DOCUMENT 07 · INPUT AND OUTPUT CONTRACTS

Input and output contracts

The mandatory structure of every SPAD artefact, readable by people and by machines, so that the "incomplete artefact" validation can be automated and agents can chain phases without losing information

Fernando García Varela Version 0.1 19-09-2026 Under construction

18	1	3	4
contracts	block common to all	verdict values	severity levels

IN THIS DOCUMENT

01 Purpose

02 Common conventions

03 Main cycle contracts

04 Complementary cycle contracts

05 Format of the contexts

06 Automatic validation

07 Related documents

08 Version control

Version 0.x, under review. SPAD is currently at version 0.x. The content is operational and may be used, adapted and redistributed under the project licence (CC BY 4.0 for content and MIT for code), crediting the author and indicating changes. Documents and tools are still being reviewed: feedback and proposals for improvement are welcome.

Legal notice and disclaimer. SPAD is a reference methodology provided "as is" and for information purposes only. It does not constitute legal, regulatory or professional advice, does not guarantee results or compliance with any law or standard and is not a certification. **Each organisation that uses SPAD is solely responsible for validating its results, identifying the regulation that applies to it, verifying that it is current and certifying its own regulatory compliance**, with the appropriate qualified advice. This methodology is a generic, free aid shared with the community so that nobody has to start from scratch; each person or organisation can and should adapt it to its own use. It must not be inferred that its legally sensitive parts have been reviewed by legal counsel: those reviews, for each company or sector, are the ultimate responsibility of the company, consultant or organisation that uses it. Although every effort is made to keep it up to date, some regulation may have changed without being reflected here. To the fullest extent permitted by law, the author accepts no responsibility whatsoever for the effects of its application in any organisation or for its full applicability. The methodology does not grant certification of any kind. The author accepts no liability for any use made of this content or for any decisions taken on the basis of it. The data, figures, companies and cases in the examples are fictitious or illustrative.

01

Purpose

Each SPAD phase produces an artefact with a **contractual structure**: mandatory sections and fields that the instruction requires (document 06) and whose absence invalidates the response (document 03). This document defines that structure in JSON, so that:

- a person can check at a glance that the output is complete;
- an automatic validator can reject an incomplete artefact before anyone sees it;
- an agent that chains phases receives from the previous one exactly what it needs.

The artefact may be presented in Markdown with the same sections; JSON is the canonical form for validation and exchange.

> **Why it matters.** "Incomplete artefact" is the most frequent cause of invalidation and the easiest to automate. An explicit contract turns that check into a mechanical step and leaves to the person the judgement that does require a person: whether the content is correct.

02

Common conventions

2.1 Common block

Every artefact includes these fields:

```
{
  "spad": { "contract_version": "0.1" },
  "topic": "string, snake_case, maximum 20 characters",
  "phase": "PLAN | AUDIT_PLAN | CODE_PRIMER | TEST_STRATEGY | IMPLEMENTATION | TEST_IMPLEMENTATION | A
  "iteration": "integer from 1",
  "date": "ISO 8601",
  "model": {
    "provider": "string",
    "model": "string",
    "model_version": "string",
    "role": "planner | reviewer | builder | fixer | analyst | diagnostic | security_reviewer",
    "instruction": "identifier and version of the instruction used",
    "global_context": "version",
    "project_context": "version"
  },
  "deviations": [
    { "situation": "string", "decision_requested": "string" }
  ]
}
```

`deviations` records the situations that the phase could not resolve and on which a decision is requested; it never contains decisions taken by the AI.

2.2 Scales

Scale	Values
Verdict	`GO` · `GO_WITH_CHANGES` · `NO_GO` (all reviews, including the security review and the behaviour evaluation).
Severity	`CRITICAL` · `HIGH` · `MEDIUM` · `LOW`.
Probability	`HIGH` · `MEDIUM` · `LOW`.
Recommendation priority	`MUST` · `SHOULD` · `NICE_TO_HAVE`.
Quality	`EXCELLENT` · `GOOD` · `ADEQUATE` · `POOR` · `MISSING`.

2.3 Human validation record

It is not part of the AI's artefact: the person writes it in the topic's `VALIDATION.md` (document 02, section 4.2). Its canonical form:

```
{
  "artefact": "SPAD_02_AUDIT_PLAN",
  "person": "identifier",
  "date": "ISO 8601",
  "outcome": "VALID | INVALID | EXCEPTION_APPROVED",
  "invalidation_cause": "PHASE_VIOLATION | INCOMPLETE_ARTEFACT | OFF_PLAN | NO_PRIOR_REVIEW | SELF_API
  "ai_verdict": "GO | GO_WITH_CHANGES | NO_GO | null",
  "verdict_accepted": "true | false | null",
  "observations": "string"
}
```

03

Main cycle contracts

3.1 PLAN

```
{
  "architecture": { "overview": "string", "patterns": ["string"] },
  "components": [
```

```
    { "name": "string", "responsibility": "string", "dependencies": ["string"], "interfaces": ["string
  ],
  "flows": [
    { "name": "string", "trigger": "string", "steps": ["string"], "data": "string" }
  ],
  "decisions": [
    { "decision": "string", "justification": "string", "alternatives": ["string"], "trade_offs": "str
  ],
  "risks": [
    { "risk": "string", "severity": "CRITICAL|HIGH|MEDIUM|LOW", "probability": "HIGH|MEDIUM|LOW", "mi
  ],
  "ai_system": {
    "applies": "true | false",
    "autonomy_level": "A0 | A1 | A2 | A3",
    "ai_components": ["model, instructions, retrieval, tools, memory"],
    "action_limits": ["string"],
    "data_to_model": { "provider": "string", "information_classes": ["string"] },
    "failure_modes": [ { "failure": "string", "safe_behaviour": "string" } ],
    "model_change": "strategy for provider updates"
  }
}
```

3.2 AUDIT_PLAN

```
{
  "issues": [
    { "id": "string", "category": "COUPLING|COHESION|SCALABILITY|CONCURRENCY|SECURITY|OBSERVABILITY|O
      "severity": "CRITICAL|HIGH|MEDIUM|LOW", "description": "string", "location": "section of the pl
  ],
  "recommendations": [
    { "issue": "id", "recommendation": "string", "priority": "MUST|SHOULD|NICE_TO_HAVE" }
  ],
  "exceptions_evaluated": [
    { "global_rule": "string", "exception": "string", "evaluation": "string" }
  ],
  "verdict": "GO|GO_WITH_CHANGES|NO_GO",
  "justification": "string",
  "next_steps": ["string"]
}
```

Consistency rule: `verdict = GO` is incompatible with any issue of severity `CRITICAL` or `HIGH`.

3.3 CODE_PRIMER

```
{
  "structure": { "directories": "tree", "modules": "string" },
  "rules": [
    { "category": "NAMING|STRUCTURE|ERROR_HANDLING|CONCURRENCY|TESTING|DOCUMENTATION|LOGGING|OTHER",
      "rule": "string", "examples": ["string"], "enforcement": "MANDATORY|RECOMMENDED" }
  ],
  "contracts": [
    { "type": "API|INTERFACE|DATA_STRUCTURE|EVENT", "definition": "string", "validation": ["string"] }
  ],
  "anti_patterns": [
    { "pattern": "string", "why": "string", "alternative": "string" }
  ],
  "ai_rules": {
    "system_instructions": ["string"], "tool_contracts": ["string"], "interaction_logging": ["string"
  }
}
```

3.4 TEST_STRATEGY

```
{
  "coverage": { "minimum_total": "percentage", "minimum_unit": "percentage", "minimum_integration": "p
  "layers": [ { "layer": "UNIT|INTEGRATION|E2E|PERFORMANCE|SECURITY", "purpose": "string", "tools": ["
  "cases": [
    { "id": "string", "name": "string", "type": "UNIT|INTEGRATION|E2E", "component": "string",
      "scenario": "string", "expected_outcome": "string", "priority": "CRITICAL|HIGH|MEDIUM|LOW" }
  ],
  "doubles_and_data": [ { "name": "string", "purpose": "string", "structure": "string" } ],
  "edge_cases": [ { "case": "string", "approach": "string" } ],
  "behaviour_evaluation": {
    "applies": "true | false",
    "sets": ["identifier and version"],
    "criteria": [ { "metric": "string", "threshold": "value" } ],
    "adversarial": ["string"],
    "bias": ["string"]
  }
}
```

3.5 IMPLEMENTATION

```
{
  "files_created": [ { "path": "string", "purpose": "string", "key_functions": ["string"] } ],
```

```
  "files_modified": [ { "path": "string", "changes": "string", "reason": "string" } ],
  "dependencies_added": [ { "name": "string", "version": "string", "licence": "string", "reason": "str
  "compliance": { "follows_primer": "true | false", "follows_plan": "true | false" },
  "next_step": "TEST_IMPLEMENTATION"
}
```

Every deviation goes in the common block `deviations`, unresolved.

3.6 TEST_IMPLEMENTATION

```
{
  "tests": [ { "file": "string", "cases": ["id"], "type": "UNIT|INTEGRATION|E2E" } ],
  "doubles_and_data": [ { "file": "string", "purpose": "string" } ],
  "execution": { "command": "string", "variables": ["string"], "requirements": ["string"] },
  "coverage_obtained": { "total": "percentage", "unit": "percentage", "integration": "percentage" },
  "compliance": { "follows_strategy": "true | false" }
}
```

3.7 AUDIT_TESTS

```
{
  "coverage": { "actual_total": "percentage", "expected_total": "percentage", "actual_unit": "percenta
  "quality": [ { "aspect": "ASSERTIONS|EDGE_CASES|ERROR_PATHS|DOUBLES|DATA|CLARITY", "status": "EXCELL
  "case_review": [ { "id": "string", "implemented": "true | false", "quality": "EXCELLENT|GOOD|ADEQUA
  "verdict": "GO|GO_WITH_CHANGES|NO_GO",
  "justification": "string",
  "required_fixes": [ { "issue": "string", "severity": "CRITICAL|HIGH|MEDIUM|LOW", "recommendation":
}
```

3.8 AUDIT_CODE

```
{
  "fidelity_to_plan": { "score": "0-100", "deviations": [ { "deviation": "string", "justified": "true
  "primer_compliance": { "score": "0-100", "violations": [ { "rule": "string", "location": "string",
  "technical_risks": [ { "risk": "string", "category": "PERFORMANCE|SECURITY|RELIABILITY|MAINTAINABILI
  "technical_debt": [ { "item": "string", "impact": "string", "recommendation": "string" } ],
  "dependencies": [ { "name": "string", "exists": "true | false", "provenance": "string", "vulnerabil
  "findings": [ { "finding": "string", "severity": "CRITICAL|HIGH|MEDIUM|LOW", "location": "string",
  "verdict": "GO|GO_WITH_CHANGES|NO_GO",
  "justification": "string",
```

```
  "next_step": "VERSION|FIX_PRIMERS|PLAN"
}
```

3.9 FIX_PRIMERS

```
{
  "fixes": [
    { "id": "string", "finding": "id of the finding", "problem": "string", "file": "string", "locatio
      "change": "string", "justification": "string", "impact": "string", "regression_risk": "LOW|MEDI
  ],
  "files_modified": ["string"],
  "tests_updated": ["string"],
  "overall_regression_risk": "LOW|MEDIUM|HIGH",
  "next_step": "AUDIT_CODE"
}
```

3.10 VERSION

```
{
  "version": { "number": "MAJOR.MINOR.PATCH", "type": "MAJOR|MINOR|PATCH", "justification": "string"
  "changes": [ { "category": "ADDED|CHANGED|DEPRECATED|REMOVED|FIXED|SECURITY", "description": "strin
  "breaking_changes": [ { "change": "string", "impact": "string", "migration": "string" } ],
  "migration": { "required": "true | false", "steps": ["string"], "effort": "string" },
  "deployment": { "prerequisites": ["string"], "plan": "string", "rollback": "string", "rollback_test
  "ai_system": { "model": "string", "parameters": "string", "instructions_version": "string", "evalua
}
```

04

Complementary cycle contracts

4.1 DOCUMENT_EXISTING (L1)

```
{
  "code_location": "string",
  "behaviour": { "purpose": "string", "functionality": ["string"], "inputs": ["string"], "outputs": [
  "components": [ { "name": "string", "type": "CLASS|FUNCTION|MODULE|SERVICE", "responsibility": "str
  "dependencies": { "internal": ["string"], "external": ["string"] },
  "diagrams": [ { "type": "FLOW|ARCHITECTURE|SEQUENCE", "file": "string", "description": "string" } ]
```

```
  "unverified": ["statements inferred without execution evidence"],
  "risks": ["string"]
}
```

4.2 DISCOVERY_LEGACY (L2)

```
{
  "change_objective": "string",
  "areas_affected": [ { "component": "string", "impact": "CRITICAL|HIGH|MEDIUM|LOW", "reason": "string
  "modification_risks": [ { "risk": "string", "severity": "CRITICAL|HIGH|MEDIUM|LOW", "mitigation": "
  "tests": { "existing": ["string"], "coverage": "percentage", "gaps": ["string"], "characterisation_
  "edge_cases": [ { "case": "string", "current_behaviour": "string", "has_test": "true | false" } ],
  "recommendations": ["string"],
  "next_step": "PLAN"
}
```

4.3 DEBUG (D1)

```
{
  "incident": { "date": "ISO 8601", "description": "string", "environment": "DEVELOPMENT|STAGING|PRODU
  "mode": "STATIC|WITH_EXECUTION",
  "evidence": { "files": ["string"], "commands": ["string"], "logs": ["string"], "traces": ["string"]
  "root_cause": { "summary": "string", "explanation": "string", "category": "LOGIC|CONFIGURATION|CONCU
  "verdict": "CODE|CONFIGURATION|INFRASTRUCTURE",
  "actions": [ { "action": "string", "priority": "CRITICAL|HIGH|MEDIUM|LOW", "next_step": "PLAN|HOTFIX
}
```

4.4 HOTFIX (U2–U7)

```
{
  "priority": "P1|P2",
  "fix_type": "TEMPORARY|DEFINITIVE",
  "root_cause": "summary of the D1 report",
  "simplified_plan": { "change": "string", "scope": "string", "risk": "string", "minimal_tests": ["str
  "accelerated_review": { "verdict": "GO|GO_WITH_CHANGES|NO_GO", "justification": "string" },
  "implementation": { "files": ["string"], "changes": "string", "tests_executed": ["string"] },
  "deployment": { "date": "ISO 8601", "method": "string", "emergency_approval": "person", "rollback_a
  "monitoring": { "duration_minutes": "integer", "indicators": ["string"], "status": "STABLE|UNSTABLE
  "post_mortem": { "timeline": ["string"], "root_cause": "string", "effectiveness": "SOLVED|MITIGATED
```

```
  "definitive_plan": { "required": "true | false", "topic": "string" }
}
```

4.5 SECURITY_AUDIT (S1)

```
{
  "scope": { "files": ["string"], "components": ["string"] },
  "vulnerabilities": [
    { "id": "string", "severity": "CRITICAL|HIGH|MEDIUM|LOW",
      "category": "INJECTION|XSS|AUTHENTICATION|AUTHORISATION|CRYPTOGRAPHY|SECRETS|INPUT_VALIDATION|A
      "reference": "entry in the reference list", "description": "string", "location": "string", "exp
  ],
  "secrets": { "found": "true | false", "locations": ["string"], "recommendation": "string" },
  "vulnerable_dependencies": [ { "name": "string", "version": "string", "vulnerability": "identifier"
  "deployment_decision": { "critical": "integer", "high": "integer", "medium": "integer", "low": "int
  "proposed_residual_risks": [ { "risk": "string", "severity": "MEDIUM|LOW", "recommendation": "string
  "verdict": "GO|GO_WITH_CHANGES|NO_GO"
}
```

Acceptance of the residual risks is not part of the contract: a person writes it in the validation record.

4.6 BEHAVIOUR_EVAL (E)

```
{
  "set": { "id": "string", "version": "string", "cases": "integer" },
  "results": [ { "metric": "string", "value": "number", "threshold": "number", "meets": "true | false"
  "failures": [ { "category": "string", "count": "integer", "examples": ["string"] } ],
  "adversarial": [ { "test": "string", "outcome": "PASSED|FAILED", "detail": "string" } ],
  "bias": [ { "test": "string", "segments": ["string"], "difference": "number", "threshold": "number"
  "human_review_sample": { "cases": "integer", "selection_criterion": "string" },
  "verdict": "GO|GO_WITH_CHANGES|NO_GO",
  "justification": "string",
  "return_phase": "PLAN|IMPLEMENTATION|null"
}
```

05

Format of the contexts

The contexts are not phase artefacts and are written in Markdown with fixed sections (document 02):

```
# GLOBAL_CONTEXT (version n)
## Artefacts
## Tests
## Versioning
## Baseline security
## Data and model providers
## Prohibitions
## Parameters of the complementary cycles
## Approved models

# PROJECT_CONTEXT (version n)
## Technology
## Architecture
## Domain rules
## Compliance
## Team conventions
## Deployment
## Justified exceptions
```

06

Automatic validation

An artefact validator checks, before the human validation:

Check	Outcome if it fails
Common block present and complete, including `model`.	Incomplete artefact.
All the mandatory sections of the phase's contract.	Incomplete artefact.
Verdict, severity and priority values within the scales.	Incomplete artefact.
`verdict = GO` with open critical or high issues.	Self-approval.
`topic` in the correct format and matching the folder.	Incomplete artefact.
`iteration` consecutive with the previous artefact of the same phase.	Incomplete artefact.
Cross-references (`finding` , `case` , `issue`) existing in the previous artefact.	Incomplete artefact.

The validator **does not judge the content**: that is for the AI Reviewer and the person. A serverless tool with this function is part of the next steps of SPAD (document 08).

07

Related documents

Document	Relationship
document 01 · Operating guide	Phases that produce each artefact.
document 02 · Contexts, topics and artefact register	Model record and validation record.
document 03 · Validation policy	Causes that the automatic validator detects.
document 04 · Complementary cycles	Contracts L1, L2, D1, U and S1.
document 05 · Systems that include AI	Contract of the behaviour evaluation and `ai_system` fields.
document 06 · Instructions by phase	Instructions that require these outputs.

08

Version control

Version	Date	Changes
0.1	19-09-2026	First version. Rewrites the previous specification: common block with model record and deviations; single scales; human validation record; main cycle contracts with dependencies and licences; complementary cycle contracts; behaviour evaluation contract; format of the contexts; checks of the automatic validator.