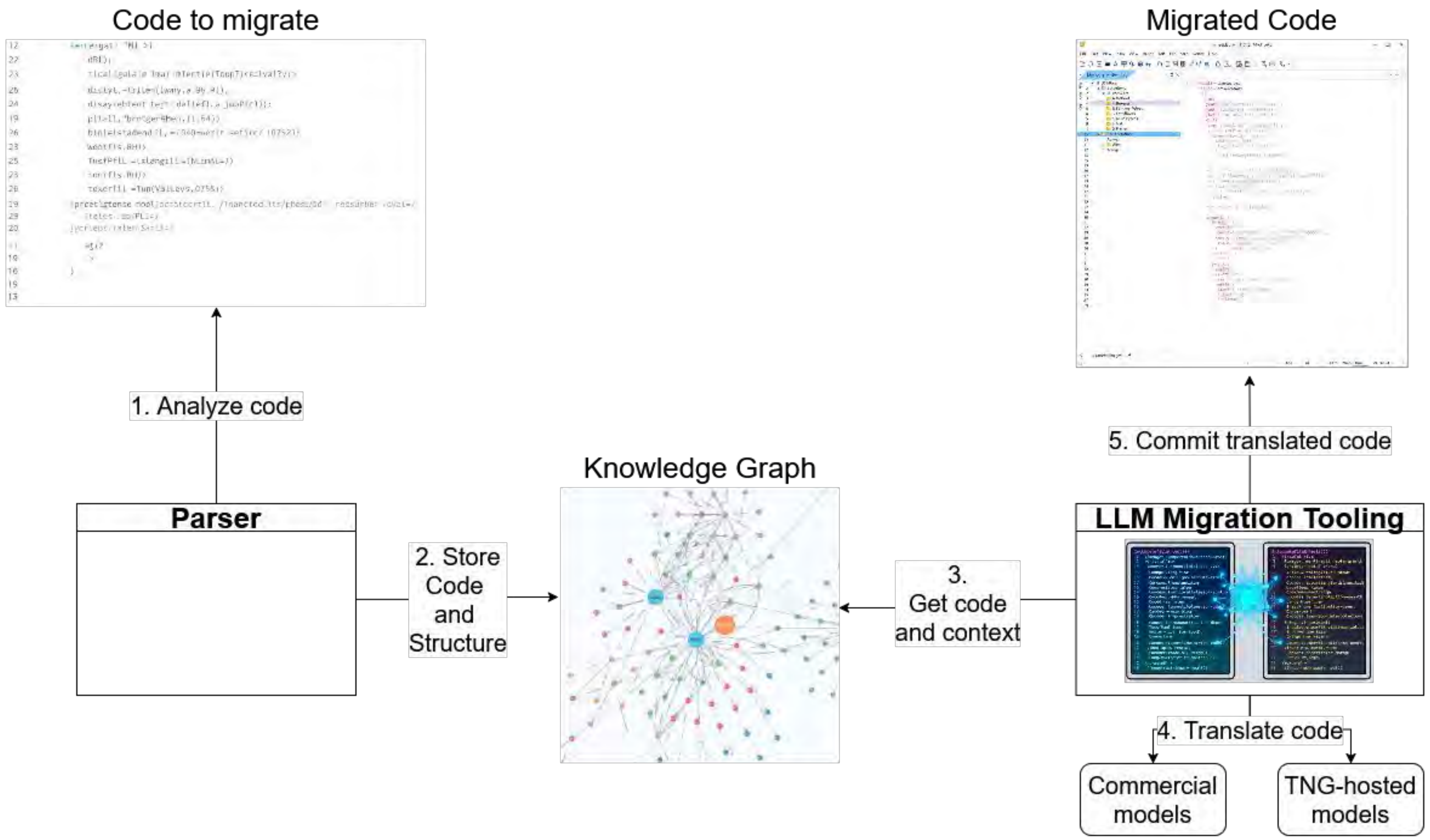
- Create tooling to assist developers migrating PL/1 into specific target Java framework
- Custom PL/I parser to extract system information
- Support for client developers via IntelliJ plugin
- Quotes of client developers:
  - „Very nice code“
  - „Looks like code which I would commit“
  - „When can we have it?“

Multiple million LOC total

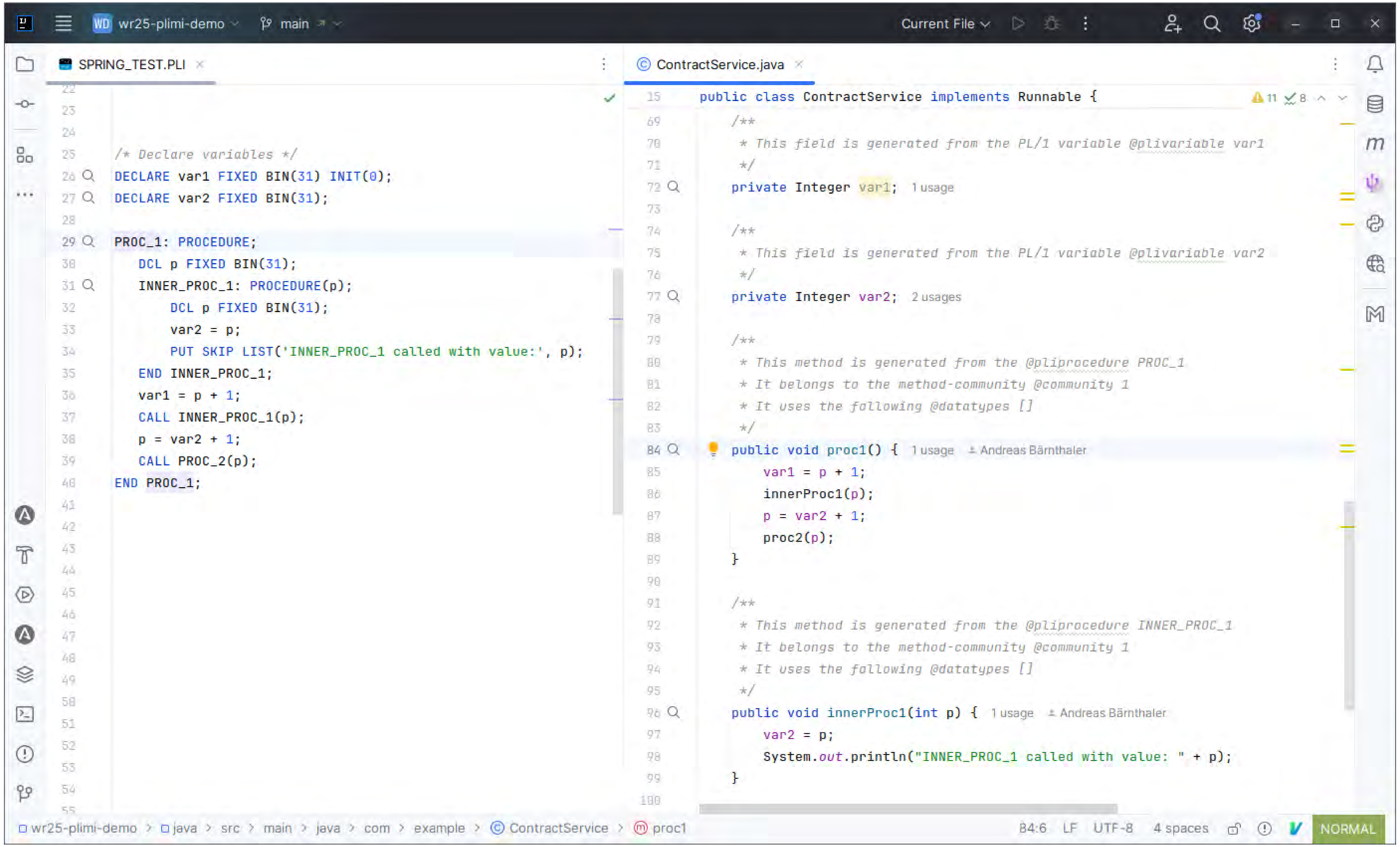
Pilot phase, extended to 1 year

Insurance industry

AI Code Migration



PL/I → Java Migration


wr25-plimi-demo
main
Current File
SPRING_TEST.PLI
ContractService.java
wr25-plimi-demo › java › src › main › java › com › example › ContractService › proc1
84:6 LF UTF-8 4 spaces NORMAL

```
/* Declare variables */
DECLARE var1 FIXED BIN(31) INIT(0);
DECLARE var2 FIXED BIN(31);

PROC_1: PROCEDURE;
   DCL p FIXED BIN(31);
   INNER_PROC_1: PROCEDURE(p);
       DCL p FIXED BIN(31);
       var2 = p;
       PUT SKIP LIST('INNER_PROC_1 called with value:', p);
   END INNER_PROC_1;
   var1 = p + 1;
   CALL INNER_PROC_1(p);
   p = var2 + 1;
   CALL PROC_2(p);
END PROC_1;
```

```
public class ContractService implements Runnable {
    /**
     * This field is generated from the PL/1 variable @plivariable var1
     */
    private Integer var1;  1 usage

    /**
     * This field is generated from the PL/1 variable @plivariable var2
     */
    private Integer var2;  2 usages

    /**
     * This method is generated from the @pliprocedure PROC_1
     * It belongs to the method-community @community 1
     * It uses the following @datatypes []
     */
    public void proc1() {  1 usage  Andreas Bärnthaler
        var1 = p + 1;
        innerProc1(p);
        p = var2 + 1;
        proc2(p);
    }

    /**
     * This method is generated from the @pliprocedure INNER_PROC_1
     * It belongs to the method-community @community 1
     * It uses the following @datatypes []
     */
    public void innerProc1(int p) {  1 usage  Andreas Bärnthaler
        var2 = p;
        System.out.println("INNER_PROC_1 called with value: " + p);
    }
```

Frontend migration

TNG TECHNOLOGY CONSULTING

- Migrating code between frontend frameworks is complex
- The architecture changes between the old and new code

AI Code Migration

Insights

- Human expertise is key for pre-analysis and review
- Migrating smaller pieces of code leads to better results
- Splitting the migration into separate steps improves the result
- Right models for right tasks
- A separate post-processing step can leverage UI to further increase code quality
- Tests are important
- Generating unit tests is a good complementary step

Thank you for your attention!

Michael Pisula
Principal Consultant
michael.pisula@tngtech.com

Marijn van Geest
Software Consultant
marijn.vangeest@tngtech.com