

---

# Spec Driven Development from Idea to Production

How structured context turns coding agents into reliable delivery systems

Dmitry Valetin

Founder of [Spexus.ai](https://spexus.ai)

[dmitry.valetin@gmail.com](mailto:dmitry.valetin@gmail.com)

<https://www.linkedin.com/in/dvaletin/>



---

# Agenda

**01** SDD workflow inside the SDLC

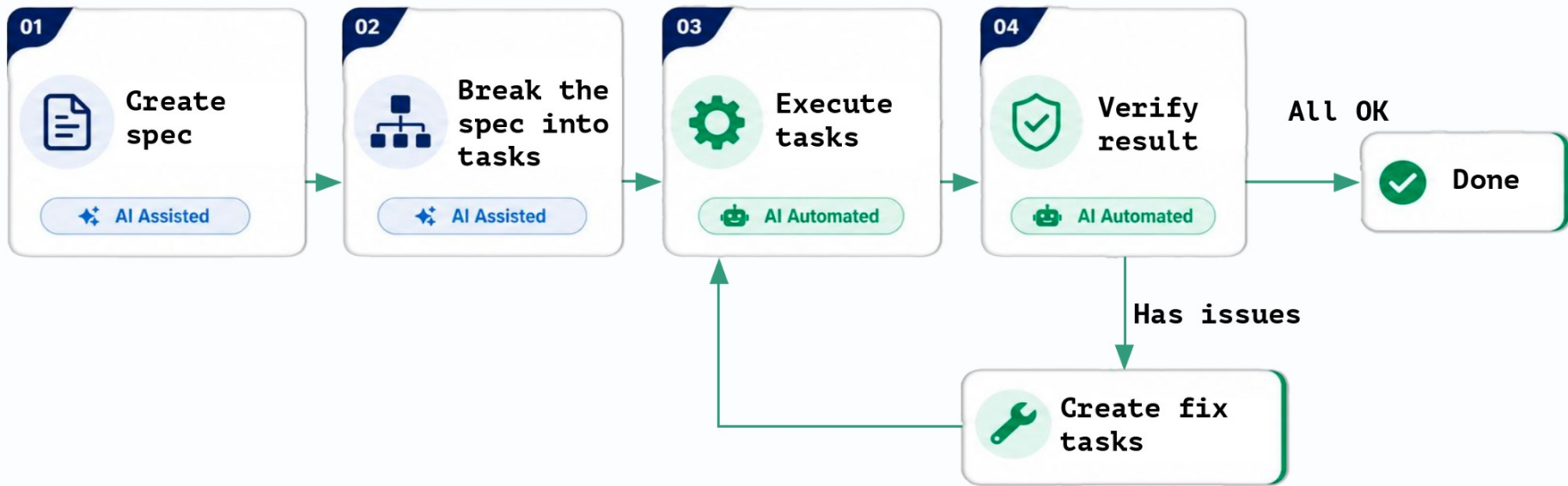
**02** Context management for LLMs

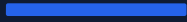
**03** SDD as a context-transfer system for coding agents

**04** Real example: MCP JSON-RPC API Handler

---

# SDD Workflow in the SDLC





# Context Management

---

# Why context matters for LLMs

## **No built-in memory**

LLMs do not know your product, users, architecture, or delivery constraints by default.

## **Unsafe confidence**

Missing context produces plausible code, wrong abstractions, and weak verification.

## **Useful context**

Intent, constraints, interfaces, and definition of done must be explicit.

## **Operational payoff**

Better context reduces token waste, shortens review loops, and improves results.

---

# SDD as Context Transfer

---

# Product context for a coding agent

Agents do not start with product memory.

SDD transfers intent through a structured chain instead of hoping context appears inside a prompt.

**Epic = intent of the increment and expected business change**

**User Story = a concrete slice of user value**

**Requirement = a guardrail and implementation constraint**

**Acceptance Criterion = a verifiable success point**

---

# Technology context and system design

## Standards

Reusable rules shared across epics or scoped to a single epic.

## System design

Architecture, boundaries, data flow, and implementation hints.

## Delivery contract

Where code should live, which patterns to use, and how to run tests.

*Together with product context, this makes agent output explainable, reviewable, and easier to verify.*

---

# CASE: Create MCP Server API

# Epic as intent + technology context

## Epic answers:

- why this increment exists
- what system change is in scope
- which terminology and technical direction the team shares

## ← MCP JSON-RPC API Handler

Implement the backend MCP entry point for the product-requirements system as a secure JSON-RPC 2.0 API. The endpoint shall expose requirements-management resources, execute mapped MCP tools, and serve prompt-related capabilities through a protocol-compliant contract aligned with the MCP architecture used in this project.

### Scope

- JSON-RPC 2.0 request handling for MCP clients
- PAT-based authentication and authorization
- MCP resources for requirements-management entities
- MCP tools for CRUD, search, and prompt management
- Prompt discovery and retrieval flows
- REST support for prompt administration
- Audit logging, validation, and structured error handling

### Out of Scope

- Non-MCP transport protocols beyond the supported HTTP entry point
- Frontend UX for prompt administration
- New business domains outside requirements, prompts, and MCP integration

### Glossary

- MCP: Model Context Protocol
- JSON-RPC: JSON-RPC 2.0 request/response protocol
- PAT: Personal Access Token used for API authentication
- Resource: Read-oriented MCP object exposed through `resources/read`
- Tool: Executable MCP operation exposed through `tools/call`
- Prompt: MCP prompt definition retrievable through prompt methods
- Active Prompt: The single prompt definition marked as active for runtime use

# Product context: stories, requirements, acceptance criteria

## Traceability layer:

- user stories define the user-facing slice
- requirements constrain implementation choices
- acceptance criteria define how success is verified

The screenshot displays a user story titled "Provide MCP tools for system prompts" with a status of "Backlog" and a priority of "High". The story description is: "As an MCP client with administrator privileges, I want to manage system prompts through MCP tools, so that I can perform CRUD operations and activate prompts via the MCP protocol." It shows 1 Requirement and 5 AC (Acceptance Criteria).

**REQUIREMENTS**

+ Add requirement

SP-REQ-014 Draft High Functional

**Expose admin-only MCP prompt tools**

THE system SHALL expose MCP prompt-management tools with administrator-only mutation access.

**ACCEPTANCE CRITERIA**

+ Add acceptance criteria

- SP-AC-078  
Given MCP prompt tools are supported, when the tool catalog is exposed, then the system provides create\_prompt, update\_prompt, delete\_prompt, activate\_prompt, list\_prompts, an
- SP-AC-079  
Given an MCP prompt tool is executed, when the system processes the request, then it requires Administrator privileges for CRUD operations, validates prompt content as plain text, a
- SP-AC-080  
Given an MCP prompt tool succeeds, when the handler returns the result, then it returns a structured result with a content array.
- SP-AC-081  
Given a non-administrator attempts an MCP prompt CRUD tool, when the system evaluates authorization, then the handler returns a JSON-RPC error with code -32002.

---

# Task slicing, implementation, verification

## Task slicing

Split the epic into traceable tasks. Each task becomes an agent prompt.

## Coding

Run the coding orchestrator to implement tasks one by one.

## Verification

Run a verification **sub-agent(\*)** to review the result against the spec.

## Fix gaps

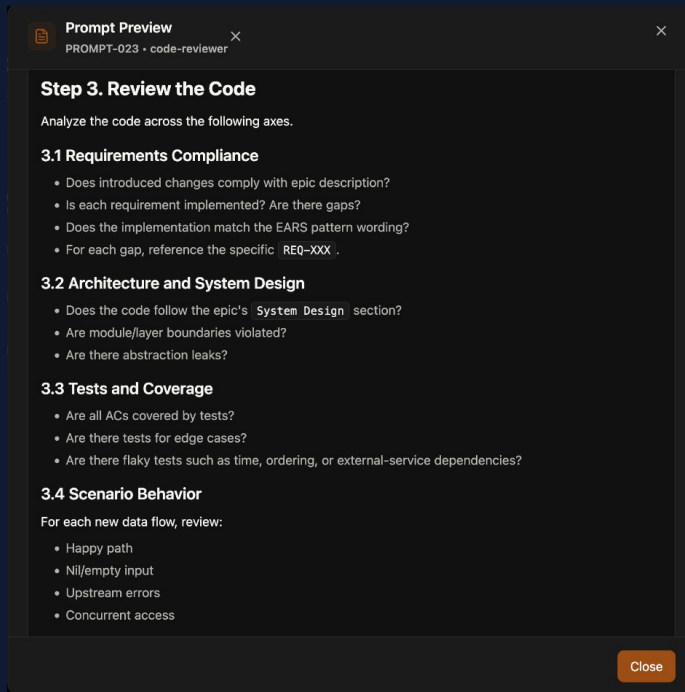
Each gap becomes a fix task. Rerun orchestration until the gaps are closed.

*\* A sub-agent does not share context with the orchestrator, which reduces the risk of hallucinations and context poisoning.*

# Verification as a first-class contract

The definition of done becomes explicit:

- each AC should map to a test or verification step
- review asks whether the implementation satisfies the spec
- evidence stays grounded in spec, not intuition



---

# What's next?

## Build

**Build the product  
with CI/CD**

## Test

**Run E2E,  
integration, and  
unit tests  
Run the linter**

## Deploy

**Deploy to  
production with  
CI/CD**

*This closes the loop from specification to production release.*

---

# Grazie mille. Ci sono domande?

Dmitry Valetin

Founder of [Spexus.ai](https://spexus.ai)

[dmitry.valetin@gmail.com](mailto:dmitry.valetin@gmail.com)

<https://www.linkedin.com/in/dvaletin/>

