



# ABAPConf 2024 South Africa

## **Designing Testable ABAP Classes and Packages**

Winfried Schwarzmann, SAP SE | November 13, 2024

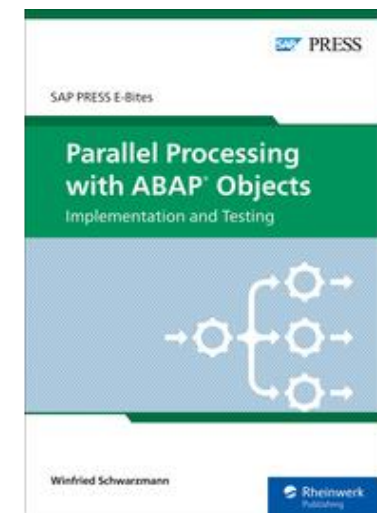
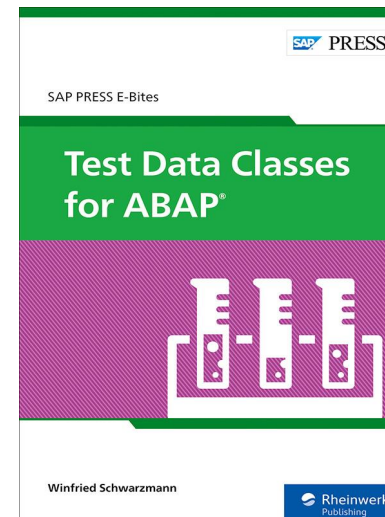
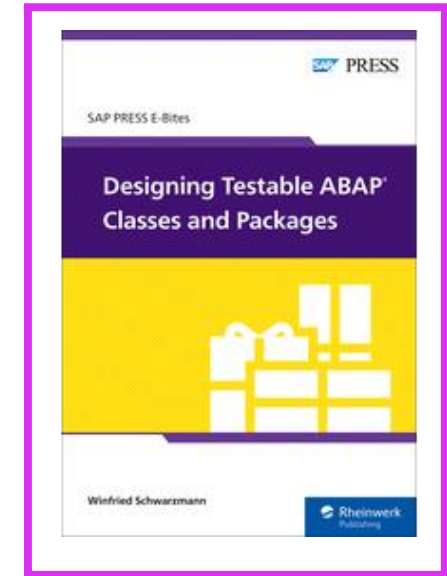
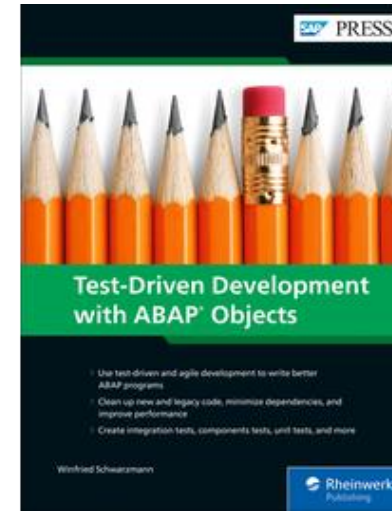
PUBLIC

# Introducing Winfried Schwarzmann

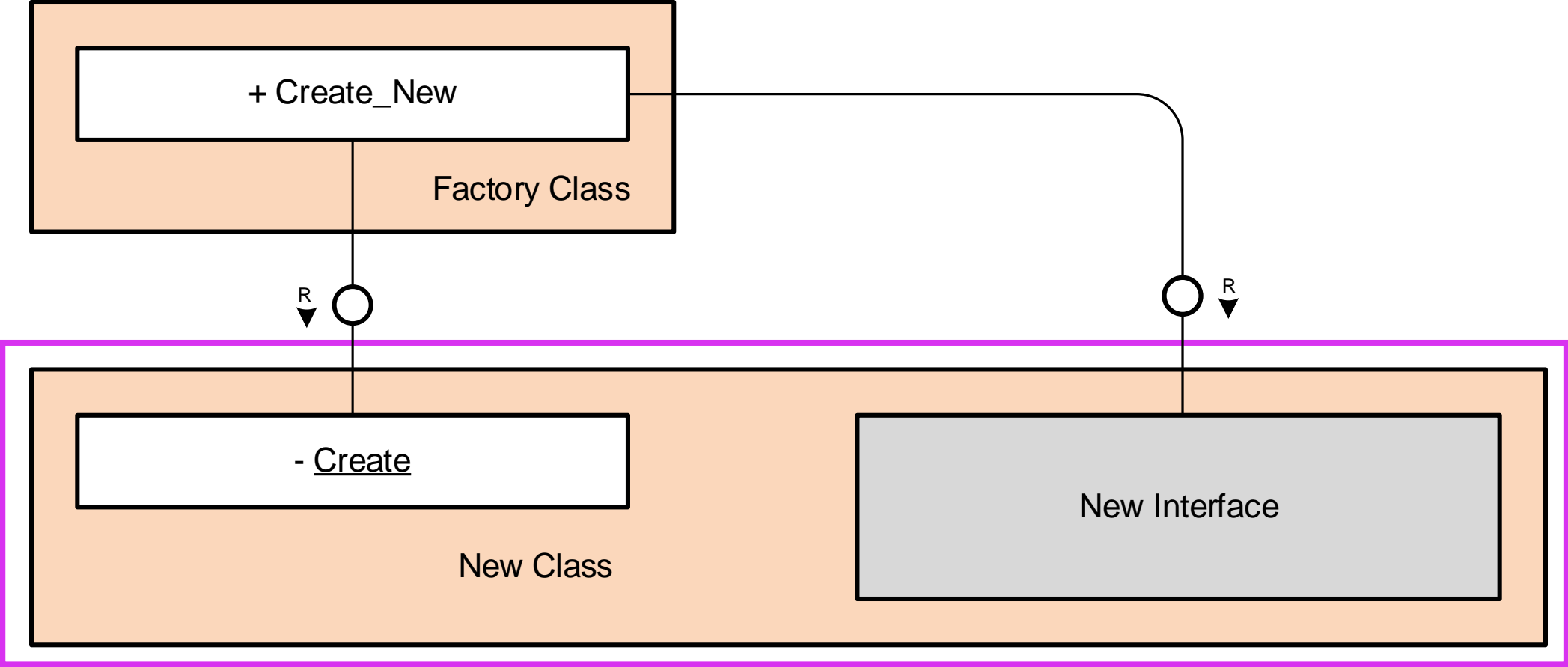


Working for SAP for more than 25 years as:

- Developer and Architect
- Agile Software Engineering Coach



# Designing a New Class



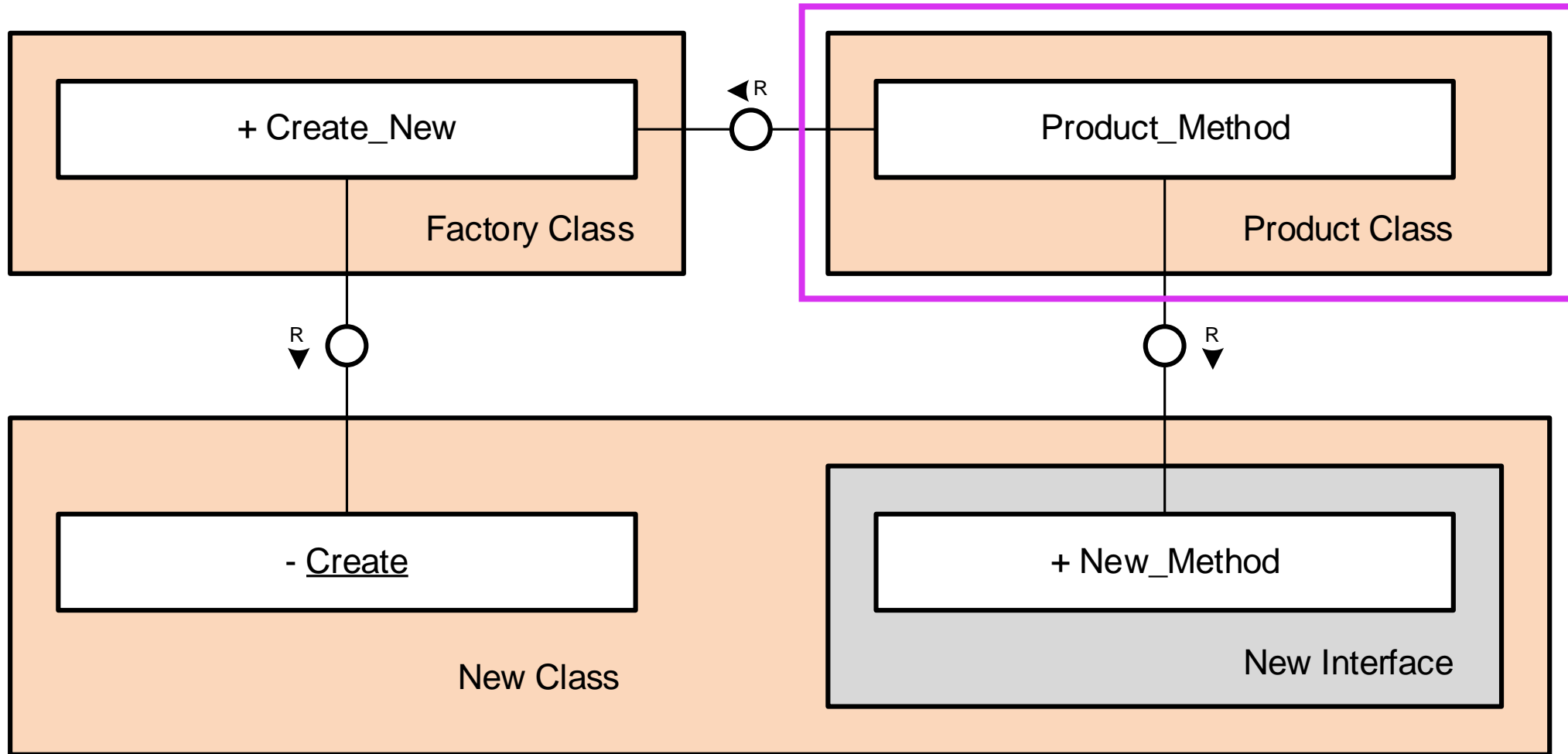
# Implementing the Skeleton of a New Class

```
CLASS cl_new DEFINITION PUBLIC FINAL CREATE PRIVATE  
GLOBAL FRIENDS cl_factory.  
    PUBLIC SECTION.  
        INTERFACES if_new.  
  
    PRIVATE SECTION.  
        CLASS-METHODS create  
            RETURNING VALUE(ro_object) TYPE REF TO cl_new.  
ENDCLASS.
```

```
CLASS cl_new IMPLEMENTATION.  
    METHOD create.  
        ro_object = NEW cl_new( ).  
    ENDMETHOD.  
ENDCLASS.
```



# Decoupling from New Class

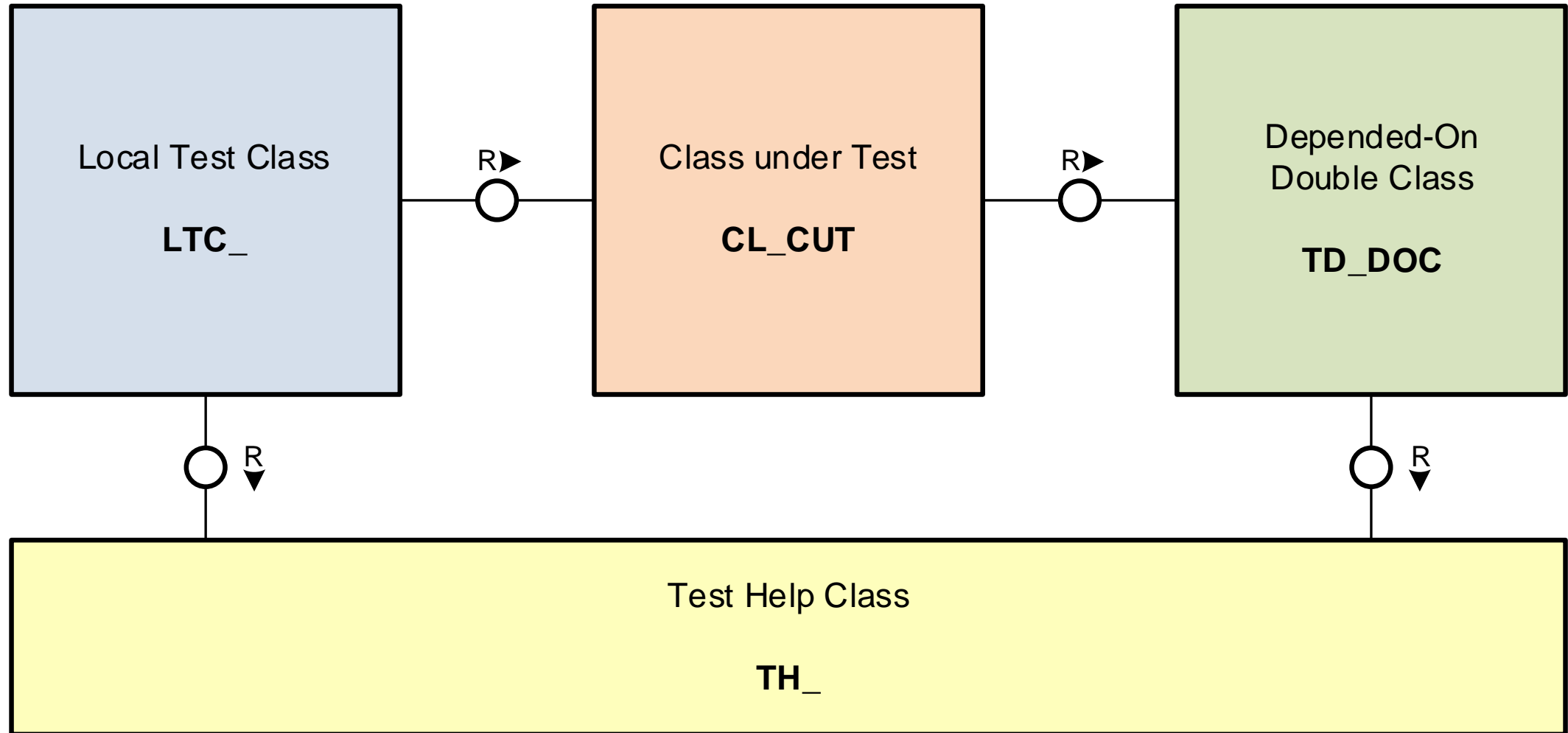


# Using the New Class

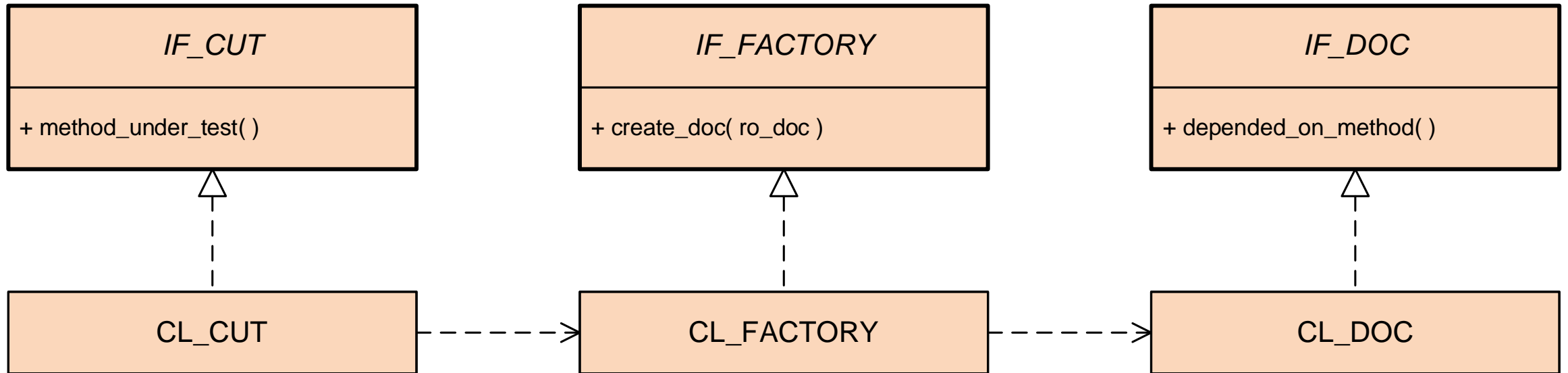
```
CLASS cl_product DEFINITION PUBLIC CREATE PUBLIC.  
    PUBLIC SECTION.  
        METHODS product_method.  
ENDCLASS.
```

```
CLASS cl_product IMPLEMENTATION.  
    METHOD product_method.  
        ...  
        DATA(lo_new_object) = cl_factory=>get( )->create_new( ).  
        lo_new_object->new_method( ).  
        ...  
    ENDMETHOD.  
ENDCLASS.
```

# Test Abbreviations

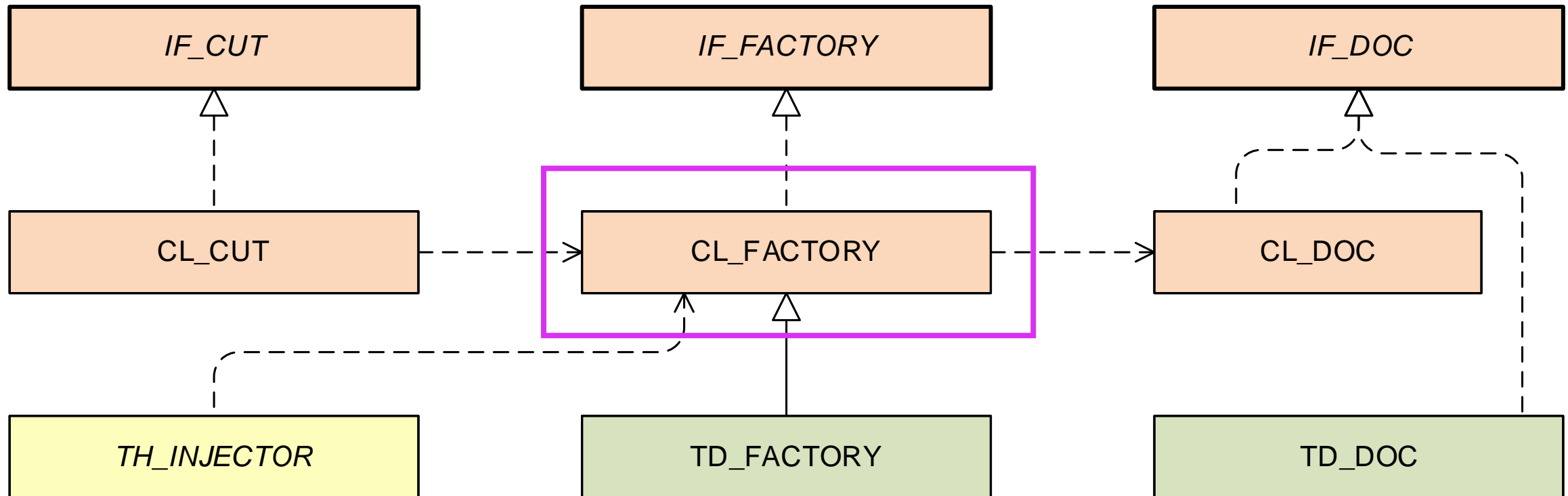


# Clean Design: Product Classes





# Clean Design: Double Classes



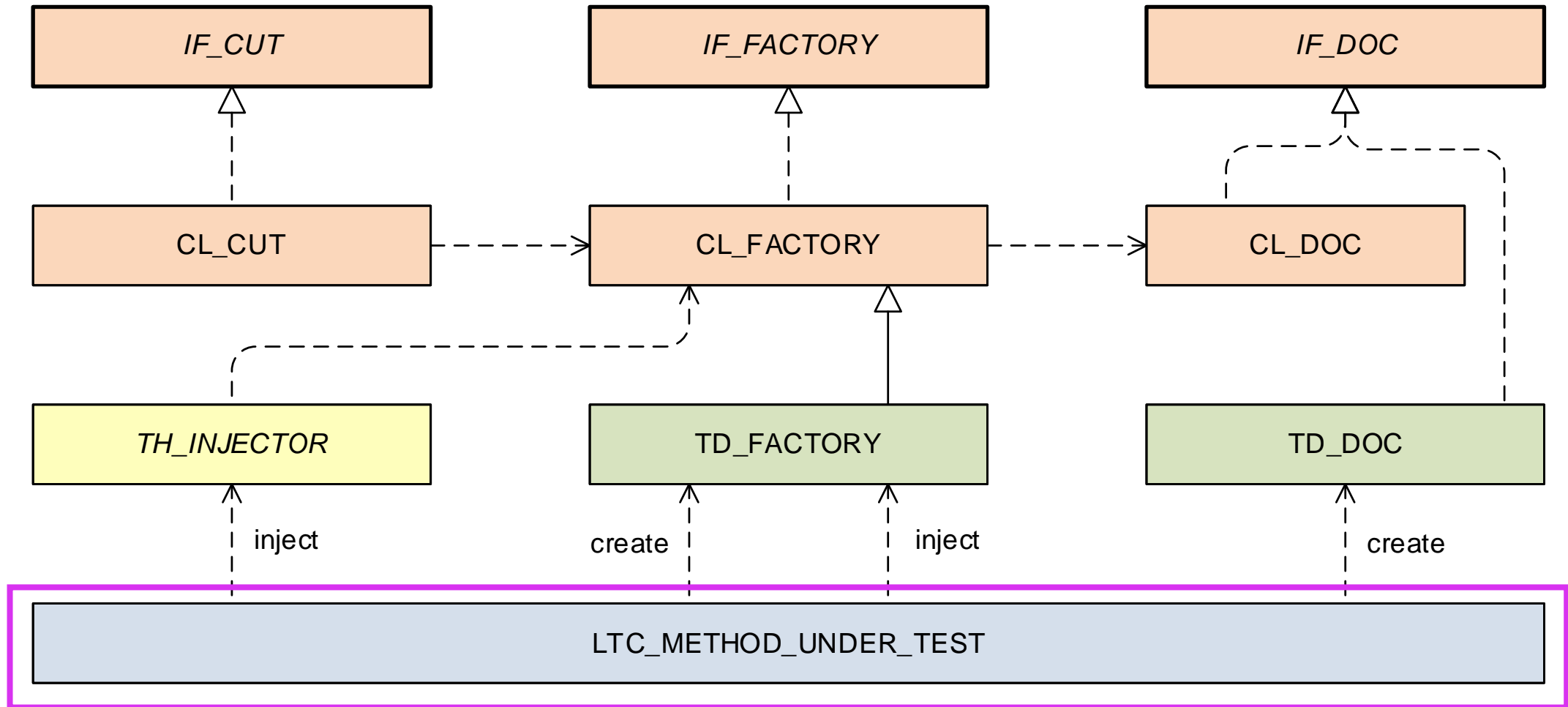
# Singleton Factory

```
CLASS cl_factory DEFINITION PUBLIC CREATE PROTECTED
GLOBAL FRIENDS th_injector.
    PUBLIC SECTION.
        CLASS-METHODS get
            RETURNING VALUE(ro_factory) TYPE REF TO if_factory.

    INTERFACES if_factory.

    PRIVATE SECTION.
        CLASS-DATA so_factory TYPE REF TO if_factory.
ENDCLASS.
```

# Clean Design: Local Test Class



# Local Test Class: Injecting Singleton Double (1)

```
CLASS ltc method under test IMPLEMENTATION.  
  METHOD setup.  
    DATA(lo_factory_double) = td_factory=>create( ).  
    th_factory_injector=>inject_factory( lo_factory_double ).  
  
    mo_doc_double = td_doc=>create( ).  
    lo_factory_double->inject_doc( mo_doc_double ).  
  ENDMETHOD.  
  
  METHOD test_method.  
    mo_doc_double->configure( ... ). "late configuration  
    ...  
  ENDMETHOD.  
ENDCLASS.
```

## Local Test Class: Injecting Singleton Double (2)

```
CLASS ltc_method_under_test DEFINITION FINAL  
FOR TESTING DURATION SHORT RISK LEVEL HARMLESS.  
    PRIVATE SECTION.  
        METHODS setup.  
        METHODS test_method FOR TESTING.  
  
        DATA mo_doc_double TYPE REF TO td_doc.  
ENDCLASS.
```

# Local Test Class: Injecting Non-Singleton Doubles (1)

```
CLASS ltc method under test IMPLEMENTATION.
```

```
  METHOD setup.
```

```
    mo_factory_double = td_factory=>create( ).
```

```
    th_factory_injector=>inject_factory( mo_factory_double ).
```

```
  ENDMETHOD.
```

```
  METHOD test_method.
```

```
    DATA(lo_doc_double_1) = td_doc=>create(...). "configuration
```

```
    DATA(lo_doc_double_2) = td_doc=>create(...). "configuration
```

```
    mo_factory_double->inject_doc( lo_doc_double_1 ).
```

```
    mo_factory_double->inject_doc( lo_doc_double_2 ).
```

```
    . . .
```

```
  ENDMETHOD.
```

```
ENDCLASS.
```

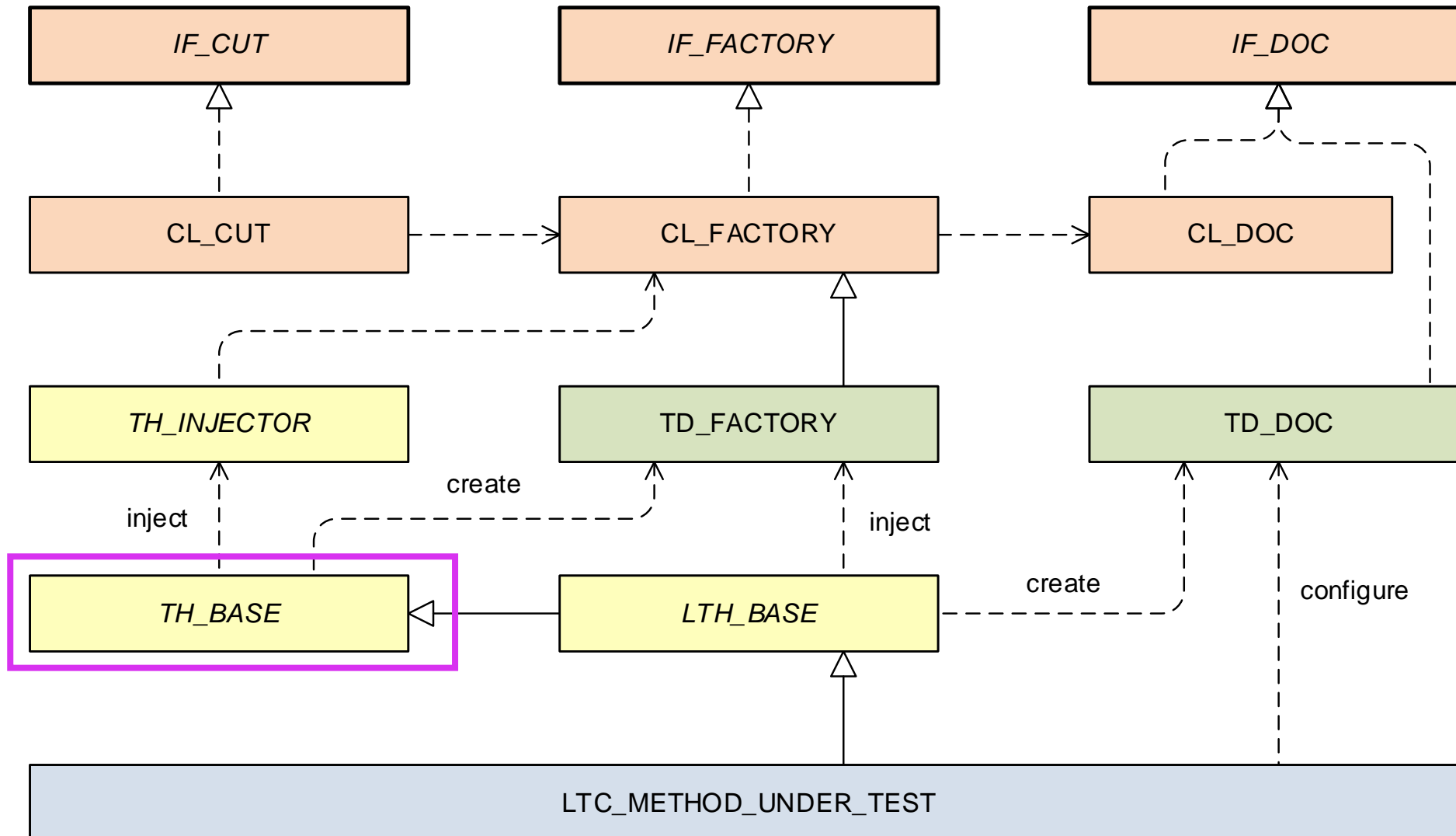
## Local Test Class: Injecting Non-Singleton Doubles (2)

```
CLASS ltc_method_under_test DEFINITION FINAL
FOR TESTING DURATION SHORT RISK LEVEL HARMLESS.
    PRIVATE SECTION.
        METHODS setup.
        METHODS test_method FOR TESTING.

        DATA mo_factory_double TYPE REF TO td_factory.
ENDCLASS.
```



# Clean Design: Test Base Classes



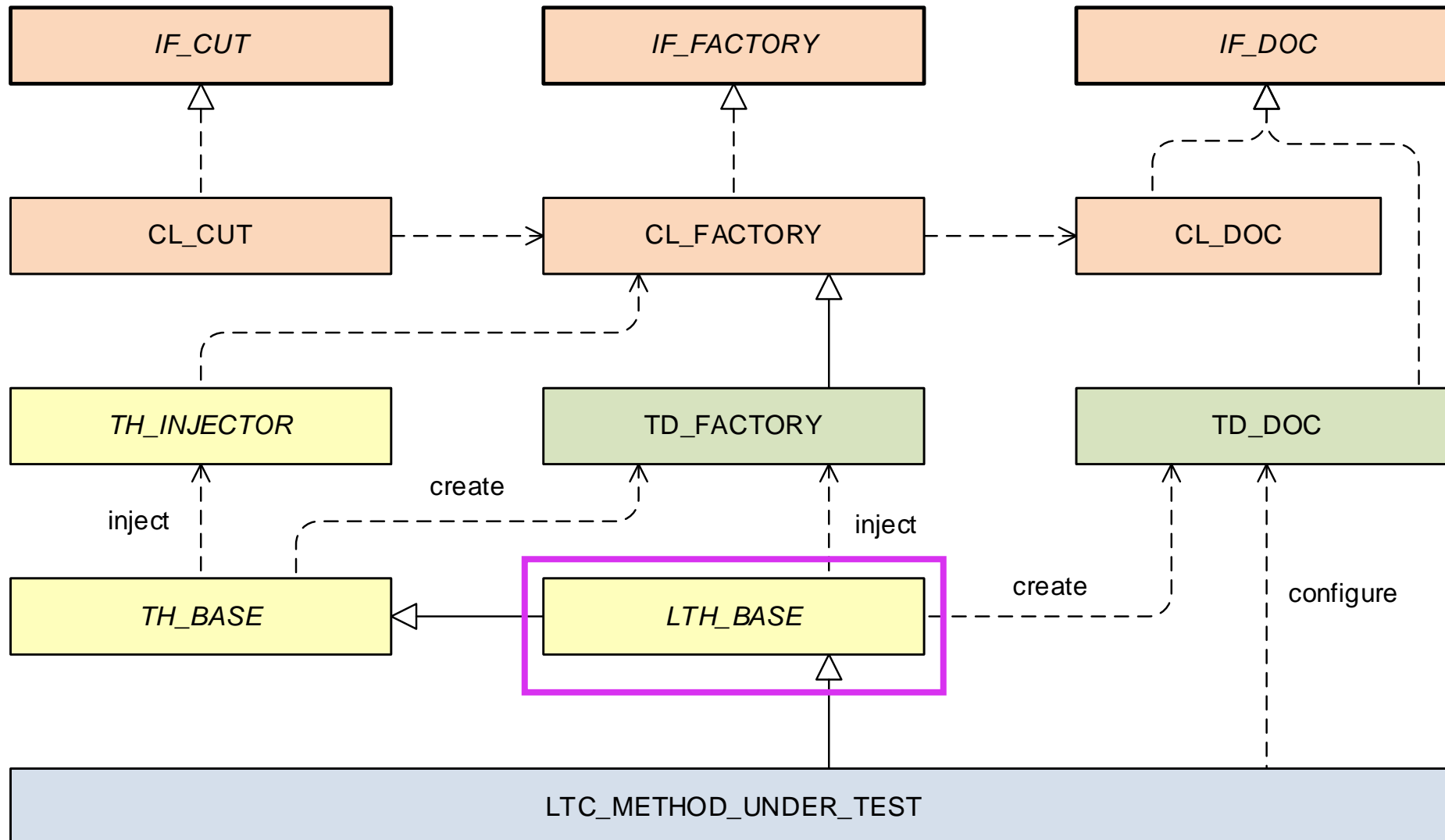
# Global Test Base Class: Injecting Factory Double

```
CLASS th_base DEFINITION PUBLIC ABSTRACT
FOR TESTING DURATION SHORT RISK LEVEL HARMLESS.
    PROTECTED SECTION.
        DATA mo_factory_double TYPE REF TO td_factory.

    PRIVATE SECTION.
        METHODS setup.
ENDCLASS.

CLASS th_base IMPLEMENTATION.
    METHOD setup.
        mo_factory_double = td_factory=>create( ).
        th_factory_injector=>inject_factory( mo_factory_double ).
    ENDMETHOD.
ENDCLASS.
```

## Clean Design: Test Base Classes



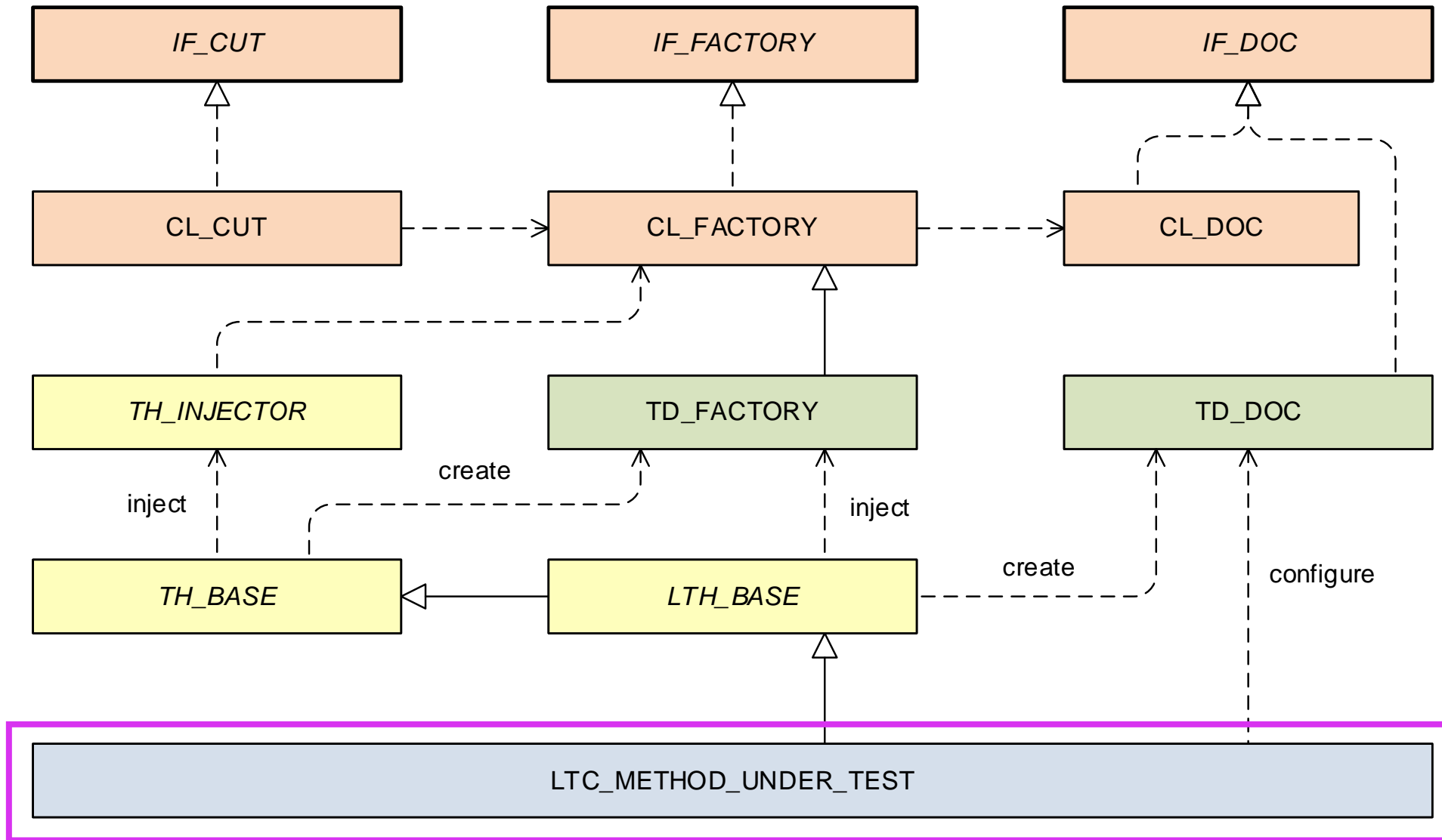
# Local Test Base Class: Injecting Singleton Double

```
CLASS lth_base DEFINITION ABSTRACT
INHERITING FROM th_base
FOR TESTING DURATION SHORT RISK LEVEL HARMLESS.
    PROTECTED SECTION.
        DATA mo_doc_double TYPE REF TO td_doc.

    PRIVATE SECTION.
        METHODS setup.
ENDCLASS.

CLASS lth_base IMPLEMENTATION.
    METHOD setup.
        mo_doc_double = td_doc=>create( ).
        mo_factory_double->inject_doc( mo_doc_double ).
    ENDMETHOD.
ENDCLASS.
```

# Clean Design: Test Base Classes

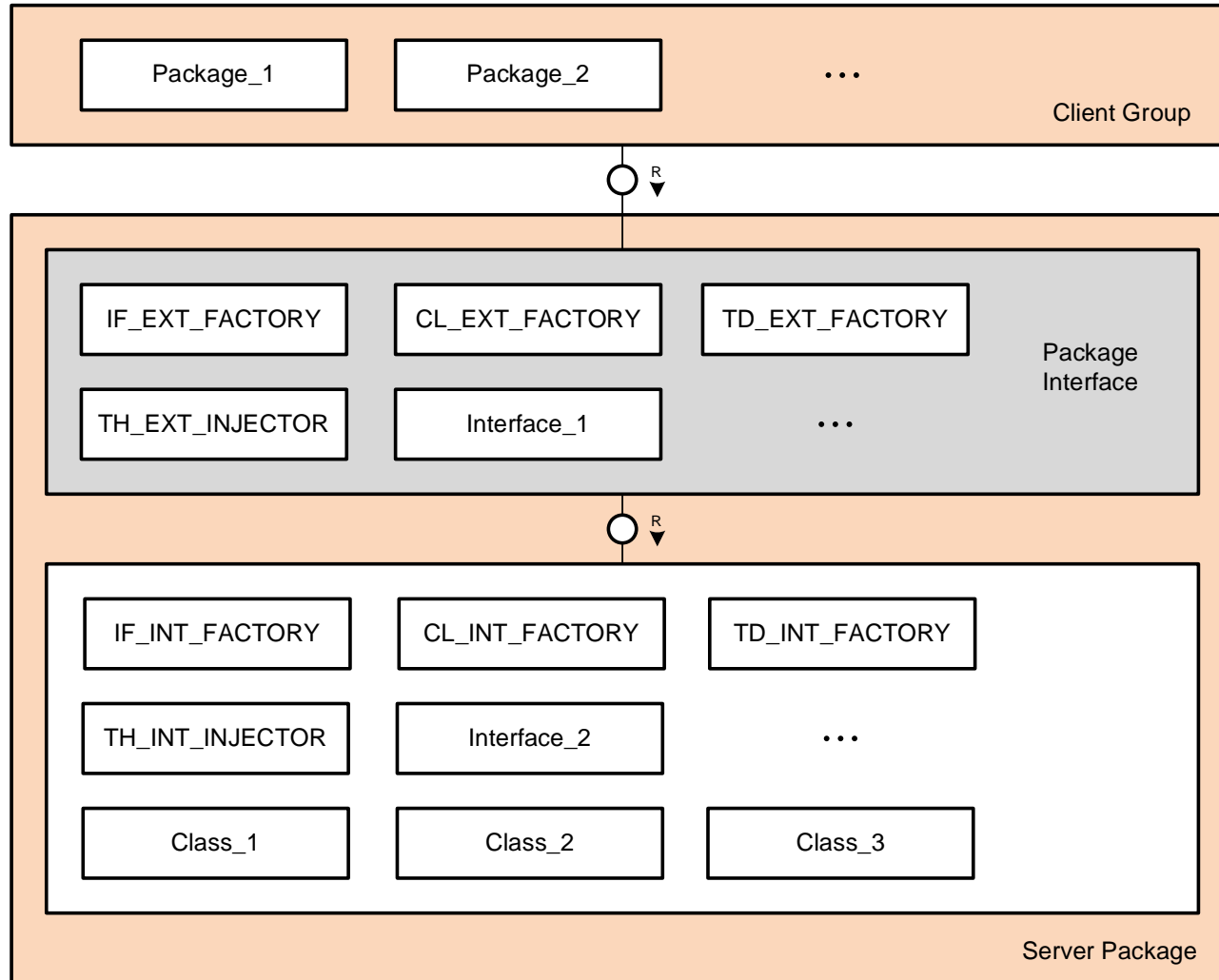


# Local Test Class: Configuring Singleton Double

```
CLASS ltc method under test DEFINITION  
INHERITING FROM lth_base FINAL  
FOR TESTING DURATION SHORT RISK LEVEL HARMLESS.  
PRIVATE SECTION.  
    METHODS test_method FOR TESTING.  
ENDCLASS.
```

```
CLASS ltc_method_under_test IMPLEMENTATION.  
    METHOD test_method.  
        mo_doc_double->configure( ... ). "late configuration  
        ...  
    ENDMETHOD.  
ENDCLASS.
```

# Clean Package

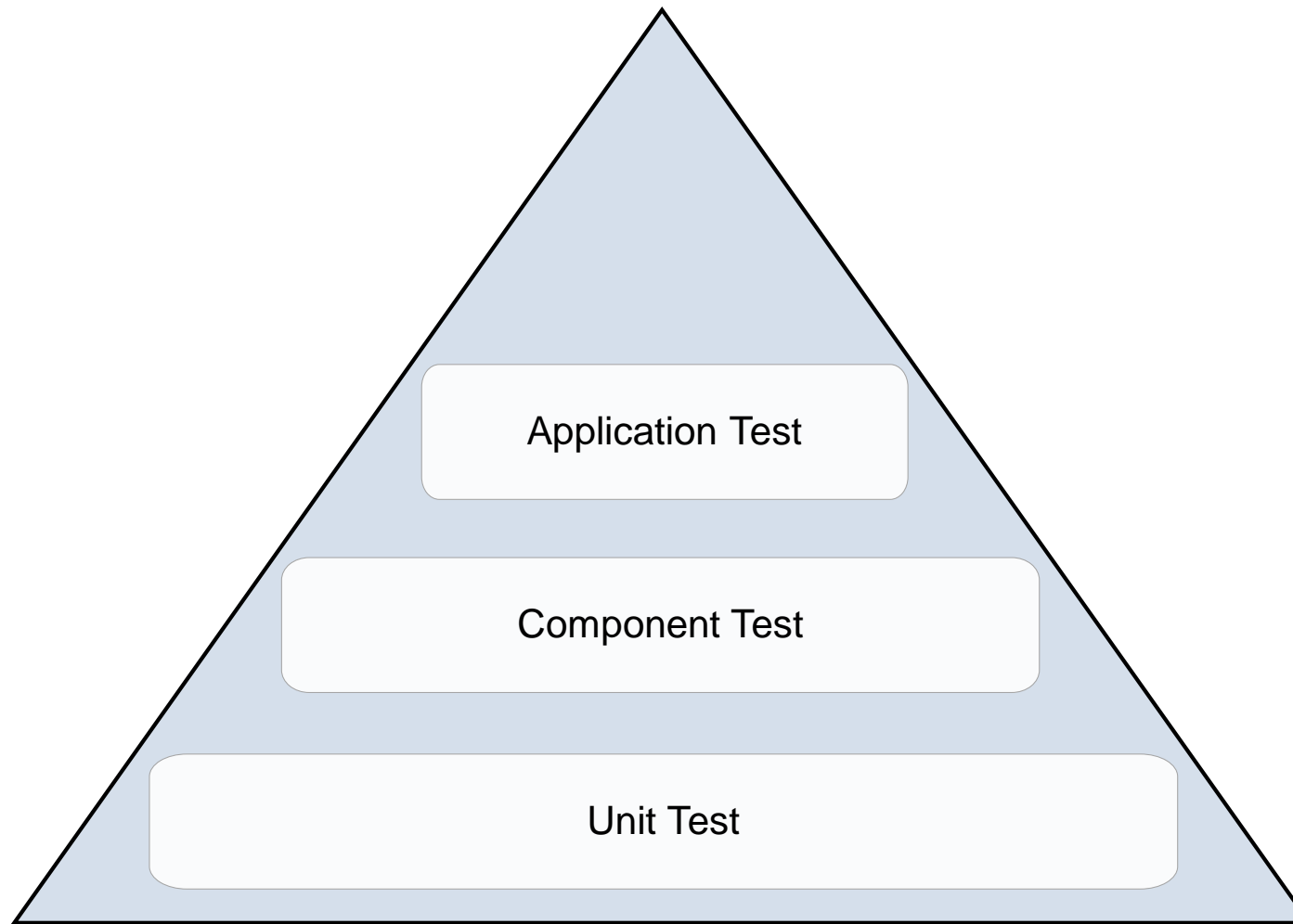


External factory provides access to the interfaces of the API Units.

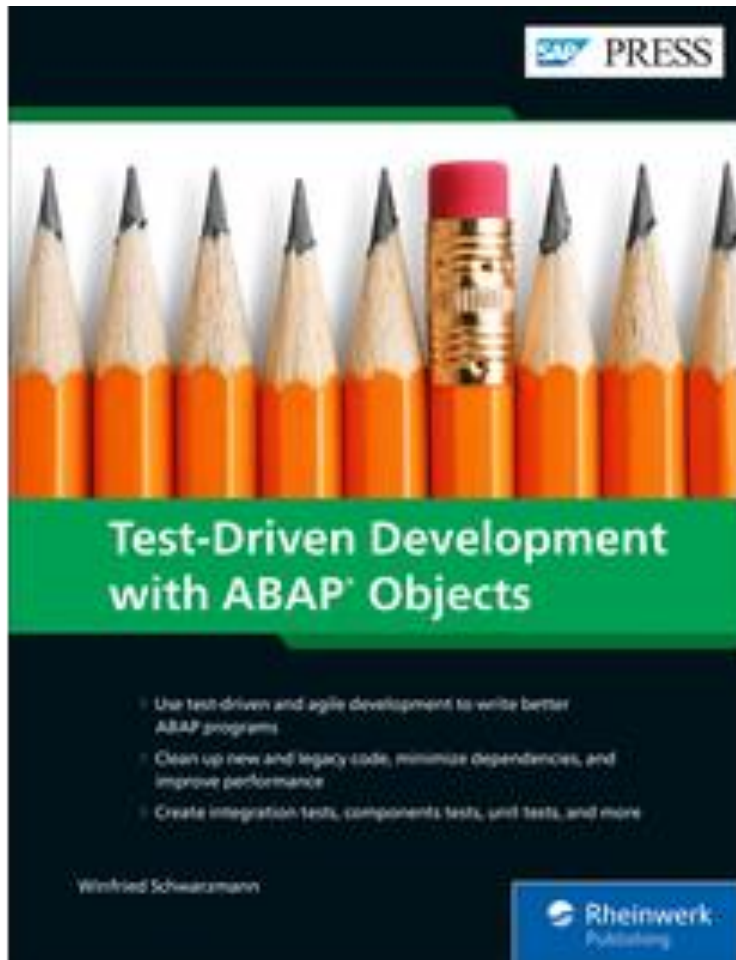
Internal factory decouples encapsulated units.



# Test Pyramid



# Test-Oriented Improvement Process

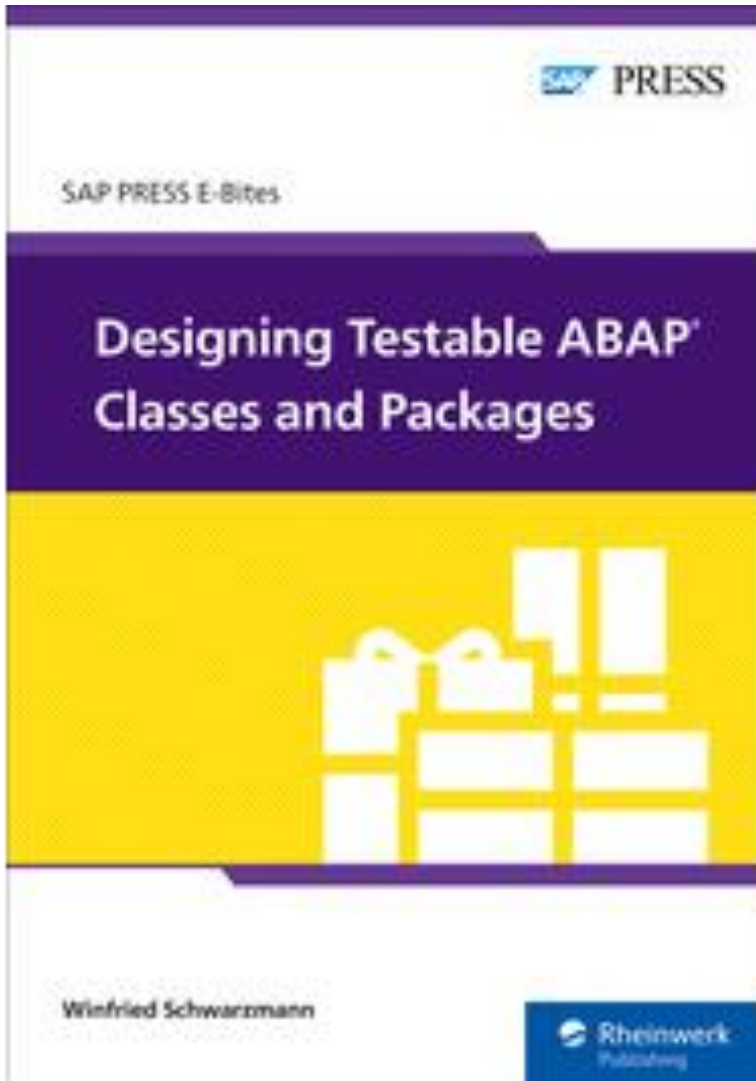


Book & E-Book  
SAP Press  
2019

## Content:

- Part I:  
Modernization of legacy code
- Part II:  
Test infrastructure
- Part III:  
Test-driven development for new code
- Part IV:  
Agile software engineering
- Part V:  
Development & test tools

# Clean Design



E-Book  
SAP Press  
2022

## Content:

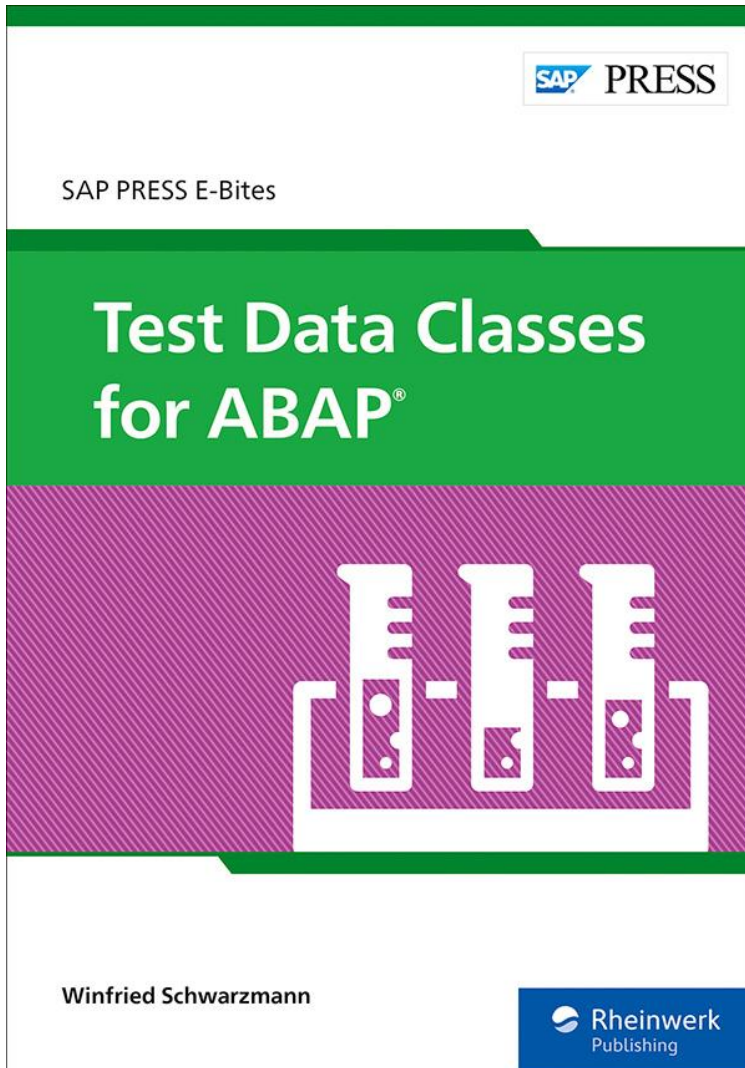
### Part I: Theory

- Classes
- Test Classes
- Packages
- BAdIs

### Part II: Training

Exercises with solutions  
for individuals and teams

# Test Data Classes



E-Book  
SAP Press  
2021

## Content:

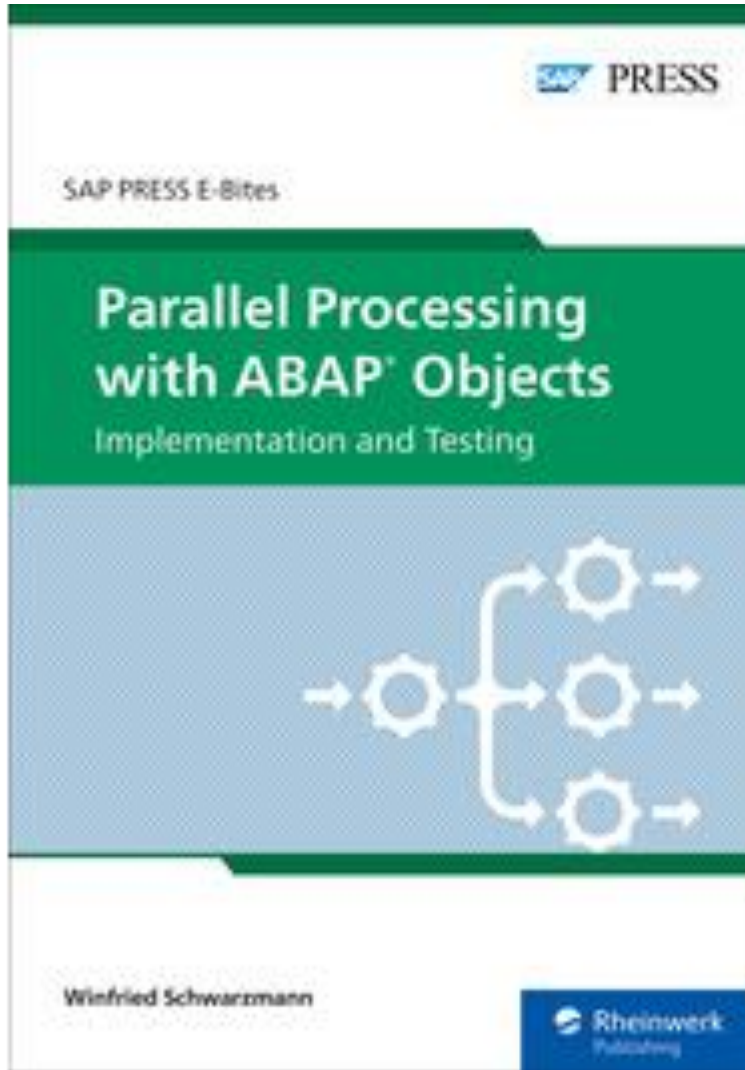
### Part I: Theory

- Designing and implementing test data classes
- Using test data classes for the entire test pyramid
- Using test message classes for verifying error handling

### Part II: Training

Exercises with solutions  
for individuals and teams

# Parallel Processing



E-Book  
SAP Press  
2022

## Content:

Implementing and testing parallel processes

Inheriting test data classes from productive data classes

Implementing, refactoring, and enhancing package design

Winfried Schwarzmann

SAP SE

Follow us



[www.sap.com/contactsap](https://www.sap.com/contactsap)

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company.

The information contained herein may be changed without prior notice. Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

In particular, SAP SE or its affiliated companies have no obligation to pursue any course of business outlined in this document or any related presentation, or to develop or release any functionality mentioned therein. This document, or any related presentation, and SAP SE's or its affiliated companies' strategy and possible future developments, products, and/or platforms, directions, and functionality are all subject to change and may be changed by SAP SE or its affiliated companies at any time for any reason without notice. The information in this document is not a commitment, promise, or legal obligation to deliver any material, code, or functionality. All forward-looking statements are subject to various risks and uncertainties that could cause actual results to differ materially from expectations. Readers are cautioned not to place undue reliance on these forward-looking statements, and they should not be relied upon in making purchasing decisions.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

See [www.sap.com/copyright](https://www.sap.com/copyright) for additional trademark information and notices.