SAP S/4HANA Test Strategy

A template for an ECC to S/4HANA transformation

This is the program-level test strategy for a move from SAP ECC to S/4HANA. It defines the test levels, who owns each one, how business processes are traced to tests, and which of those tests are worth automating — so that the regression effort does not grow linearly with every wave.

> ***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. Program context

Program name	
Source system and release (e.g. ECC 6.0 EhP8)	
Target (e.g. S/4HANA Cloud Private Edition via RISE with SAP)	
Approach (greenfield / brownfield / selective data transition)	
Systems integrator	
Go-live approach (big bang / phased by region or entity)	
Program test lead	
Business test lead	

> *Note the approach honestly. A brownfield conversion carries a much heavier regression obligation than a greenfield build, because the business expects everything that worked yesterday to still work.*

2. Test levels

> *Map your program's names onto these if they differ. The point is that each level proves something the level below cannot.*

Level	What it proves	Owner	Environment	Typically
TUT — Technical Unit Test	A custom object works in isolation — the WRICEF items: workflows, reports, interfaces, conversions, enhancements, forms	[SI development]	[DEV]	Developer-executed, evidence attached to the object
FUT — Functional Unit Test	A configured process step behaves per the design — one transaction or Fiori app at a time	[SI functional]	[DEV / QA]	Functional consultant, script per step
SIT — System Integration Test	An end-to-end business process works across modules, interfaces and surrounding systems	[joint SI and business]	[QA]	Cycles — usually 2 to 4
UAT — User Acceptance Test	The business agrees the process is right and they can do their job	[business]	[QA / pre-prod]	Business users, scenario-based
Regression	What already worked still works after each transport, wave or support pack	[ ]	[QA]	The largest repeated effort — and the automation target

> Regression is deliberately listed as a level rather than a phase. In a phased program it runs continuously from the first wave to well past the last, and treating it as an event at the end is how programs discover they have no capacity left.

3. Business process inventory

> *This is the spine of the whole test effort. Build it once, from your fit-to-standard workshop output and your process hierarchy, and reuse it at every level.*

L2 / L3 process	Process ID	Module(s)	Criticality (H/M/L)	Volume	Owner
[Order to Cash — standard sales order]	[O2C-01]	[SD]	H	[high]	[ ]
[Procure to Pay — three-way match]	[P2P-01]	[MM/FI]	H		
[Record to Report — period close]	[R2R-01]	[FI/CO]	H		

Criticality here should be a **business** judgment, not a technical one. The question is what it costs if the process silently stops working in production, not how complex it was to configure.

4. Tracing processes to requirements and tests

Process ID	Requirement / design doc	WRICEF objects	TUT	FUT	SIT scenario	UAT scenario
[O2C-01]	[FD-012]	[RICEF-104]	[ ]	[ ]	[SIT-07]	[UAT-03]

> *One row per process, not per test. The value is being able to answer "if this process broke, which tests would tell us"*

> *without a workshop.*

5. Access route — the column that decides automation

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 (Fiori / Web GUI / SAP GUI for Windows / other)	In scope for browser automation?
[Create sales order]	[Fiori app F1234]	[yes]
[Post goods issue]	[Web GUI — VL02N]	[yes]
[Legacy batch config]	[SAP GUI for Windows]	[no — stays manual]

> Do this classification early, in the Explore phase. Discovering in cycle 2 of SIT that half your scope is desktop-only is an expensive surprise.

6. Automation candidates

> *Score the processes from section 3 that survived section 5. Everything else stays manual, and that is a legitimate answer.*

Process ID	Access route OK?	Run frequency	Deterministic?	Master data controlled?	Automate?
[O2C-01]	[yes]	[every regression cycle]	[yes]	[yes]	[yes]

Prioritize in this order, which is not the order most programs default to:

1. **High-volume regression** — the same processes re-executed every cycle, every wave. This is where the manual effort actually is.
2. **Critical business processes** — order to cash, procure to pay, period close.
3. **Processes touched by every transport** — pricing, output, authorizations.
4. **Cross-system journeys** — where the seam is owned by nobody.

> **With Functionize Studio:** a test is described in plain language rather than scripted against screen elements, so a functional consultant or a business user can create and read one. In a program where test authorship is concentrated in a small number of automation specialists, that is usually the constraint that lifts.

7. Test data and master data

> *In an S/4HANA program this is the most common cause of failed test cycles — more common than defects in the software.*

How master data is loaded into the test environment

How often it is refreshed, and who decides

Number range strategy in test

Org structure used for testing (company code, plant, sales org)

Who owns test data for each process area

> Automated tests should create the data they need, or use values you control. A test that depends on a record somebody else might change is a scheduled false alarm, and in a shared SAP test client that is the default situation unless you design against it.

8. Environments and cutover

Environment	Purpose	Refreshed from	Frequency	Owner
[DEV]	TUT, FUT	[ ]	[ ]	[ ]
[QA]	SIT, regression	[ ]	[ ]	[ ]
[Pre-prod]	UAT, cutover rehearsal	[ ]	[ ]	[ ]

Number of cutover rehearsals planned

Regression suite executed at each rehearsal?

Hypercare period and who tests during it

9. Entry and exit criteria per level

Level	Entry	Exit
TUT	[object built, unit test script written]	[all objects executed, evidence attached]
FUT	[configuration complete, TUT passed for related objects]	[ ]
SIT	[FUT passed, interfaces connected, master data loaded]	[all in-scope processes executed, no open sev 1/2]
UAT	[SIT exit met, regression green, business users allocated]	[business sign-off obtained]

10. Responsibilities: you and your systems integrator

> *Ambiguity here is expensive. Agree it before the first cycle, not during it.*

Activity	SI	Client	Joint
Test strategy and plans	[ ]	[ ]	[ ]
TUT and FUT execution	[X]		
SIT scenario design			[X]
SIT execution			[X]
UAT scenario design and execution		[X]	
Regression suite build	[ ]	[ ]	[ ]
Regression suite ownership after go-live	[ ]	[ ]	[ ]
Test data provisioning	[ ]	[ ]	[ ]
Defect triage			[X]

> The row that matters most is regression suite ownership after go-live. A suite the SI built and nobody on your side can read becomes unmaintainable the day they leave. Plain-language tests are materially easier to hand over than scripted ones — make handover a contractual deliverable either way.

11. Measures

Measure	Baseline	Target
Manual test person-days per regression cycle	[ ]	Falling per wave
Processes covered by automated regression	[0]	[ ]
Defects found in SIT vs UAT	[ ]	Most found in SIT — UAT is not a defect hunt
Defects escaping to production after go-live	[n/a]	[ ]
Time to execute a full regression pass	[ ]	[ ]

12. Sign-off

Program test lead / date

SI test lead / date

Business process owner / date

Next review due