Consultingwerk
software architecture and development

AI-Powered Modernization for OpenEdge Applications

Mike Fechner
Consultingwerk

The full stack modernization framework
SmartComponent Library
by Consultingwerk

Mike Fechner

35 years of experience in Progress (from Version 5 to OpenEdge 12)

Active member of the OpenEdge community and speaker at international conferences

Expert in software architecture, object-oriented design, and web technologies

Modernization in Focus

Modernization of Legacy OpenEdge Applications

Deep Technical Expertise

Global IT Partner with Local Presence

More than Consulting – We Deliver Tools & Solutions

SmartComponent Library

Accelerated OpenEdge Modernization with a Full-Stack Framework

Full-stack modernization framework with a robust architectural foundation

Future-proof backend architecture as the central home for business logic and architectural base

Support for relational and object-relational (ORM) data models

Comprehensive **application framework** including: Authentication, localization, dynamic menus, workflow management, etc.

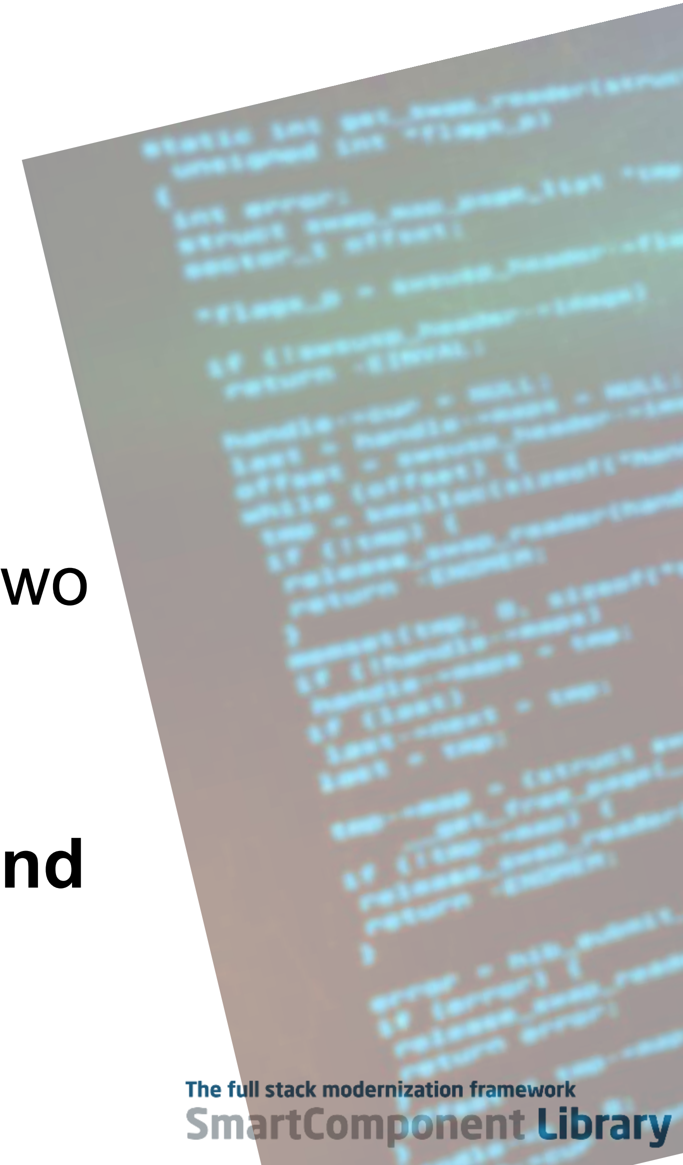
Integrated RESTful functionality – ready for modern interfaces

Versatile **user interfaces**: desktop, web, and mobile

Seamless **integration** into existing OpenEdge applications and frameworks

At a Glance

- **No Lock-In:** Full source code is delivered to our customers
- **Future-Proof:** Always one step ahead of customer requirements
- **Designed for Customization and Extensibility:** No two customers have exactly the same needs
- **DevOps-Ready:** Templates for build, test & deploy
- **Supported on OpenEdge 11.7 (phasing out), 12.2 and 12.8**
- **Support and Maintenance Services**



Application Backend

OpenEdge-Centric Architecture

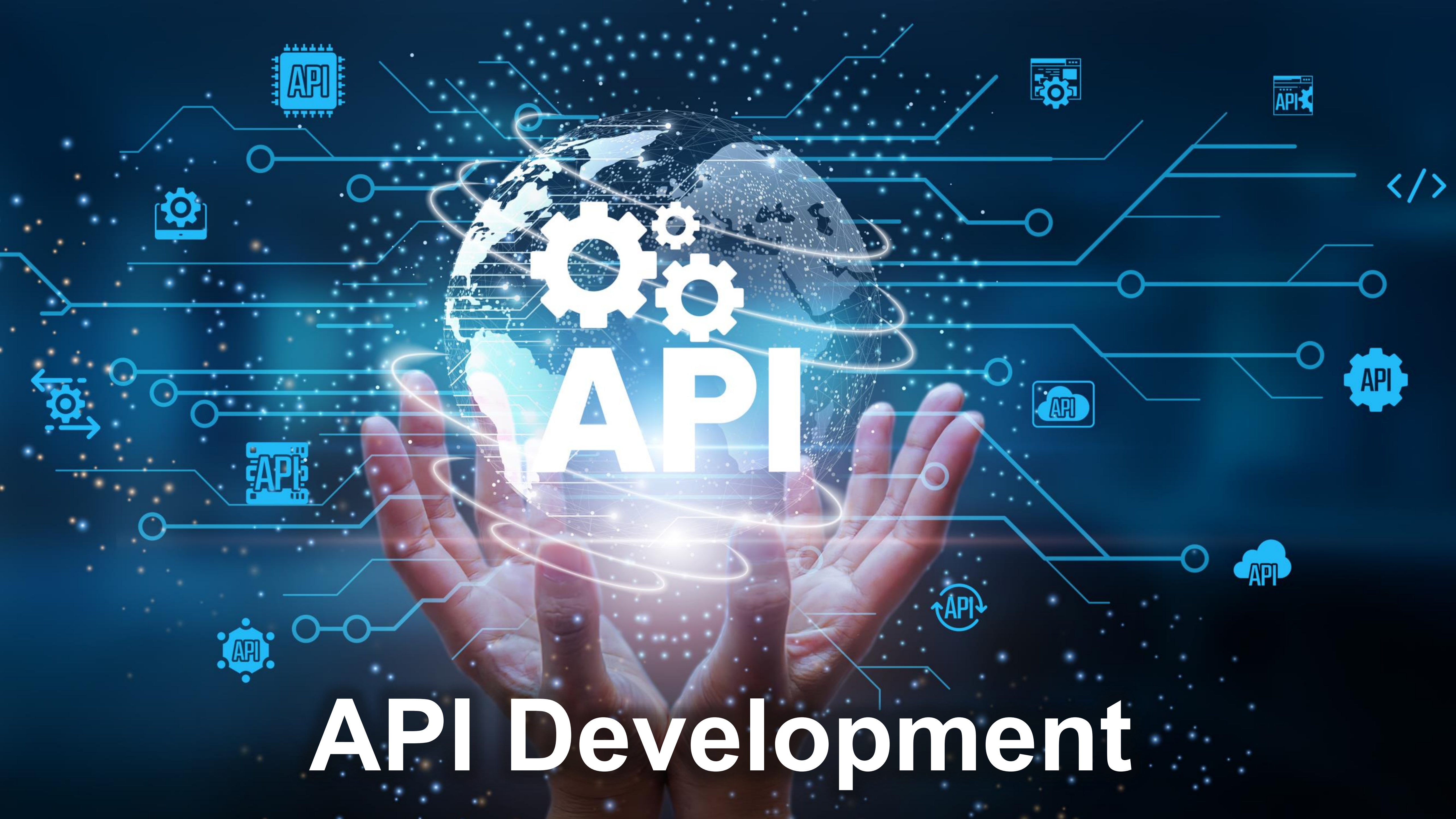
- Designed for the Progress Application Server (PASOE)
- Supports a wide range of deployment scenarios:
    - Physical servers
    - Virtual servers
    - Cloud-hosted environments
    - Docker/Kubernetes
- Multi-Tenancy / Multiple instances / Company key in DB table
- Client options: Fat client GUI application or web client or browser client or mobile

Frontend Options

- OpenEdge GUI for .NET (SmartComponent Library)
- Angular Web UI with Telerik Kendo UI (SmartComponent Library)
- Headless Applications
- Partner User Interfaces

BUILD.ONE

RAPID CLOUD
DEVELOPMENT



API Development

REST/RESTful

- Standard protocol for application integration and UI flexibility
- SmartComponent Library provides the **simplest and most flexible** way of implementing **RESTful services with OpenEdge**
- Typical use-cases
  - Implement new functionality as RESTful services
  - Provide existing (legacy) functionality as RESTful service
- Open API / Swagger documentation / test suite out of the box, generated automatically
- No need to deploy services, code declares the API
- API-first design – implement service based on Open API specification

AI Coding Assistants for OpenEdge accelerate development and refactoring with AI-powered assistance.

GenAI

GenAI in Software development

- Supporting of developers while writing code
- More intelligent proposals while coding
- Chat: „write me a method that solves XYZ “, „write me a unit test for this method “), Boiler Plate Code – *Prompts*, the new kind of wizards
- Refactoring of Code
- Explanation of Code, proposal for comments
- Understanding of the context
  - Programming languages
  - Frameworks
  - Application model / Terms, e.g. Database schema

Gen AI in Application Modernization

- Currently the focus of our R&D
- Interpretation of existing code
- Transition to new application architecture
- Separation of UI and backend code, refactoring of source code

- Combination of Windsurf / Cascade / Devin with Proparse and XREF
- Proparse and compiler XREF are deterministic
- MCP Protocol – Model Context Protocol, Tools and Resources

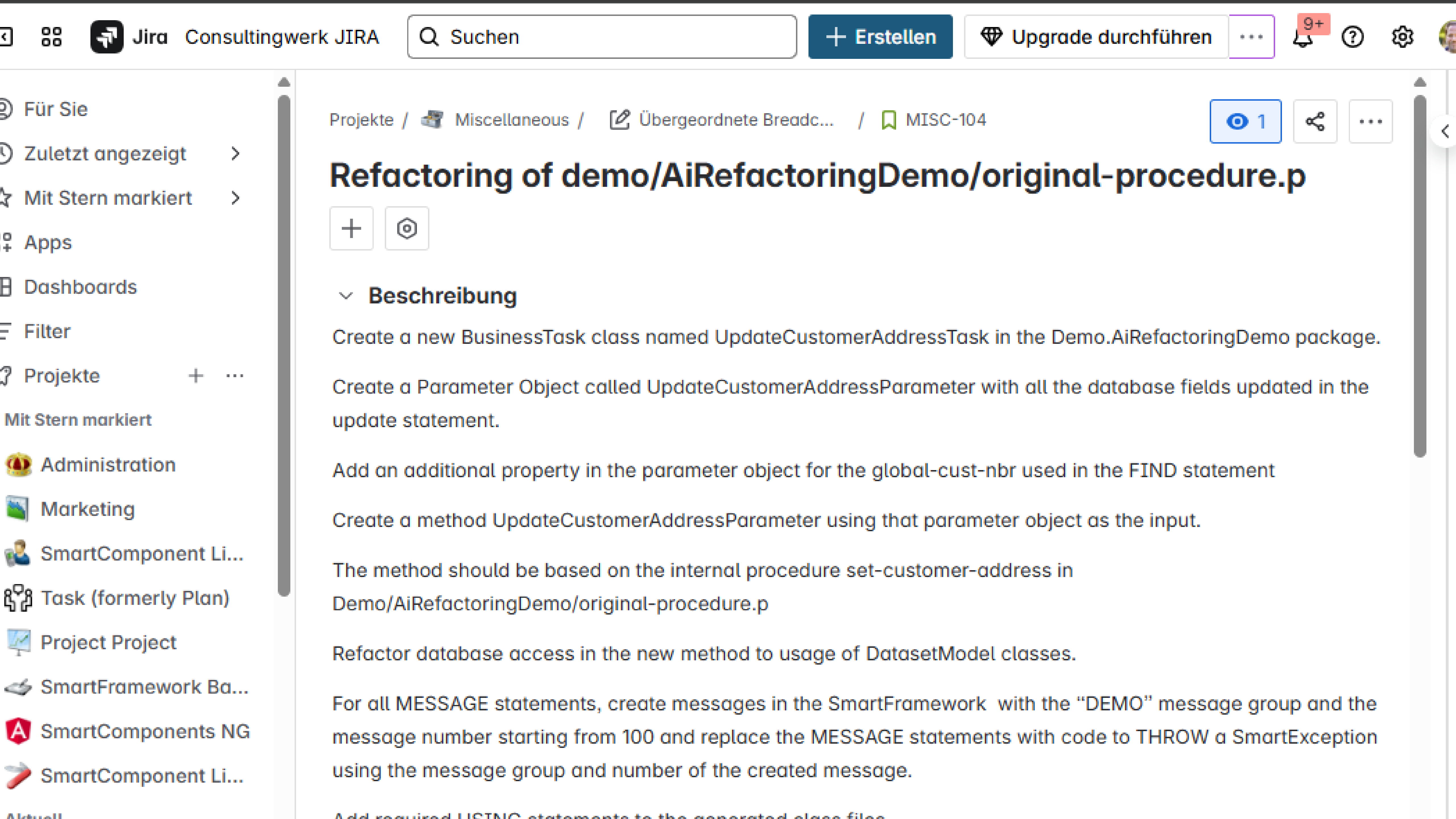
Using Gen AI for Refactoring

- Coding Assistants great in understanding existing source code
- ABL programs typically very verbose
- Refactoring – building ***new*** programs
  - using existing source code as part of the requirements
  - using new patterns, API’s, architecture
  - separate UI and business logic
  - separate domain logic and application logic
  - de-duplication of logic

Refactoring Demo 1

- Sample sample based on real world code of a customer
  - slightly simplified and translated to Sports2000
- Legacy code
- Demo video
- Refactored code

„Look at JIRA ticket MISC-104 and implement as required”



Refactoring of demo/AiRefactoringDemo/original-procedure.p

Beschreibung

Create a new BusinessTask class named UpdateCustomerAddressTask in the Demo.AiRefactoringDemo package.

Create a Parameter Object called UpdateCustomerAddressParameter with all the database fields updated in the update statement.

Add an additional property in the parameter object for the global-cust-nbr used in the FIND statement

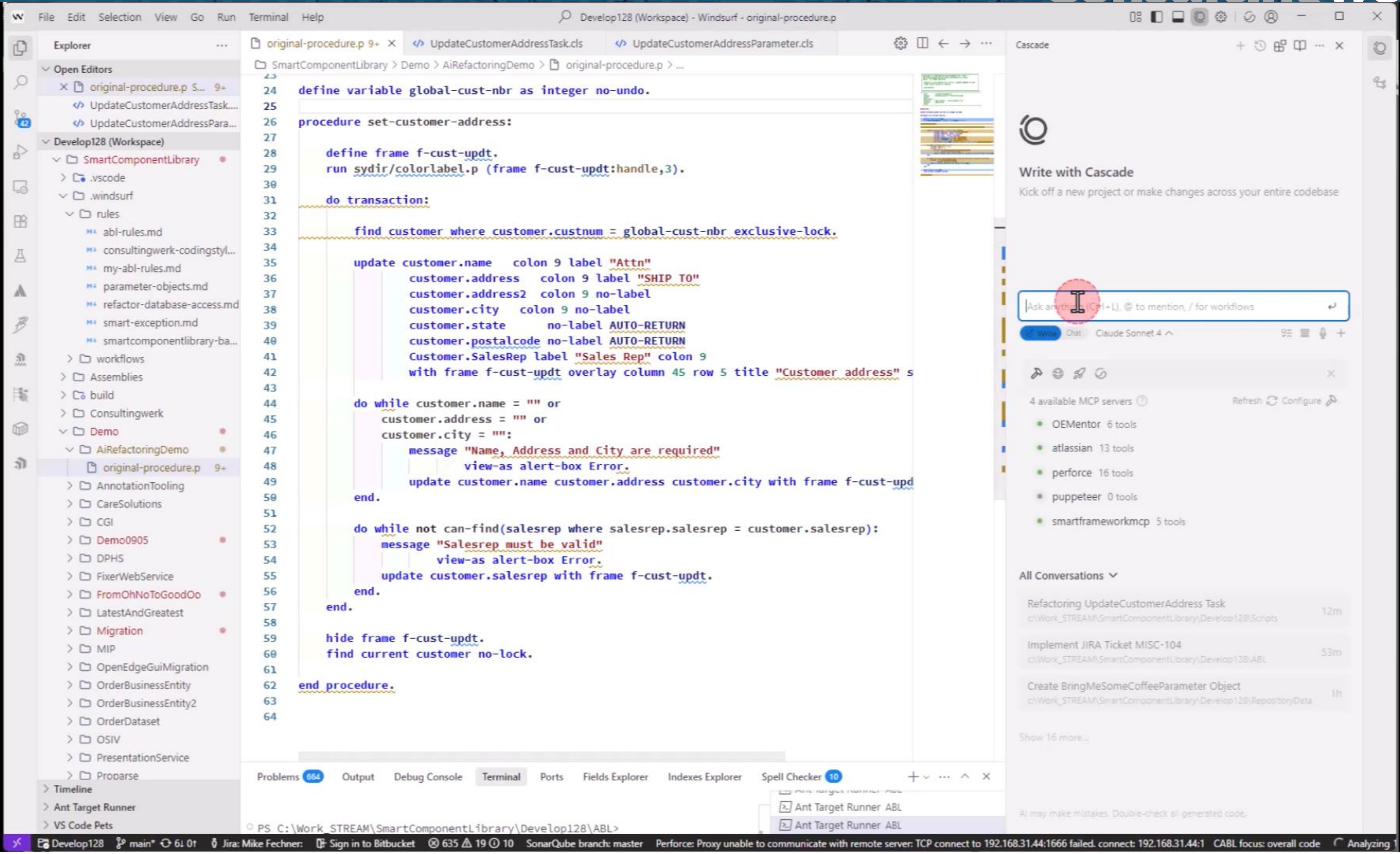
Create a method UpdateCustomerAddressParameter using that parameter object as the input.

The method should be based on the internal procedure set-customer-address in Demo/AiRefactoringDemo/original-procedure.p

Refactor database access in the new method to usage of DatasetModel classes.

For all MESSAGE statements, create messages in the SmartFramework with the “DEMO” message group and the message number starting from 100 and replace the MESSAGE statements with code to THROW a SmartException using the message group and number of the created message.

Add required USING statements to the generated class files.


File
Edit
Selection
View
Go
Run
Terminal
Help
Develop128 (Workspace) - Windsurf - original-procedure.p
Explorer
original-procedure.p 9+
UpdateCustomerAddressTask.cls
UpdateCustomerAddressParameter.cls
Cascade

SmartComponentLibrary > Demo > AiRefactoringDemo > original-procedure.p > ...

Open Editors
- original-procedure.p S... 9+
- UpdateCustomerAddressTask...
- UpdateCustomerAddressPara...

Develop128 (Workspace)
- SmartComponentLibrary
  - .vscode
  - .windsurf
    - rules
      - abl-rules.md
      - consultingwerk-codingstyl...
      - my-abl-rules.md
      - parameter-objects.md
      - refactor-database-access.md
      - smart-exception.md
      - smartcomponentlibrary-ba...
    - workflows
  - Assemblies
  - build
  - Consultingwerk
  - Demo
    - AiRefactoringDemo
      - original-procedure.p 9+
    - AnnotationTooling
    - CareSolutions
    - CGI
    - Demo0905
    - DPHS
    - FixerWebService
    - FromOhNoToGoodOo
    - LatestAndGreatest
    - Migration
    - MIP
    - OpenEdgeGuiMigration
    - OrderBusinessEntity
    - OrderBusinessEntity2
    - OrderDataset
    - OSIV
    - PresentationService
    - Proparse

Timeline
Ant Target Runner
VS Code Pets

```
define variable global-cust-nbr as integer no-undo.

procedure set-customer-address:

    define frame f-cust-updt.
    run sydir/colorlabel.p (frame f-cust-updt:handle,3).

    do transaction:

        find customer where customer.custnum = global-cust-nbr exclusive-lock.

        update customer.name   colon 9 label "Attn"
                customer.address   colon 9 label "SHIP TO"
                customer.address2  colon 9 no-label
                customer.city   colon 9 no-label
                customer.state      no-label AUTO-RETURN
                customer.postalcode no-label AUTO-RETURN
                Customer.SalesRep label "Sales Rep" colon 9
                with frame f-cust-updt overlay column 45 row 5 title "Customer address" s

        do while customer.name = "" or
            customer.address = "" or
            customer.city = "":
                message "Name, Address and City are required"
                        view-as alert-box Error.
                update customer.name customer.address customer.city with frame f-cust-upd
        end.

        do while not can-find(salesrep where salesrep.salesrep = customer.salesrep):
            message "Salesrep must be valid"
                    view-as alert-box Error.
            update customer.salesrep with frame f-cust-updt.
        end.
    end.

    hide frame f-cust-updt.
    find current customer no-lock.

end procedure.
```

```
PS C:\Work_STREAM\SmartComponentLibrary\Develop128\ABL>
```

Ant Target Runner ABL
Ant Target Runner ABL

Write with Cascade

Kick off a new project or make changes across your entire codebase

Ask anything (Ctrl+L), @ to mention, / for workflows

4 available MCP servers

- OEMentor 6 tools
- atlassian 13 tools
- perforce 16 tools
- puppeteer 0 tools
- smartframeworkmcp 5 tools

All Conversations

- Refactoring UpdateCustomerAddress Task c:\Work_STREAM\SmartComponentLibrary\Develop128\Scripts 12m
- Implement JIRA Ticket MISC-104 c:\Work_STREAM\SmartComponentLibrary\Develop128\ABL 53m
- Create BringMeSomeCoffeeParameter Object c:\Work_STREAM\SmartComponentLibrary\Develop128\RepositoryData 1h

Show 16 more...

AI may make mistakes. Double-check all generated code.

Develop128 main* 6↓ 0↑ Jira: Mike Fechner: Sign in to Bitbucket ⊗ 635 ⚠ 19 ⓘ 10 SonarQube branch: master Perforce: Proxy unable to communicate with remote server: TCP connect to 192.168.31.44:1666 failed. connect: 192.168.31.44:1 CABL focus: overall code Analyzing 1

Windsurf customization options

- Rules
  - markdown documents
  - mix of natural language and code samples
- Workflows
- MCP Servers

- Similar customization options available in other AI coding assistants (Github Copilot, Cursor, Claude Code, Devin, …)

MCP – Model Context Protocol

- Standard for tools plugged into AI agents, introduced by Anthropic (Claude) end of 2024, *the USB-C standard of AI*
- Large adoption already, just 10 month later
- Defines integration of
  - **Prompts** are pre-defined templates for the LLM that can be selected by the user through slash commands, menu options, and the like
  - **Resources** are structured data, like files, data from a database, or a commit history that provide additional context to the LLM
  - **Tools** are functions that allow the model to take action, like interacting with an API or writing something to a file

MCP Servers to improve AI experience

- AI is *not deterministic* – that means not consistent
- You give AI two times the same task – and it may provide a different solution each time
- Solutions may (should) all be correct
- But inconsistency makes code validation and maintenance harder (for human and AI)
- MCP Serves can query and update framework database (repository)

MCP Servers to implement critical design patterns

- SmartComponent Library is providing (customizable and proven) code generators since long time
- Business Entity with all bells and whistles and ORM mapper can be ~ *20 class files* following a certain structure
- Using MCP server to generate Business Entities combines best of both worlds
  - Gen AI capabilities to analyze legacy code
  - Deterministic ABL based code generator for generating Business Entity skeleton – not playing Russian Roulette with AI on that
  - Gen AI to implement custom rules (e.g. validation) in Business Entity

SmartComponent Library MCP Services

- Code Generation
- Cross Reference information (e.g. Business Entity or ORM mapper used for database tables)
- User Interface Repository Management
- Meta-Data Management (e.g. Error/Warning Messages)
- Access to RI-Information

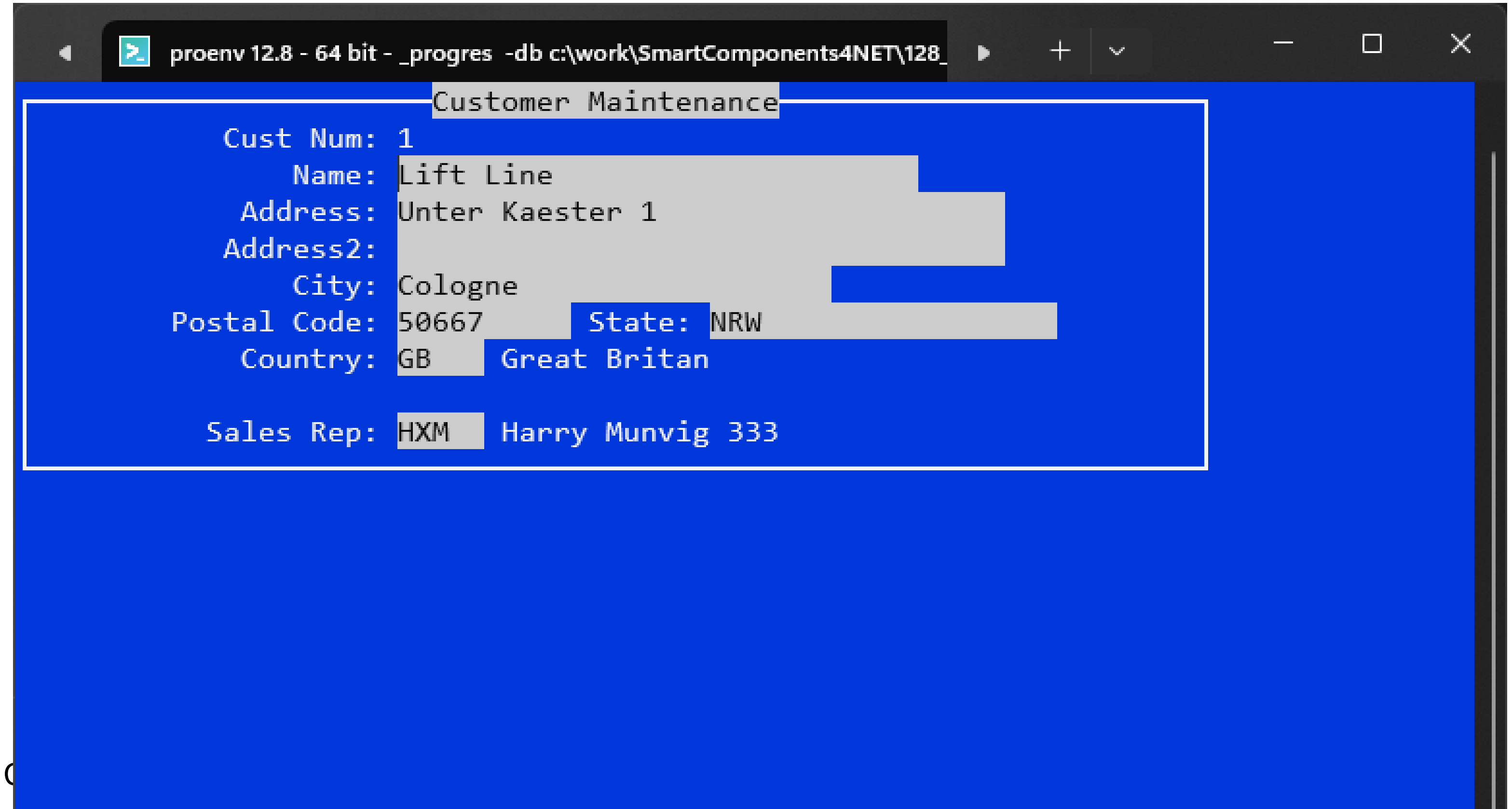
Demo

- SmartFramework MCP Server
- Anthropic MCP inspector

- *“Add method HelloAiAudience to the logic object of the CustomerViewer in the repository.”*

Refactoring Demo 2

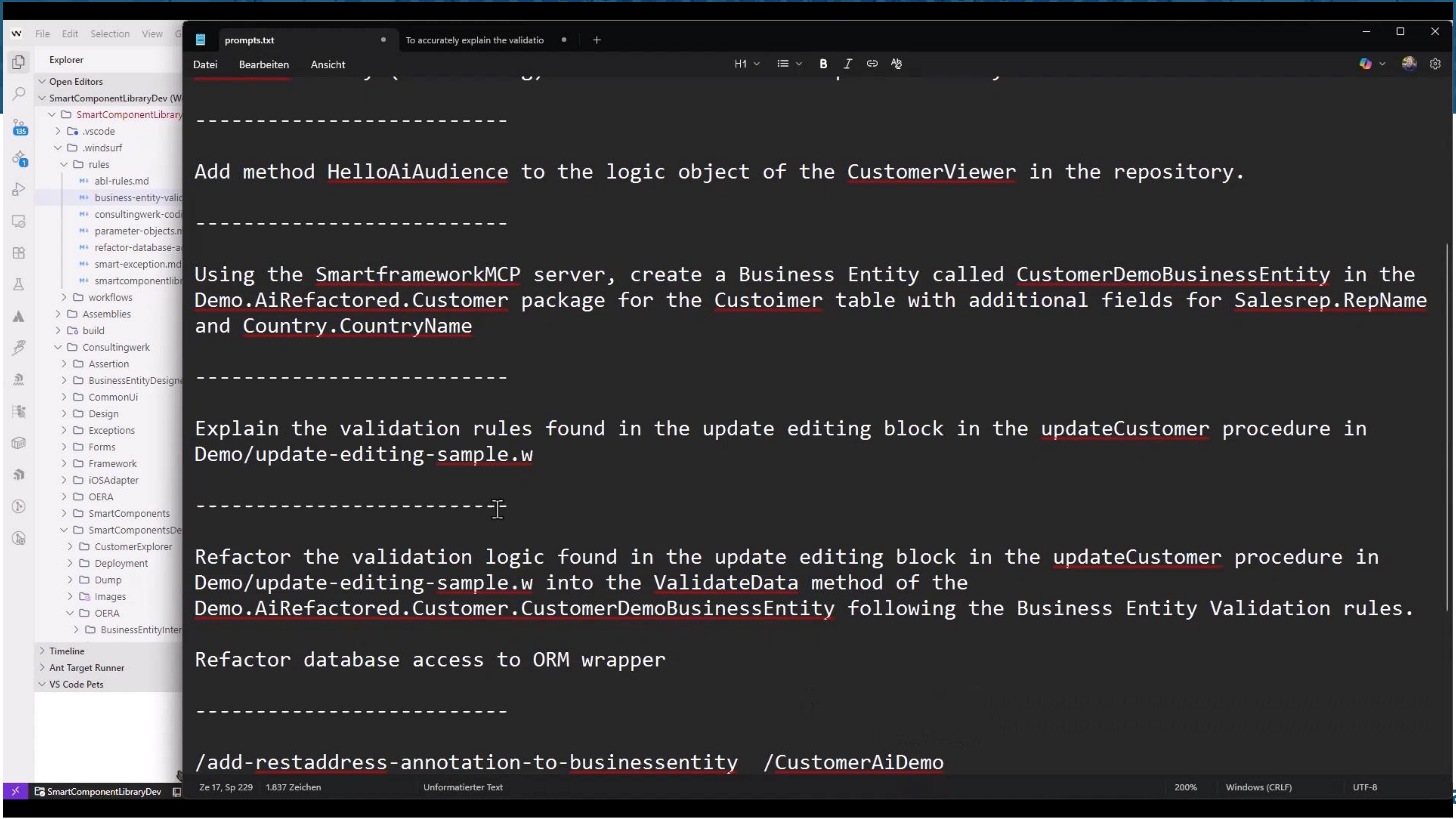
- Migration of TTY style application with UPDATE EDITING Block


proenv 12.8 - 64 bit - _progres -db c:\work\SmartComponents4NET\128_

```
                    Customer Maintenance
          Cust Num: 1
              Name: Lift Line
           Address: Unter Kaester 1
          Address2:
              City: Cologne
       Postal Code: 50667      State: NRW
           Country: GB     Great Britan

         Sales Rep: HXM    Harry Munvig 333
```

Refactoring Demo 2.1

- Implement RESTful annotations



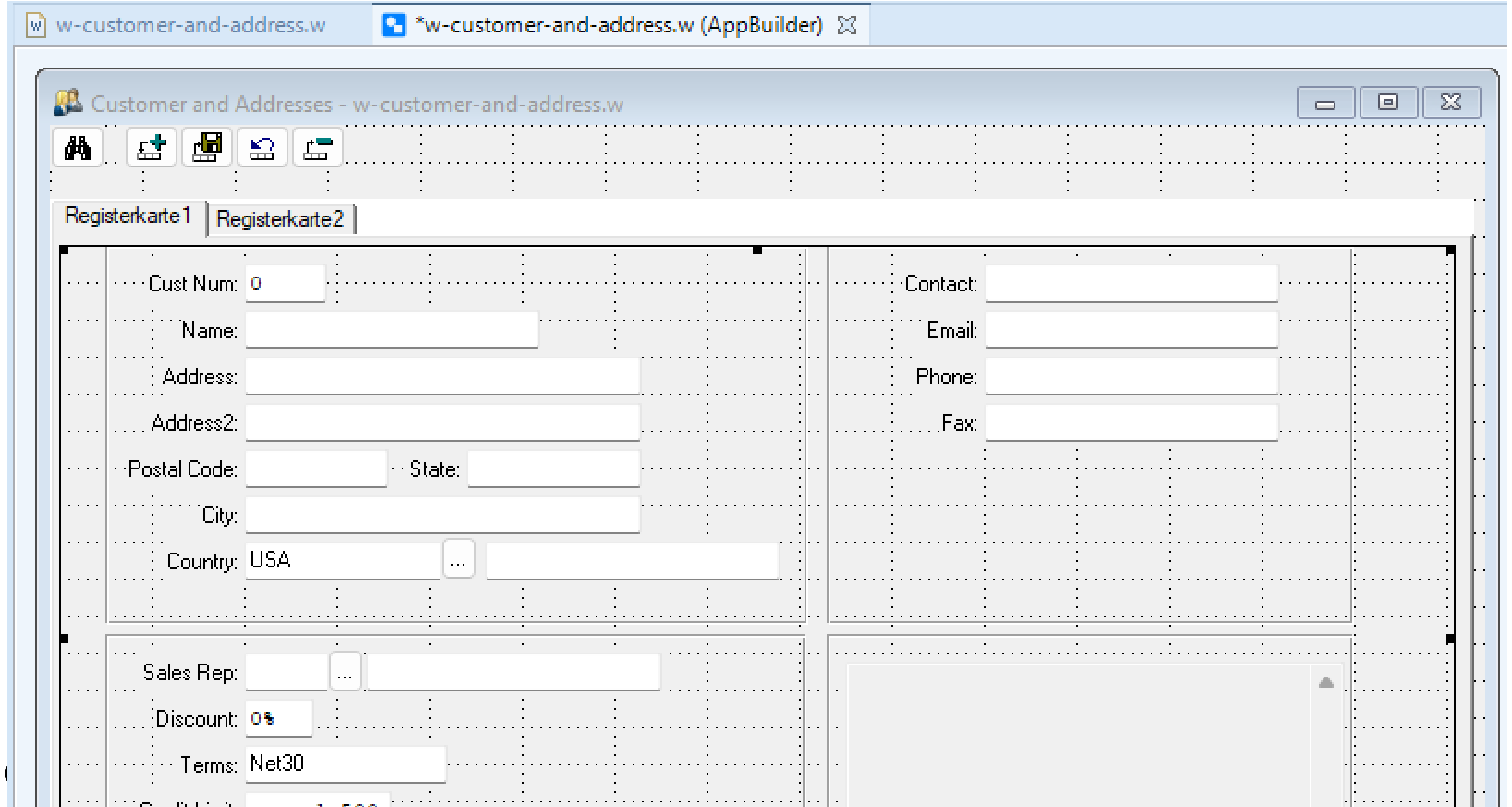
```
-------------------------

Add method HelloAiAudience to the logic object of the CustomerViewer in the repository.

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

Using the SmartframeworkMCP server, create a Business Entity called CustomerDemoBusinessEntity in the Demo.AiRefactored.Customer package for the Custoimer table with additional fields for Salesrep.RepName and Country.CountryName

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

Explain the validation rules found in the update editing block in the updateCustomer procedure in Demo/update-editing-sample.w

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

Refactor the validation logic found in the update editing block in the updateCustomer procedure in Demo/update-editing-sample.w into the ValidateData method of the Demo.AiRefactored.Customer.CustomerDemoBusinessEntity following the Business Entity Validation rules.

Refactor database access to ORM wrapper

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

/add-restaddress-annotation-to-businessentity  /CustomerAiDemo
```

Refactoring Demo 2.2

- Implement RESTful annotations

Refactoring Demo 3

- Migration of GUI application (AppBuilder, no ADM)


w-customer-and-address.w
*w-customer-and-address.w (AppBuilder)
Customer and Addresses - w-customer-and-address.w
Registerkarte1
Registerkarte2
...
...

Cust Num: 0
Name:
Address:
Address2:
Postal Code: State:
City:
Country: USA
Contact:
Email:
Phone:
Fax:
Sales Rep:
Discount: 0%
Terms: Net30

Refactoring Demo 3

- Migration of UI components from existing ABL GUI
- Migration of Lookups based on multiple ABL widgets and parsing of trigger code
- Review in Repository Designer

Consultingwerk
software architecture and development
SmartUpdate
Stay Tuned for the Latest News,
Product Updates, and Event Highlights!

Subscribe Now to Consultingwerk’s Official Newsletter!

www.consultingwerk.com/newsletter

or marketing@consultingwerk.com

Join Our Next Webinars

- October 9th - Implementing MCP Servers With OpenEdge
- Integrate Chatbots Into OpenEdge Applications
- Using Windsurf IDE As An AI Coding Assistant For OpenEdge

UPCOMING
WEBINARS

Questions

Consultingwerk

software architecture and development