Agile-Centric Prompt Engineering Framework

A Practical Guide for AI-Enhanced Agile Practices

What is this framework?

A structured approach for agile professionals to create effective AI prompts that deliver valuable results.

Who should use it?

Scrum Masters, Product Owners, Agile Coaches, Developers, and anyone applying agile practices.

Why does it work?

It applies core agile principles—iterative improvement, collaboration, and continuous adaptation—to AI interactions.

What's the outcome?

High-quality, actionable AI outputs that solve real agile challenges while saving time and improving quality.

Note:

Created by **Stefan Wolpers** and **Holger Dierssen** with the help of **ChatGPT** and **Claude** in March 2025.

A. Framework Overview

This framework is organized into three tiers of importance:

🔴 Must-Have Elements (Core)

Essential components for any effective agile prompt

🟠 Should-Have Elements (Enhanced)

Elements that significantly improve output quality and relevance

🟢 Could-Have Elements (Advanced)

Sophisticated techniques for optimal results in complex scenarios

---

MUST-HAVE ELEMENTS

1. Clarify the Agile Context

WHY:

Providing context about team composition, Sprint cycles, and known bottlenecks yields more relevant AI suggestions.

HOW:

- Describe the team setup (on-site, remote, hybrid) and any skill gaps.
- Share velocity or capacity data from past sprints.
- Highlight specific pain points (e.g., stakeholder changes at the last minute).

EXAMPLES:

- <Context>8-person distributed Scrum team (6 devs, 1 PO, 1 SM) with 2-week Sprints. Average velocity: 32 points. Challenge: Stakeholders change requirements mid-sprint. </Context>
- <Context>Our distributed Scrum team has 7 devs, 2 of them juniors, and we run a 2-week Sprint. We frequently miss acceptance criteria due to last-minute changes from stakeholders. </Context>

TRY THIS:

Identify 3-5 key contextual factors most relevant to your specific request.

---

2. Define the Core Task or Goal

WHY:

A concise, clearly stated goal keeps the AI output focused and actionable.

HOW:

- Use action verbs (e.g., “Propose,” “Design,” “Refine,” “Suggest”) and specify the outcome or scope.

EXAMPLES:

- <Goal>Design a 45-minute Retrospective agenda specifically focused on addressing our recurring problem with mid-Sprint requirement changes.</Goal>
- <Goal>Produce a 45-minute Retrospective agenda that addresses late acceptance criteria and encourages better collaboration across time zones. </Goal>

TRY THIS:

Write your goal in a single sentence, then check if it passes the "clear to a stranger" test.

---

3. Assign Roles & Perspectives

WHY:

Without guidance, AI may adopt an inconsistent or inappropriate perspective.

HOW:

- Explicitly define which agile role or expertise the AI should emulate.
- Use a short statement: “Act as a Senior Scrum Master with 10+ years of remote team experience.”

EXAMPLES:

- <Role>Act as an experienced Agile Coach who specializes in improving Product Owner and stakeholder interactions in enterprise environments. </Role>
- <Role>Act as a Product Owner dedicated to maximizing business value for an e-commerce platform. </Role>

TRY THIS:

Match the assigned role to the nature of your problem (e.g., technical debt = senior developer perspective).

---

4. Specify Output Format & Style (Include Readability Targets)

WHY:

Scrum artifacts often need to be clear and scannable. A specific format—such as bullet points, tables, or a particular reading-level target—helps achieve this.

HOW:

- Define precisely how information should be structured and presented.
- Ask for short paragraphs or bullet points.
- Aim for a target readability measure (for instance, Gunning Fog Index around 8).

EXAMPLES:

- <Format>Present your recommendations as:
    - (1) a bulleted agenda with timeboxes,
    - (2) facilitation notes under each agenda item, and
    - (3) a list of materials needed.

      Target a Gunning Fog readability index of 8-10. </Format>
- <Format>Provide your answer in bullet points under clear headings, each bullet under 40 words. Aim for a reading level accessible to a broad audience. </Format>

TRY THIS:

Consider how the output will actually be used and format accordingly.

---

5. Incorporate Real or Sample Data

WHY:

Theoretical guidance often fails to address the nuances of your actual situation. Providing partial Sprint data, velocity, or Product Backlog items ensures more tailored and pragmatic outputs.

HOW:

- Include relevant metrics, examples, or artifacts from your team.

EXAMPLES:

- <Sample Data>Here are two user stories that faced requirement changes: [examples]. </Sample Data>
- <Sample Data>Use the following data—sprint velocity of ~18 points, 3 incomplete stories—to refine our next sprint goal. </Sample Data>

TRY THIS:

Share anonymized versions of actual artifacts when possible, or synthesize realistic examples.

SHOULD-HAVE ELEMENTS

6. Impose Constraints or Special Requirements

WHY:

Without constraints, AI might suggest ideal but impractical solutions.

HOW:

- Clearly state your limitations (for example, time, resources, organizational):
  - “We only have one QA engineer.”
  - “We have capacity of 20 story points left in this Sprint.”

EXAMPLES:

- <Constraints> (1) Must work for a fully remote team, (2) some team members are in their first agile project, (3) time limited to 60 minutes total, (4) no specialized tools beyond Miro and Jira available. </Constraints>
- <Constraints> Focus on solutions that do not exceed 20 story points, since two developers are on vacation. </Constraints>

TRY THIS:

List any "hard limits" that would make a solution unusable for your team.

---

7. Ask for Iterations or Variations

WHY:

Initial solutions rarely address all aspects of complex agile challenges; having multiple solution options makes it easier to evaluate trade-offs.

HOW:

Request multiple approaches to compare or different levels of implementation.

EXAMPLES:

- <Variations>Provide three different approaches for addressing our acceptance criteria issues: (1) A minimal intervention we could implement tomorrow, (2) A moderate change for next Sprint, and (3) A comprehensive solution for long-term improvement.<Variations/>
- <Variations>Give me two Sprint planning outlines: one for fully remote teams, another for a hybrid setup.</Variations>
- <Variations>Propose two Product Backlog management strategies, each with pros and cons. </Variations>

TRY THIS:

Specify the dimensions along which you want to see variations (e.g., timeframe, complexity, focus area).

---

8. Build a Feedback Loop: Iterate & Refine

WHY:

One-shot prompts rarely produce perfect results, especially for complex agile scenarios. Agile thrives on inspect-and-adapt cycles. The same applies to prompt engineering: you can refine the AI's suggestions over multiple exchanges.

HOW:

- Treat AI interaction as an iterative process, as a conversation between knowledgeable peers—just like Agile itself:
    - Provide immediate feedback on partial outputs. For example, open all suggested links and check all listed sources.
    - Re-prompt with new or updated constraints or insights.
    - Always ask: how can we improve this prompt?

EXAMPLES:

- "I'll implement this Retrospective format and share the results with you. Then we can refine the approach for our next session based on what we learn."
- "We tried your approach, but the Retrospective took too long. Shorten it to 30 minutes for the next session." (You need to choose the thread that was previously used.)

TRY THIS:

Plan for at least one feedback cycle for essential prompts.

---

9. Prohibited Words or Phrases (Style Guidelines)

WHY:

Some expressions commonly appear in AI-generated text and can make outputs feel robotic. Restricting these words encourages more human-sounding content and aligns with brand or editorial guidelines.

HOW:

Explicitly ban overused phrases and terms that don’t match your team’s voice.

EXAMPLES:

- <Style>Please avoid these terms in your response: 'dive into', 'leverage', 'in today's fast-paced world', 'game-changer', and any variation of 'unlock potential'. Use straightforward language instead.</Style>

TRY THIS:

Start with 5-10 terms your team finds particularly unhelpful or inauthentic, and expand from there. When you use “projects,” be sure to provide your list with prohibited words as a file for the knowledge base.

🟢 COULD-HAVE ELEMENTS

10. Verify & Address Hallucinations

WHY:

AI models may invent or misinterpret details or “hallucinations.” Fact-checking avoids confusion. AI can sometimes fabricate false information, especially about agile practices.

HOW:

- Build in verification requests and fact-checking mechanisms.
- Ask for source references or disclaimers:
    - “Which Scrum Guide principle are you referencing?”
    - “Identify any acceptance criteria that you fabricated.”
- Open all suggested links and check all listed sources.
- Disallow information sources you do not want to be referenced. For example, all Scrum Guides before 2020.

EXAMPLES:

- <Hallucinations>For any claim about Scrum practices, explicitly note whether it comes from the official Scrum Guide of 2020 or is a common practice not found in the Scrum Guide 2020. If suggesting metrics, explicitly note whether you've invented them for this example. </Hallucinations>
- <Hallucinations>If you introduced any new acceptance criteria, mark them as hypothetical rather than from our real-life Product Backlog. </Hallucinations>

TRY THIS:

Ask the AI to rate its confidence in different parts of its response. When the ratings fall below a helpful threshold, use them to inspect the response and prompt it with additional rounds.

---

11. Privacy & Ethics Guidelines

WHY:

Sharing sensitive information with AI tools can create privacy or security concerns. Agile teams often handle sensitive or proprietary data, which must be protected in prompts.

HOW:

- Establish clear boundaries and anonymization practices.
- Anonymize or abstract critical info:
    - Replace real brand names or user data with placeholders.
    - Follow compliance guidelines if using public models.
    - Beware of models hosted in China, such as DeekSeek.com.

EXAMPLES:

- <Ethics_Guidelines>I've anonymized all stakeholder and product information. Please maintain this anonymization in your response and avoid making assumptions about actual people or specific product details. </Ethics_Guidelines>
- <Ethics_Guidelines>Use pseudonyms for all customer information to ensure privacy. </Ethics_Guidelines>

TRY THIS:

Create templates with placeholders for sensitive information. Upload governance rules to the project's knowledge base.

---

12. Baked-In Collaboration Flow (Treat AI as a 'Colleague')

WHY:

AI outputs often need human judgment and adaptation before implementation. Therefore, frequent feedback and ongoing discussion typically yield better outcomes than a single, unchanging prompt. Also, such "conversations" reduce the probability of hallucinations.

HOW:

- Structure prompts to encourage pauses for human consideration and iteration.

EXAMPLES:

- <Collaboration_Flow>First suggest only the Retrospective structure and await my confirmation before providing the detailed facilitation notes. This will allow me to verify the approach fits our team's needs.</Collaboration_Flow>
- <Collaboration_Flow>Break your plan into two parts. We'll implement the first half, then check if it meets stakeholder expectations before finalizing the second half.</Collaboration_Flow>

TRY THIS:

Identify natural breakpoints in complex requests where human judgment adds value.

B. Tips

1. Read-Aloud or Dictation Approach

WHY:

Written prompts can become overly formal and miss conversational clarity.

HOW:

- Read prompts aloud or use voice input to ensure they sound natural.

EXAMPLES:

- “I noticed I wasn't clear about our team structure. Let me add that we have 3 backend and 2 frontend developers, with specialized testing roles.”
- “Talking it through helped me notice we skipped testing dependencies—please add that.”

TRY THIS:

Record prompts via voice when possible, or read written prompts aloud before sending.

2: Optional Use of HTML/XML-Like Tags

WHY:

For complex prompts, using HTML/XML-style tags can improve organization. Some teams prefer these tags to structure multi-part prompts. This is **optional** but can be helpful for longer or more complex inputs.

Headings (Markdown) and bullet points can also replace these tags if your AI doesn't handle XML well. Some models may produce more structured answers when parsing tags—experiment to see what works best.

POTENTIAL BENEFITS:

- **Clarity & Readability:** Tags like <context> or <constraints> separate different prompt sections.
- **Optional Parsing:** If you have automated workflows, tags can be parsed programmatically.
- **Consistency:** A standard "prompt template" fosters uniformity across the team.

EXAMPLE of a Tag-Structured Prompt:

- **<role>**You are a Senior Scrum Master with experience facilitating remote teams during organizational change.**</role>**

  <context>Our team of 7 developers is transitioning from 2-week to 1-week Sprints due to business needs. Team members are expressing concerns about increased ceremony overhead.**</context>**

  <task>Design a streamlined Sprint Planning approach that works within a 45-minute timebox while ensuring proper story refinement and capacity planning.**</task>**

  <constraints>Must work in our remote-only environment. Two team members are in a different time zone. We can only use tools already approved: Jira and Microsoft Teams**</constraints>**

  <output_Format>Provide a detailed agenda with timeboxes, facilitation tips for each section, and specific questions to ensure efficient planning.**</output_Format>**

C. Additional Heuristics & Tips

- **Concrete Response Guidelines**
  Use a small checklist to confirm each required element is present (role assignment, data reference, constraints, banned words compliance).
- **Style Enforcement**
  Encourage consistent brand or editorial styles: short headings, bullets, or plain language. (Uploading style guides to a project's knowledge base helps.)
- **Prompt Notebook**
  Keep an organized archive of prompts and outputs for future reference.
- **Draft First, Decide Later**
  AI outputs are initial suggestions; real-world validation comes from your experience and testing.

D. Complete Examples

Example 1: Retrospective Design for Hybrid Teams

<role>
You are an experienced Agile Coach specializing in hybrid team dynamics and psychological safety.
</role>

<context>
I facilitate a Scrum team with six developers, one PO, and one SM (me). Three team members work remotely full-time, while four are in the office. We've been using this hybrid model for two months.

Our last two retrospectives revealed tension between in-office and remote members. Remote members feel left out of informal discussions, while in-office members feel remote colleagues aren't as engaged.
Our Sprint velocity is stable at around 28 points, but team satisfaction scores are dropping.
</context>

<task>
Design a 60-minute retrospective format to address the remote/in-office divide and strengthen team cohesion regardless of location.
</task>

<constraints>
- Must engage both remote and in-office participants equally
- Should address underlying tensions without creating blame
- Time limit: 60 minutes total
- Available tools: Miro, Zoom, Slack
- One remote team member has an unreliable internet connection
</constraints>

<outputFormat>
Provide:
1. A detailed timeline with specific activities and timeboxes
2. Required preparation steps
3. Facilitation notes for each activity
4. Guidance on managing potential difficult moments
5. Techniques to ensure balanced participation
</outputFormat>

<prohibitedWords>
Unlock potential, leverage, dive deep, journey, game-changer, at the end of the day,
moving forward, synergy, bandwidth, touch base.
</prohibitedWords>

Example 2: Sprint Planning Optimization

<role>
You are a Product Owner with Product Backlog refinement and Sprint Planning expertise.
</role>

<context>
My Scrum team consists of five developers with varying levels of seniority. Our Sprint Planning sessions consistently exceed the allotted 2-hour timebox, frustrating everyone. The main issues appear to be:
1. Product Backlog items aren't sufficiently refined beforehand
2. Too much technical discussion happens during Sprint Planning instead of before, during refinement
3. Estimation takes too long with frequent debates, see #2.

Our last three Sprints had velocities of 12, 14, and 11 Product Backlog items. We're using 2-week Sprints.

Here's an example of a typical Product Backlog item from our Product Backlog:

"As a premium user, I want to download my transaction history to keep records for my accounting."
</context>

<task>
Redesign our Sprint Planning process to fit within a 90-minute timebox while ensuring the creation of proper Sprint Goals based on the Product Owner's business objective for the upcoming Sprint.
</task>

<constraints>
- Need to maintain thorough Sprint Planning despite shorter timeframe based on the Product Owner's objective
- Must work for two junior developers who need more support to compensate for their lack of experience
- We're limited to using Jira and can't change tools
- Can modify our Product Backlog refinement process if needed.
</constraints>

<outputFormat>
Provide:
1. A revised Sprint Planning agenda including time estimates
2. Recommended pre-planning activities
3. Facilitation techniques for keeping discussions on track
4. A template for capturing essential information during Sprint Planning
5. Suggestions for post-planning activities.
</outputFormat>

E. Prompt Design Worksheet

Use this worksheet to build your prompt following the framework:

1. **What is the problem we are trying to solve?**
    - *[Your answer here]*
2. **What specific context does the AI need to understand?**
    - Team structure: *[Your answer]*
    - Sprint/workflow details: *[Your answer]*
    - Current challenges: *[Your answer]*
    - Relevant metrics: *[Your answer]*
3. **What role should the AI adopt?**
    - *[Your answer here]*
4. **What specific output do I need?**
    - *[Your answer here]*
5. **What format will be most helpful?**
    - *[Your answer here]*
6. **What constraints must the solution respect?**
    - *[Your answer here]*
7. **Would multiple approaches or variations help?**
    - *[Your answer here]*
8. **What real examples or data can I provide?**
    - *[Your answer here]*
9. **How will I verify and implement the AI's suggestions?**
    - *[Your answer here]*
10. **Are there any style preferences or prohibited terms?**
    - *[Your answer here]*

F. Measuring Success

Evaluate your prompt engineering effectiveness using these metrics:

1. Relevance Score (1-5)

How well did the AI output address your specific context and needs?

2. Actionability (1-5)

Could you implement the suggestions without significant additional work?

3. Iteration Count

How many prompt refinements were needed to get a practical result?

4. Time Savings

Estimate time saved compared to traditional approaches.

5. Implementation Success

Did the implementation of AI suggestions achieve the desired outcome?

NOTE:

Track these metrics to refine your prompt engineering skills by creating a prompt repository. Use beneficial prompts as examples for the LLM and provide those as a document for the project's knowledge base.

G. Troubleshooting Guide

Problem: AI gives generic, theoretical advice

Fix: Add more specific context and real examples from your team. Include metrics and actual artifacts. Inspect and adapt, and create multiple prompt iterations.

Problem: AI suggests complex solutions that won't work in your environment

Fix: Explicitly state constraints and limitations. Be specific about available tools, time restrictions, and team capabilities. Define a solution room that works for you, not for the model.

Problem: AI makes factually incorrect statements about agile practices

Fix: Ask it to cite specific sources (e.g., Scrum Guide) or explicitly note when it's suggesting something not from official methodologies. Exclude documents in your prompt you consider harmful; for example, all Scrum Guides from earlier than 2020.

Problem: AI suggestions feel robotic or inauthentic

Fix: Use the prohibited words list and specify the tone you want. Consider using the read-aloud technique to craft more natural prompts; the model will answer in kind.

Problem: AI output is too long or detailed for practical use

Fix: Specify output constraints like "Limit to 5 bullet points" or "Keep the entire response under 500 words." Alternatively, try shortening the model's reply with a follow-up prompt: "Shorten by half."

H. Model-Specific Considerations

Different AI models may require adjustments to your prompt approach:

ChatGPT / GPT-4

- Excels with detailed instructions and context
- Benefits from clear formatting guidance
- May need explicit requests to avoid generating unnecessary theory.

Claude

- Handles nuanced ethical considerations well
- Works effectively with structured tag formats
- More explicit instructions may be needed to generate practical examples.

Perplexity / Bard

- Strong with factual recall about agile methodologies
- May require more guidance on output structure
- Benefits from specific questions rather than open-ended requests.

Experiment with different approaches for each model to discover what works best for your particular needs.

I. Learning Path

Beginner

1. Start with Must-Have elements only
2. Focus on context clarity and specific outputs
3. Practice with simple requests (e.g., meeting agendas, basic Retrospective formats).

Intermediate

4. Add Should-Have elements to your prompts
5. Experiment with variations and constraints
6. Try more complex scenarios (e.g., conflict resolution, cross-team coordination).

Advanced

7. Incorporate all Could-Have elements
8. Develop custom structured templates for recurring needs
9. Create feedback loops between AI outputs and team implementation
10. Add a repository of beneficial prompts to the model's knowledge base.

Remember: AI assistance is most effective when combined with your expertise and judgment. These tools support—not replace—the human elements that make Agile successful.

J. Legal

Berlin Product People GmbH
Stöckter Deich 101
D-21423 Winsen (Luhe)
Germany

Registered at: AG Lüneburg (HRB 212146)
Managing Director: Stefan Wolpers
VAT identification number according to §27a Umsatzsteuergesetz: DE227895748