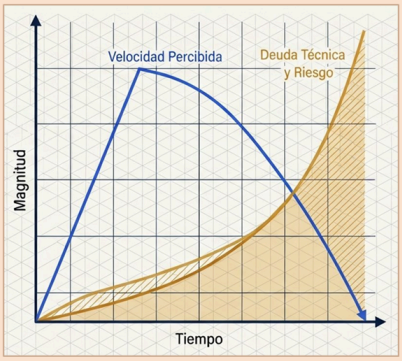
SPAD IN 5 SLIDES

The problem: the illusion of speed when coding with AI

AI generates plausible code very fast. Without a method, that speed hides decisions, errors and risk.



- **What you see** Code that works in the demonstration within hours, even though nobody designed the solution.
- **What you do not see** Design decisions taken by the AI inside the code, few tests and vulnerabilities that nobody reviewed.
- **The effect** Perceived speed falls while technical debt and risk grow. Conceptual chart, no data.
- **The cause** It is not an AI problem: it is a method problem that AI amplifies. Illustration in Spanish.

A design error found in the plan review costs a conversation; found in production it costs an incident and sometimes a customer.

→ The problem it solves → What SPAD is

SPAD · Version 0.x, under review · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 1 / 5

Notes

SPAD stands for Structured Prompt-Driven Engineering. It is a way of working to build software with the help of AI in a controlled manner, so that the work can be reviewed, reproduced and audited.

The chart is conceptual and has no data. It shows an intuition: at first, coding with AI looks much faster. Over time, if nobody designs, nobody reviews and nothing is written down, perceived speed falls while technical debt and risk grow.

Typical symptoms that SPAD addresses:

- The code works in the demonstration and fails in production, because code was requested until it «worked».
- Nobody knows why the system is built this way, because the AI took the decisions inside the code.
- Vulnerabilities appear, because the same AI that wrote the code reviewed it, or nobody did.
- An audit asks for evidence of the development and there is none: the work stayed in conversations with the AI.

A simple example: a team asks an AI assistant for a payments screen. It works in one afternoon. Nobody has decided how retries are handled or who can see the data. The failure appears months later, in front of a customer.

Why it matters to an executive: speed without control moves the cost into the future, as incidents, rework and reputational or regulatory risk. SPAD does not promise improvement figures; each organisation measures its effect against its own starting position.

Question for the classroom: who in your organisation could explain today why an application generated with AI is built the way it is?

SOURCES AND LINKS

SPAD IN 5 SLIDES

- **SPAD 00 · What SPAD is and what it is for, section 2** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#the-problem-it-solves
- **SPAD 00 · When to use SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#when-to-use-spad
- **The problem it solves** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#the-problem-it-solves
- **What SPAD is** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html



Notes

SPAD has three properties. It is sequential: work advances through phases in a fixed order. It is blocking: no phase starts until the previous one has produced its result and a person has validated it. It is auditable: each phase leaves a written artefact (a plan, a review report, a test strategy, a release record) that is kept.

The first three cards are the three pillars: think, validate and execute. The other three are non-negotiable rules. The operational guide sets out twelve, among them:

- Every decision is explicit: there are no design decisions hidden in the code.
- Minimal fixes: surgical and justified, never disguised rewrites.
- Mandatory tests, with minimum coverage set before building.
- Security by design: sensitive code undergoes a security review and critical and high findings block.
- Model recorded: each artefact states which model and which instructions produced it.

The principle that orders everything: compliance with the flow is more important than the apparent quality of the result. A technically correct response that breaks the rules of its phase is discarded in its entirety and the phase is repeated. Example: a plan is requested and the AI adds «an example implementation» of five hundred lines. Even if the code is good, the response is invalid.

Why it matters to an executive: if a response that breaks the rules is accepted «just this once» because the code works, the process stops existing. Discipline is what makes the evidence reliable.

Question for the classroom: which rule on this slide would a team in a hurry find hardest to follow, and why?

TERMS ON THIS SLIDE

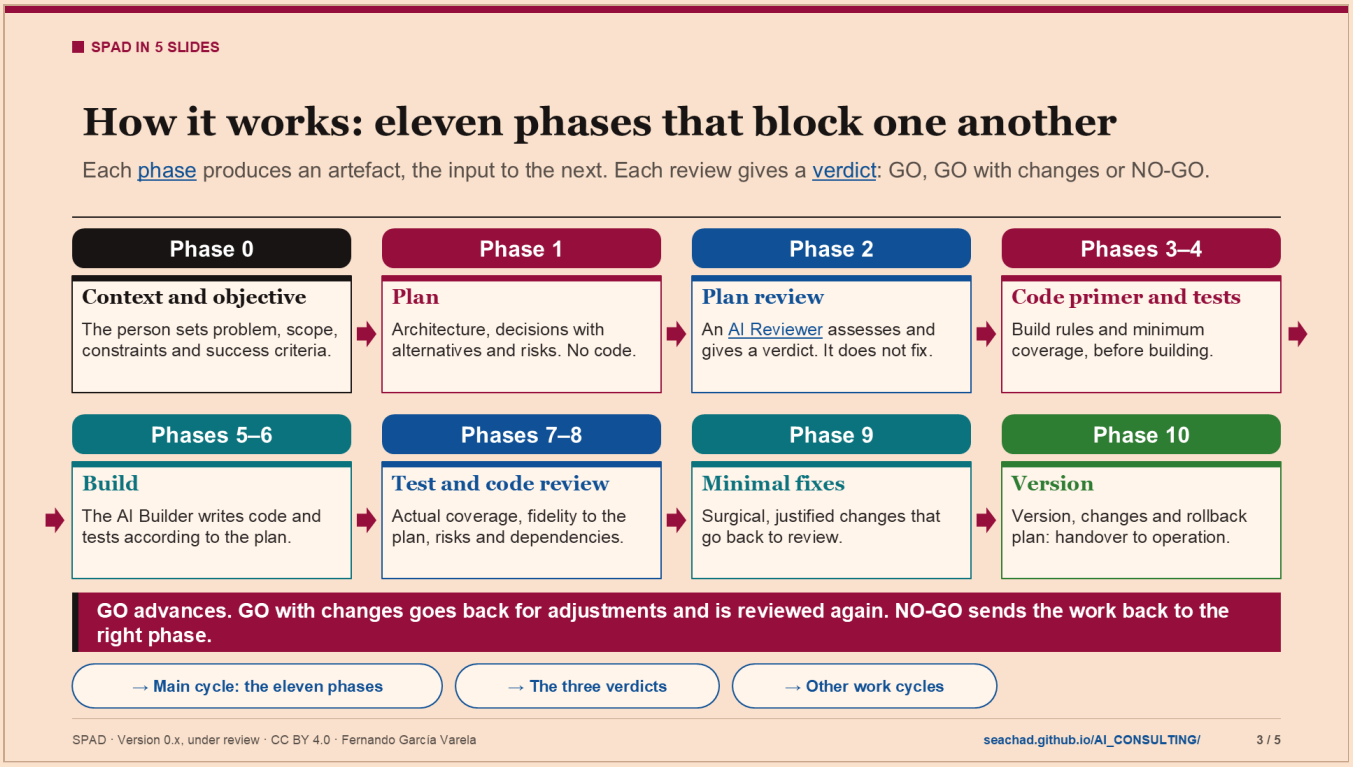
- **Blocking execution principle**: SPAD's central rule: no phase starts until the previous one has delivered its artefact and received a favourable verdict. It prevents AI from building on a plan nobody has reviewed. The link opens the principle in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#the-blocking-execution-principle

SPAD IN 5 SLIDES

- **SPAD verdicts** (GO / NO-GO): Every SPAD review ends with one of three verdicts: GO (continue), GO WITH CHANGES (continue after fixing what was flagged) or NO-GO (return to the relevant phase). A single vocabulary across all reviews avoids interpretation. The link opens the verdicts in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#verdicts
- **SPAD main cycle** (fases 0–10): SPAD's eleven phases: context, plan, plan review, code guide, test strategy, implementation, testing, test review, code review, fixes and release management. Each one leaves an artefact that is reviewed. The link opens the cycle in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#main-cycle-phases-0-to-10
- **SPAD Lite**: A reduced version of SPAD for small, low-risk changes, with only three artefacts. It keeps review and human validation, but with fewer steps. It avoids loading a minor change with the whole cycle. The link opens the reduced version in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#reduced-version-spad-lite
- **Human code review**: In SPAD, code generated by AI is always reviewed by a person before it is merged, with a second review for sensitive code, autonomous agents and critical production. AI helps to build, but it does not approve itself. The link opens the rule in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#3-1-human-code-review
- **Validation policy**: A SPAD document that sets what a person must validate, when and how rigorously, depending on the risk of the change, and what evidence is kept of that validation. It is the guarantee that AI does not decide alone what goes into the code. The link opens the document. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/03_SPAD_Politica_de_validacion.html

SOURCES AND LINKS

- **SPAD 00 · The principle that orders everything** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#the-principle-that-orders-everything
- **SPAD 01 · Non-negotiable operating rules** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#non-negotiable-operating-rules
- **SPAD 03 · Validation policy, examples of decisions** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/03_SPAD_Politica_de_validacion.html#examples-of-decisions
- **The principle that orders everything** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#the-principle-that-orders-everything
- **The twelve non-negotiable rules** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#non-negotiable-operating-rules



Notes

The main cycle is used to build a new feature and has eleven phases, from 0 to 10. The slide groups some of them so they can be read at a glance.

- Phase 0, context and objective: without success criteria there is no phase 1.
- Phase 1, plan: the AI Planner designs without writing code and records the alternatives it discarded.
- Phase 2, plan review: a different AI assesses coupling, scalability, security and other aspects. It assesses; it does not fix.
- Phases 3 and 4, code primer and test strategy: these are the most often skipped and the ones that protect most value. The primer stops the builder from deciding; the strategy prevents tests from being written at the end just to confirm what already exists.
- Phases 5 and 6, build: if the plan does not foresee a situation, the AI Builder does not decide; it flags it and a person resolves it.
- Phases 7 and 8, reviews: they include the dependencies added and their licence.
- Phase 9, fixes: each fix goes back to the code review.
- Phase 10, version: the system is ready to operate, with a deployment and rollback plan.

The verdicts are always the same: GO (work advances), GO with changes (adjust and review again) and NO-GO (redo). A NO-GO in the plan review sends work back to phase 0, because the problem is badly framed. A well-founded NO-GO is not a failure: it is the process working.

There are also cycles for legacy code, debugging, urgent fixes and security. In an emergency the phases are accelerated, but review and tests remain mandatory.

Why it matters to an executive: the cycle brings expensive decisions forward to the moment when they are still cheap.

TERMS ON THIS SLIDE

SPAD IN 5 SLIDES

- **SPAD main cycle** (fases 0–10): SPAD's eleven phases: context, plan, plan review, code guide, test strategy, implementation, testing, test review, code review, fixes and release management. Each one leaves an artefact that is reviewed. The link opens the cycle in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#main-cycle-phases-0-to-10
- **SPAD verdicts** (GO / NO-GO): Every SPAD review ends with one of three verdicts: GO (continue), GO WITH CHANGES (continue after fixing what was flagged) or NO-GO (return to the relevant phase). A single vocabulary across all reviews avoids interpretation. The link opens the verdicts in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#verdicts
- **SPAD roles**: The roles in SPAD: the orchestrator (a person who leads the cycle and validates) and several AIs with separate functions: planner, reviewer, builder, fixer, analyst and diagnostic. Whoever builds does not review what they build. The link opens the SPAD presentation document, which describes each role. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html

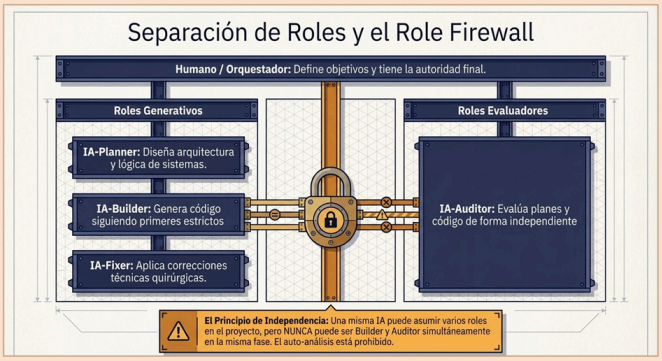
SOURCES AND LINKS

- **SPAD 00 · Other work cycles** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#other-work-cycles
- **Other work cycles** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#other-work-cycles

SPAD IN 5 SLIDES

Who does what: AI produces and reviews, the person decides

SPAD separates the roles. Whoever builds does not approve their own work, and a person always has the final say.



- **Orchestrator** A person defines the objective, loads the contexts, validates each output and has the final say.
- **AI roles** AI roles: Planner, Builder and Fixer produce; the Reviewer assesses. Never two roles in the same phase.
- **Human review** A person does the human code review. In sensitive code, another person who did not orchestrate it also reviews.
- **Contexts and records** Contexts set the rules before starting; records per artefact note model and validation. Image in Spanish.

Whoever does the work does not approve it. SPAD applies this classic control between artificial intelligences too.

→ Roles and human review | → Contexts, topics and records | → Self-approval: a cause of invalidity

SPAD · Version 0.x, under review · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 4 / 5

Notes

SPAD distinguishes logical roles, not specific people or tools. The same AI may take on several roles in a piece of work, but never two in the same phase, and the AI that reviews an artefact is never the one that produced it.

- The Orchestrator is a person. They define the objective, load the contexts, set the topic, validate each output and decide whether to advance. They cannot delegate validation to the AI that produced the output.
- The AI Planner designs and does not write code. The AI Builder writes code and does not take design decisions. The AI Fixer applies minimal fixes and does not redesign.
- The AI Reviewer assesses and does not fix. The illustration labels it «IA-Auditor»: it is the AI Reviewer. It is an AI, not a person: its assessment does not replace human review.

Two artificial intelligences can make the same mistake. That is why a competent person also reviews the code before it is merged. In authentication, payments, personal data, agents or critical systems, a second person who did not orchestrate the generation also reviews.

Contexts: before any phase, the organisation's rules (for example, «no real personal data in instructions to external providers») and the project's rules are loaded. Topic: a short name that groups all the artefacts of a piece of work in one folder. Records: each artefact states which model produced it, who validated it and with what result; and each discarded response is logged.

Why it matters to an executive: this is the segregation of duties of any serious process. The person who answers to the organisation for the code being correct is a person, and that responsibility is not delegated to a tool.

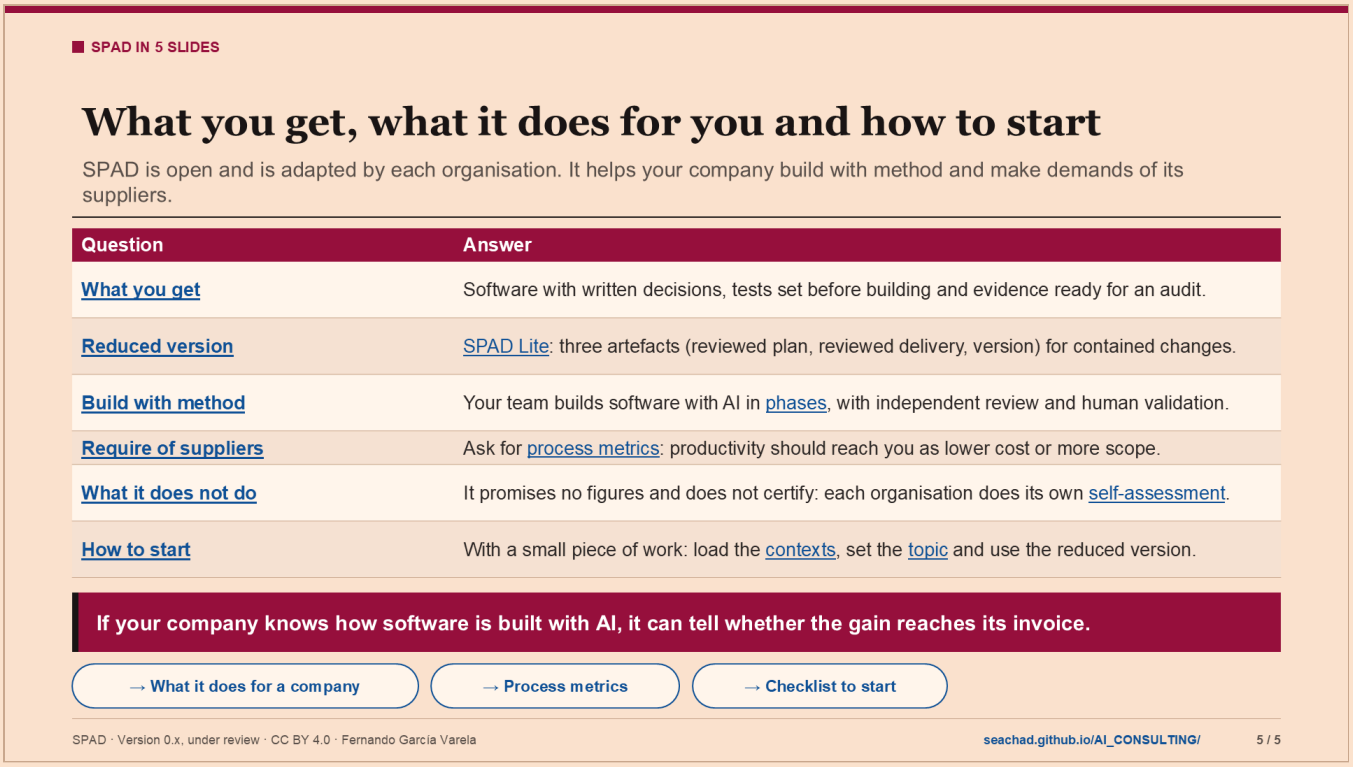
Question for the classroom: which data of your company should never reach an external AI tool?

TERMS ON THIS SLIDE

- **SPAD roles**: The roles in SPAD: the orchestrator (a person who leads the cycle and validates) and several AIs with separate functions: planner, reviewer, builder, fixer, analyst and diagnostic. Whoever builds does not review what they build. The link opens the SPAD presentation document, which describes each role. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html
- **Human code review**: In SPAD, code generated by AI is always reviewed by a person before it is merged, with a second review for sensitive code, autonomous agents and critical production. AI helps to build, but it does not approve itself. The link opens the rule in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#3-1-human-code-review
- **SPAD contexts**: Rule documents the AI must respect in each phase: the global context (organisation rules, data, approved models) and the project context. Exceptions are assessed by the reviewer AI and approved by a person. They ensure the AI does not work on its own assumptions. The link opens the contexts. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#contexts
- **Artefact records**: Each SPAD artefact carries three records: the model record (supplier, model, version, instruction and contexts used), the human validation record (who reviewed it and when) and the violation log (rules not complied with). They make the process auditable. The link opens the records. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#4-1-model-record

SOURCES AND LINKS

- **SPAD 02 · Global context** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#2-1-global-context
- **SPAD 02 · Records per artefact** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#records-per-artefact
- **SPAD 03 · Validation policy: self-approval** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/03_SPAD_Politica_de_validacion.html#3-5-self-approval
- **Roles and human review** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#roles
- **Contexts, topics and records** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html
- **Self-approval: a cause of invalidity** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/03_SPAD_Politica_de_validacion.html#3-5-self-approval



SPAD IN 5 SLIDES

What you get, what it does for you and how to start

SPAD is open and is adapted by each organisation. It helps your company build with method and make demands of its suppliers.

Question	Answer
What you get	Software with written decisions, tests set before building and evidence ready for an audit.
Reduced version	SPAD Lite: three artefacts (reviewed plan, reviewed delivery, version) for contained changes.
Build with method	Your team builds software with AI in phases, with independent review and human validation.
Require of suppliers	Ask for process metrics: productivity should reach you as lower cost or more scope.
What it does not do	It promises no figures and does not certify: each organisation does its own self-assessment.
How to start	With a small piece of work: load the contexts, set the topic and use the reduced version.

If your company knows how software is built with AI, it can tell whether the gain reaches its invoice.

→ What it does for a company | → Process metrics | → Checklist to start

SPAD · Version 0.x, under review · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 5 / 5

Notes

What you get: decisions taken before code is written, one artefact per phase, independent reviews and traceable fixes. Everything is kept in the topic folder, which is the unit of audit. The organisation can carry out a self-assessment of conformity, by a person independent of those who built. It is not a certification: nobody issues or endorses it from outside.

Reduced version (SPAD Lite): it groups the phases into three artefacts. It suits contained changes, with no personal data or money, and a small team. What is never cut is the separation between the AI that builds and the one that reviews, human validation, the validation policy and the topic register. Sensitive code, agents with high autonomy and critical systems require the complete version.

What it does for your company (the author's position, with no figures). First use: building your own software with AI with method, with phases, reviewed artefacts, independent review and human validation. Second use, as a client: a company that knows how software is built with AI can require its suppliers and integrators to pass the productivity gain on to it, as lower cost or more scope for the same price. It can ask them for what SPAD already defines, without inventing anything else: the artefact of each phase, the record of the model and of human validation, the verdict of each review and the process metrics (time per phase, iterations to GO, escaped defects, rework). Why it matters: AI changes what it costs to build software; without that knowledge, the company does not know whether the change reaches its invoice or stays with the supplier. SPAD sets no percentage: it lets you negotiate with data.

A separate framework. SPAD deals only with how software is built; it is not the governance of the company's AI. Whoever wants to govern what is built, with what risk and how its value is measured has a separate framework, SEVEN-G. The only bridge is its document 53, and SPAD verdicts do not replace its decision gates.

How to start: choose a small piece of work; load the global and project contexts; set the topic; assign an AI Reviewer different from the builder; open the validation record; and check which information may be sent to each provider. At first it will be slower; measure rework and incidents against your starting position.

SPAD is published under CC BY 4.0 (content) and MIT (code): it can be used and adapted, citing the authorship.

TERMS ON THIS SLIDE

- **SPAD Lite**: A reduced version of SPAD for small, low-risk changes, with only three artefacts. It keeps review and human validation, but with fewer steps. It avoids loading a minor change with the whole cycle. The link opens the reduced version in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#reduced-version-spad-lite
- **SPAD main cycle** (fases 0–10): SPAD's eleven phases: context, plan, plan review, code guide, test strategy, implementation, testing, test review, code review, fixes and release management. Each one leaves an artefact that is reviewed. The link opens the cycle in the operating guide. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/01_SPAD_Guia_operativa.html#main-cycle-phases-0-to-10
- **Process metrics**: Indicators measured by a team using SPAD, such as how many reviews end in NO-GO, how many defects reach production or how long a topic takes. SPAD promises no figures: it proposes measuring them and setting your own targets. The link opens the metrics. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/08_SPAD_Autoevaluacion_y_metricas.html#process-metrics
- **Conformity self-assessment**: A review with which a team checks whether it applies SPAD as it claims: which phases it follows, which records it keeps and what fails. It is valid for twelve months and nobody outside grants it. It helps improve the process with data rather than impressions. The link opens the self-assessment. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/08_SPAD_Autoevaluacion_y_metricas.html#what-the-self-assessment-is
- **SPAD contexts**: Rule documents the AI must respect in each phase: the global context (organisation rules, data, approved models) and the project context. Exceptions are assessed by the reviewer AI and approved by a person. They ensure the AI does not work on its own assumptions. The link opens the contexts. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#contexts
- **SPAD topics**: A topic is SPAD's unit of work: a feature or a change that goes through the full cycle with its own artefact folder. It lets you follow from start to finish what was asked, planned and built, and who validated it. The link opens its definition. https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#topics

SOURCES AND LINKS

- **SEVEN-G 53 · Building solutions with AI (the only bridge)** https://seachad.github.io/AI_CONSULTING/SEVEN-G/html/en/53_SEVEN-G_Construccion_de_soluciones_con_IA.html
- **SPAD 00 · What it does for a company** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#what-it-does-for-a-company
- **What it does for a company** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/00_SPAD_Que_es_y_para_que_sirve.html#what-it-does-for-a-company
- **Checklist to start** https://seachad.github.io/AI_CONSULTING/SPAD/html/en/02_SPAD_Contextos_temas_y_registro.html#checklist-before-starting

LEGAL NOTICE

Legal notice: a generic, free methodological aid provided "as is" for information purposes; it is not legal advice and has not been legally reviewed for any case. Each organisation adapts it and is responsible for its legal review and its regulatory compliance; it may not reflect recent regulatory changes. To the fullest extent permitted by law, the author accepts no liability whatsoever for the effects of its application. It does not grant certification of any kind. Details in document 93.

CC BY 4.0 (content) · MIT (code) · Fernando García Varela · seachad.github.io/AI_CONSULTING/