# SAP TUT and FUT Test Plan

**Technical and functional unit testing for an S/4HANA program**

Unit-level testing in an SAP program splits in two: technical unit testing of custom objects, and functional unit testing of configured process steps. Both are executed before anything is integrated, and the quality of both determines how much pain SIT delivers.

> ***How to use this template.*** *Everything in italics is guidance and should be deleted once you have replaced it. Square brackets mark a value to fill in. These four documents are designed to be used together: the strategy sets the rules, and the three plans execute them at each level. Where your program uses different level names, keep your names and map them in section 1 rather than renaming everything.*

## 1. Scope

| | |
|---|---|
| Wave / release covered | |
| Modules in scope | |
| Number of WRICEF objects in scope | |
| FUT owner | |
| TUT owner | |

## 2. Technical Unit Test — custom objects

> *One row per WRICEF object. Evidence is the deliverable: a screenshot or log attached to the object, not a tick in a spreadsheet.*

| Object ID | Type (W/R/I/C/E/F) | Description | Developer | Test data used | Result | Evidence attached |
|---|---|---|---|---|---|---|
| [RICEF-104] | [I] | [inbound IDoc — order confirmation] | [ ] | [ ] | [pass/fail] | [y/n] |
| | | | | | | |

Interfaces and conversions deserve extra attention, because they are the objects most likely to pass in isolation and fail in SIT:

- **Interfaces** — test the error path, not just the happy path. What happens to a malformed message, a duplicate, a message for a customer that does not exist?
- **Conversions** — test with production-shaped volumes and production-shaped mess, not with ten tidy records.
- **Forms and output** — check the rendered output, not just that the job ran.
- **Enhancements** — confirm the standard path still works with the enhancement inactive as well as active.

## 3. Functional Unit Test — configured steps

> *One row per process step. These scripts are the raw material for SIT and, later, for your automated regression suite — so write them as though somebody else will execute them, because somebody else will.*

| Step ID | Process | Transaction / Fiori app | Pre-conditions | Expected result | Result |
|---|---|---|---|---|---|
| [FUT-O2C-010] | [O2C-01] | [VA01 / F1234] | [customer and material exist] | [order created, status = open] | [ ] |
| | | | | | |

Write the expected result as a **specific, checkable value**. "Order is created" passes when the order is created with the wrong pricing; "order created with net value 1,250.00 EUR and status Open" does not.

## 4. Access route classification

Studio drives a **real browser**. That covers **SAP Fiori** and **SAP GUI for HTML (Web GUI)**, plus the surrounding web applications in scope — Ariba, SuccessFactors, Concur, supplier and customer portals, and your own custom front ends. It does **not** drive SAP GUI for Windows, which is a desktop client. Classify each in-scope transaction by its access route before you plan automation, because that one column decides what is and is not a candidate.

| Process step | Access route<br>(Fiori / Web GUI / SAP GUI for Windows / other) | In scope for browser automation? |
|---|---|---|
| [FUT-O2C-010] | [Fiori F1234] | [yes] |
| | | |

## 5. Authorizations

> *Roles are usually tested last and cause defects first. Include them here rather than discovering them in UAT.*

| Role | Process steps it must allow | Steps it must **not** allow | Tested? |
|---|---|---|---|
| [Z_SD_CLERK] | [create order, display customer] | [release credit block] | [ ] |
| | | | |

> The negative column is the valuable one, and the one almost always left empty.

## 6. Defects and exit

| Exit criterion | Met? |
|---|---|
| Every in-scope WRICEF object has a TUT result and attached evidence | [ ] |
| Every in-scope process step has a FUT result | [ ] |
| No open severity 1 or 2 defects at unit level | [ ] |
| Access routes classified for every in-scope step | [ ] |
| FUT scripts reviewed for reuse in SIT and regression | [ ] |

**With Functionize Studio:** a reviewed FUT script is already most of what an automated test needs, because the description of intent and the expected value are the same in both. Tagging candidates during FUT rather than after SIT is the cheapest moment to do it.