Intelligent Human Systems Integration (IHSI), Vol. 200, 2026, 1–11
https://doi.org/10.54941/ahfe1007059

Artificial Intelligence Maturity Model (AIMM)

James Helfrich[1] and Chris Mijangos Xicara[2]

[1]Brigham Young University – Idaho, Rexburg, ID 83440, USA
[2]Brigham Young University, Provo, UT 84602, USA

ABSTRACT

Maturity models have long served as effective frameworks for guiding organizations toward progressively advanced practices, offering structured pathways for capability development and benchmarking. Models such as the Capability Maturity Model Integration (CMMI), Organizational Project Management Maturity Model (OPM3), Portfolio, Program and Project Management Maturity Model (P2CMM), and the Data Management Maturity Model (DMM) have each demonstrated the value of systematic assessment in driving adoption, standardization, and continuous improvement across diverse fields. Despite the rapid rise of artificial intelligence (AI) adoption, no broadly accepted maturity model exists to help organizations evaluate and advance their AI capabilities. This paper introduces the Artificial Intelligence Maturity Model (AIMM), a structured framework designed to assess individual and organizational sophistication in AI agent utilization. The proposed model defines distinct maturity levels, enabling organizations to identify their current state, benchmark progress, and establish a roadmap for advancement. By providing a standardized approach to AI capability assessment, AIMM can accelerate more effective use of AI technologies across industries.

Keywords: AI, Agent, Maturity model, LLM

INTRODUCTION

Maturity Models such as CMMI (software and process improvement), OPM3 (organizational project management), DMM (data management), PCMM (people capability management), AMM (accessibility) and others have proven to be useful tools to both help practitioners access the sophistication of their practice, as well as provide benchmarks they can aspire to. Though there is a high demand for integrating Artificial Intelligence (AI) tools in a wide variety of intellectual pursuits, there is as of yet no maturity model characterizing their use. Through a literature review of existing maturity models and their uses, in addition to a survey of AI uses in a variety of contexts, a maturity model has been described allowing practitioners and managers to characterize the sophistication of their utilization of AI technologies.

There is a huge demand for more effective use of AI Agents. A large number of technologies, best practices, and techniques has been developed. However, the AI Agent sophistication has remained low. There is an urgent demand for enhanced capability and use of AI agents.

Received September 13, 2025; Revised November 10, 2025; Accepted November 25, 2025; Available online February 1, 2026

This paper describes how various maturity models have guided the advancement of a variety of disciplines in recent years. It then surveys existing AI agent users to qualify the components of more sophisticated use. Finally, it proposes a new maturity model designed to characterize the sophistication of AI utilization.

- RQ1: How can the sophistication of an individual's use of AI agents be best characterized?

LITERATURE REVIEW

There are two components to the literature review: Maturity Models and AI Prompt Techniques.

Maturity Models

The term "maturity model" was coined by Watts Humphrey in 1988 and is defined as the progressive formalization and optimization of organizational processes, moving from ad hoc practices to defined steps, measurable metrics, and continual improvement (Humphrey, 1988).

The first maturity model was sponsored by the U.S. Department of Defense in 1987, published by Humphrey at Carnegie Mellon's Software Engineering Institute (Humphrey, 1987) and formally established in 1993 as the definitive Capacity Maturity Model (CMM) Version 1.1 (Paulk, 1993). The fundamental characteristic of the CMM is a five-level framework: Initial, Repeatable, Defined, Managed, and Optimizing (Humphrey, 1988). It was widely adopted by software development organizations worldwide (Paulk et al., 1991) and then extended to become Capacity Maturity Model Integrated (CMMI) in 2002 (CMMI Product Team, 2002). CMMI benefits software development teams by providing a framework for continuous improvement and organizational capability assessments (Paulk, 1993).

Leveraging the success of CMMI, many organizations began using maturity models for management and improvement. Organizations recognized the critical need to create strategic business objectives and enhance their competitive advantage, therefore new models such as OPM3 (PMI, 2003) and P2CMM (Mansur, 2013) emerged as essential frameworks for organizational development. OPM3 was first published by the Project Management Institute (PMI) in 2003 (PMI, 2003) to help organizations assess and improve project, program, and portfolio practices (Miller, 2004). OPM3 consists of three domains (project, program, and portfolio management) and operates through four progressive stages: standardize, measure, control, and continuous improvement (Miller, 2004). Studies show that the OPM3 model provided a positive contribution to project performance (Mansur, 2013). P2CMM was first described by Zhang Lianying (Lianying et al., 2012) integrating PRINCE2 methodology with the project management approach that contains eight unique management processes: Starting Up a Project (SU), Initiating a Project (IP), Directing a Project (DP), Controlling a Stage (CS), Managing Product Delivery (MP), Managing Stage Boundaries (SB), Closing a Project (CP), and Planning (PL) (Lianying et al., 2012). Another

model dedicated to software engineering management organizations was the People Capability Maturity Model (P-CMM) (Curtis et al., 1995). P-CMM was developed in 1995 by SEI at Carnegie Mellon University, building on Humphrey's process maturity framework to extend software process improvement principles to workforce management (Curtis et al., 1995). The P-CMM framework contains five maturity levels with 22 process areas from ad hoc practices for continuous improvement (Curtis et al., 1995).

Models such as Control Objectives for Information and Related Technology (COBIT) and the Cybersecurity Maturity Model Certification (CMMC) represent an evolution from audit-focused IT governance to the use of frameworks for different sectors in IT. COBIT was introduced in 1996 by ISACA as the first standardized framework specifically designed to help financial auditors navigate IT related environments (ISACA, 1996). It introduced maturity assessments for governance and management processes (Gail Ridley, 2004) and provides control objectives to ensure alignment between IT use and business goals (George Mangalaraj, 2014). CMMC was introduced in 2020 by the U.S. Department of Defense (DoD, 2020). It ensures that defense contractors adequately protect sensitive unclassified information–specifically Federal Contract Information and Controlled Unclassified Information–from cyber threats (Peters, 2020). The current CMMC 2.0 model features three maturity levels: Foundational (Level 1), Advanced (Level 2), and Expert (Level 3). (Department of Defense, 2024).

Other models process dimensions and capabilities to leverage data-driven organizations such as the Data Management Maturity Model (DMMM) (Pörtner et al., 2025). The DMMM "is designed around 4 building blocks: 4 data categories, 14 data capabilities, 5 maturity levels, and a scoring system" (Zitoun et al., 2021) and was officially presented by CMMI Institute in 2014 (Weisman, 2014). The Gartner Data and Analytics Maturity Score for CDAOs (Chief Data and Analytics Officers) is not a single, specific invention with a set date (Gartner Inc., 2024). It is designed with 5 levels: Basic, Opportunistic, Systematic, Differentiating and Transformational (Gartner Inc. , 2018). The W3C Accessibility Maturity Model is "a guide for organizations to evaluate and improve their business processes to produce digital products that are accessible to people with disabilities" (W3C, 2022). It was first published in 2022 and continues to be updated (W3C, 2025).

A common characteristic of all these maturity models is: "maturity models offer organizations a simple but effective possibility to measure the quality of their processes" (Wendler, 2012), and that they are "widely used technique that is proved to be valuable to assess business processes or certain aspects of organizations, as it represents a path towards an increasingly organized and systematic way of doing business" (Proença & Borbinha, 2016). They also have a prerequisite structure where qualification for Level n can only occur after Level n–1 has first been obtained.

AI Prompt Techniques

Gwern Branwen first coined the term *prompt engineering* in his 2020 essay *Prompts and Programming* (Branwen, 2020), although the term did not appear in an academic paper until 2021 (Liu & Chilton, 2021).

Prompt engineering is best described as "strategically designing task-specific instructions, referred to as prompts, to guide model output without altering parameters" (Sahoo et al., 2024). Since the widespread adoption of large language models (LLMs), several prompt engineering techniques have been developed. The first is Chain-of-Thought (CoT) prompting, in which the LLM is instructed to "generate a series of short sentences that mimic the reasoning process a person might employ in solving a task" (Wang et al., 2022). Other terms representing the same concept include prompt design (Amatriain, 2024), prompt programming (Liang et al., 2025), prompt tuning (Lester et al., 2021), and instruction tuning (Wei et al., 2025). Related methods include Logical Chain-of-Thought (LogiCoT) prompting (Xhao et al., 2023) and Chain of Symbol (CoS) prompting (Hu et al., 2024). Zhang et al. (Zhang et al., 2022) streamlined CoT through Automatic Chain-of-Thought (Auto-CoT), which leverages question clustering and demonstration sampling. Another approach is Least-to-Most (LtM) prompting, which "break[s] down a complex problem into a series of simpler subproblems and then solve[s] them in sequence" (Zhou et al., 2022). A further refinement is Self-Consistency prompting, where LLMs generate multiple "diverse set of reasoning paths instead of only taking the greedy one, and then selects the most consistent answer" (Wang et al., n.d.). A closely related technique is Tree-of-Thoughts (ToT) prompting (Yao et al., 2023). Each of these techniques can be described as zero-shot prompting, where the model is expected to generate a correct answer without examples or demonstrations (Brown, 2020). The common characteristics include familiarity with LLM technologies (Zhou et al., 2022), providing adequate context (Sahoo et al., 2024), specificity (Sahoo et al., 2024), and hints (Xhao et al., 2023).

A second class of prompting is called "iterative prompting," the term coined in 2022 (Wang et al., 2022), and defined as "a technique in prompt engineering that involves maintaining context over multiple interactions between the user and the AI model ." The same concept is called multi-turn prompting (Zheng et al., 2024), prompt refinement (Pandita et al., 2025), human-in-the-loop (Shah, 2025), supervised prompting (Lin et al., 2024), rhetorical prompt engineering (Nupoor et al., 2025). Perhaps Zheng et al. (Zheng et al., 2024) defined this technique best in the context of code generation:

> In multi-turn code generation, we can distinguish between a *Natural-Language-to-Code* (NL → Code) task in the first turn and *Code-to-Code* (Code → Code) generation in subsequent turns. For a given problem, we generate a sequence of intermediary code samples $c_1,\ldots,c_T$ rather than just one. After each turn i, the code sample c_i is fed back into the model π together with an *execution feedback* prompt to obtain the next sample $c_i{+}1$. This process is repeated T times until we either pass all public tests or until a maximum number of turns N is reached (Zheng et al., 2024, p. 3).

There are several variations of iterative prompting. One is Prompt Optimization with Human Feedback (POHF), a technique where the LLM produces a collection of eligible responses presented to the human user for preference feedback (Lim et al., 2024). Prompt Optimization in Multi-Step

Tasks (PROMST) offers a similar strategy, in which human agents provide feedback in multi-step tasks to "offset direct suggestions for improvement" (Chen et al., 2024, p. 3859). Finally, the iPrOp model proposes a four step workflow: the human produces the initial prompt, the LLM rephrases it to create an update, the human evaluates the change (potentially with a revised prompt), and finally the prompt is executed (Li & Klinger, 2025). The common thread to each of these models is the human-in-the-loop where humans are active contributors in the results-generating process (Shah, 2025, p. 57), including such activities as road mapping (Lim et al., 2024), delegation (Chen et al., 2024), evaluation (Li & Klinger, 2025), redirection (Li & Klinger, 2025), and assembly of the final solution.

The third and final class of prompting is through the use of AI agents first described by Castelfranchi in 1998 (Castelfranchi, 1998). It is perhaps best defined as a system that perceives its environment and acts upon it in a way that maximizes its expected performance given its percept history and knowledge (Russell & Norvig, 2022). The process of building such systems called *agent engineering* (Liu, 2001), defined as "the development of autonomous computational or physical entities capable of perceiving, reasoning, adapting, learning, cooperating and delegating in a dynamic environment." Agent engineering has also been labelled as agent-oriented software engineering (Wooldridgey & Ciancarini, 2000), agent-based software engineering (Jennings, n.d.), agentic systems (Anthropic, 2024), and In-Context Learning (ICL) (Radford et al., 2019). These systems are characterized with autonomy, reactivity, and pro-activeness (Wooldridgey & Ciancarini, 2000). Several AI agents have been built and studied to perform specific tasks, including SWE-Agent, an AI agent to assist with some software engineering tasks (Yang et al., 2024), WebShop, and agent helping users navigate an e-commerce site (Yao et al., 2022), MineDojo, an agent capable to perform Mindcraft tasks (Fan et al., 2022), and others. Agents are enabled to perform these special tasks through imitation learning (Yao et al., 2022), reinforcement learning (Fan et al., 2022), task-specific fine-tuning (Chebotar, 2023), and system-level scaffolding (Yang et al., 2024). The common thread with all these tools is perhaps best summarized by Shah: "We also need this [agent] to be systematically created with enough generalizability and explainability to be useful for the same or similar experiments by other researchers, by other LLMs, and at different times" (Shah, 2025, p. 57).

EXPERIMENTAL FRAMEWORK

The experimental framework comprised four steps: participant selection, interviews, coding, and grouping.

Participant Selection

A group of software engineer participants was recruited through LinkedIn and similar platforms to achieve a diverse set of experiences, attitudes, and backgrounds. A prototype survey was created, but inconsistent understanding of AI prompting terminology produced unsatisfactory outcomes. Thus, a qualitative data collection strategy was employed.

Interviews

Each participant was individually interviewed and asked to describe their attitude toward and use of AI in professional and personal contexts. The interviewer periodically prompted participants to elaborate on previously made statements, but care was taken to avoid leading the participants down any particular line of inquiry. Each interview was transcribed, personally identifiable information (PII) was removed, and a random 6-digit number was assigned to preserve anonymity.

Coding

Each interview was meticulously coded (Berelson, 1952) according to the activities identified in the literature review as presented in Table 1.

Table 1: Codes and their corresponding definitions.

Code	Definition
ASK	The LLM is instructed to ask the user questions to gather data for a prompt response.
ASSEMBLY	The participant mentioned assembling a response from multiple sub-responses generated by an LLM.
CONTEXT	The participant mentioned systematically providing context necessary for the LLM to respond to the prompt.
EVALUATION	The participant mentioned evaluating the results of one prompt with the intention of creating a follow-up prompt.
EXAMPLES	The participant mentioned providing the LLM with multiple problem/solution pairs. These are often called exemplars or shots.
HINT	The participant mentioned giving the LLM hints of possible solutions or ways a problem might be approached.
INSTRUCTION	The participant mentioned creating an instruction file describing how to solve a problem before executing a prompt.
MULTIPLE	The participant mentioned at least one instance of solving a problem that involved more than one planned interaction in a chat.
REDIRECTION	The participant mentioned giving the LLM a clarification, redirection, or refinement based on previous results.
ROLE	The participant mentioned describing the role the LLM was to play in fulfilling a given prompt.
SAVE	The participant mentioned saving a prompt for reuse.
SIMPLE	The participant mentioned at least one use of AI.
SPECIFICITY	The participant mentioned being careful to avoid ambiguity in a prompt.
STEPS	The participant mentioned giving the LLM explicit steps to follow to solve a problem.

GROUPING

Once each interview was tagged, an additional field for each participant, N_TAGS, was calculated by summing the number of tags assigned to that participant. Similarly, a field, N_PARTICIPANTS, was calculated for each tag by summing the number of participants associated with that tag. The tags were then sorted by the N_PARTICIPANTS field, and the participants were sorted by the N_TAGS field. From the resulting table, groupings of similar responses were generated.

RESULTS

A total of 27 interviews were conducted, with a combined duration of 496 minutes and an average length of 18.4 minutes. The use of AI prompting techniques by participants is presented in Table 2.

Table 2: Prompting techniques employed by participants column-sorted by **N_TAGS** and row-sorted by **N_PARTICIPANTS.**

	126543	453725	756245	981272	831165	987252	346532	714782	372635	459214	537822	723423	654825	439912	723573	221887	559213	312996	881426	239187	812990	191237	349560	539001	627635	656765	972638
SIMPLE	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
SPECIFICITY							■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
CONTEXT								■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■
HINT													■	■	■	■		■	■	■	■	■	■	■	■	■	■
REDIRECTION																	■	■	■	■	■	■	■	■	■	■	■
EXAMPLES																	■	■	■	■	■	■	■	■	■	■	■
STEPS																	■		■	■	■	■	■	■	■	■	■
ROLE																	■		■	■	■	■	■	■	■	■	■
ASSEmBLY																		■	■	■	■	■	■	■	■	■	■
MULTIPLE																■		■			■	■	■	■	■	■	■
ASK																■		■		■		■	■	■	■	■	■
SAVE																	■	■	■			■	■	■	■	■	■
INSTRUCTION																	■			■	■	■	■	■	■	■	■
EVALUATION																■						■	■	■	■	■	■

From the data presented in Table 2, four distinct groupings emerged. Three of these groupings correspond to the three classes of AI prompt techniques identified in the literature review: prompt engineering, agent management, and agent engineering. These groups are presented in Table 3.

Table 3: Groupings of AI prompting techniques.

#	Codes	Participants
1	SIMPLE	126543, 453725, 756245, 981272, 831165, 987252
2	SIMPLE, SPECIFICITY, CONTEXT, HINT	346532, 714782, 372635, 459214, 537822, 723423, 654825, 439912, 723573
3	SIMPLE, SPECIFICITY, CONTEXT, HINT, REDIRECTION, ASSEMBLY, EVALUATION, MULTIPLE	221887, 559213, 312996, 881426, 239187, 812990
4	SIMPLE, SPECIFICITY, CONTEXT, HINT, REDIRECTION, ASSEMBLY, EVALUATION, MULTIPLE, ASK, EXAMPLES, STEPS, ROLE, SAVE, INSTRUCTIONS	191237, 349560, 539001, 627635, 656753, 972638

CONCLUSION

Six interviews revealed no strategies characteristic of prompt engineering. These were given the level name "Novice," indicating a lack of sophistication in AI utilization. Nine interviews revealed some or all of the characteristics of prompt engineering identified in the literature, including specificity in questioning and the provision of adequate context. Additionally, many participants provided hints or solution fragments to help the AI agents. This level was given the name "Prompt Engineer." Six interviews revealed all the characteristics of "Prompt Engineer," along with evidence of interactions with the AI agent or the use of multiple steps in the process. This level was given the name "Agent Manager." Note that there is far less uniformity in this level than in the others; use of various agent management techniques is used sporadically. Finally, six interviews revealed all the characteristics of "Agent Manager," as well as evidence of specifically training agents to perform tasks they had previously been incapable of performing. This level was given the name "Agent Engineer."

Based on previously cited work and the data collected for this study, we propose a four-level maturity model describing the sophistication of individual AI agents use.

1. **Novice**: Individuals demonstrating little knowledge of how LLMs function and fail to provide the necessary information for a prompt to provide satisfactory results.
2. **Prompt Engineer**: Individuals who leverage LLM knowledge to solve problems using one or more techniques, including SPECIFICITY, CONTEXT, and HINTS.
3. **Agent Manager**: Prompt Engineers who manage the solution-generation process through multiple interactions arranged in a planned sequence of steps. They employ one or more of the following techniques: MULTIPLE, EVALUATION, REDIRECTION, and ASSEMBLY.
4. **Agent Engineer**: Agent Managers who train LLMs to perform new tasks. They apply one or more of the following techniques: INSTRUCTIONS, SAVE, STEPS, ROLE, and EXAMPLES.

Several potential avenues for future research include correlating the sophistication of AI utilization with the quality of results, developing novel agent management techniques, designing strategies for optimizing agent training, and exploring teaching methods to help human AI users advance their sophistication in AI usage. Finally, a correlation was noted between the sophistication in AI usage with the complexity of the task attempted. A dedicated study could determine how these variables interact.

REFERENCES

Amatriain, X., 2024. *Prompt Design and Engineering: Introduction and Advanced Methods*. [Online]. Available at: https://arxiv.org/html/2401.14423v3?utm_source=chatgpt.com

Anthropic, 2024. *Building Effective Agents*. [Online]. Available at: https://www.anthropic.com/engineering/building-effective-agents

Berelson, B., 1952. *Content analysis in communication research*. s.l.:s.n.

Branwen, G., 2020. *GPT-3 Creative Fiction*. [Online]. Available at: https://gwern.net/gpt-3#sn5

Brown, T. e. a., 2020. Language Models are Few-Shot Learners. *Advances in neural information processing systems 33*, pp. 1877–1901.

Castelfranchi, C., 1998. Modelling Social Action for AI Agents. *Artificial Intelligence,* 103(1–2), pp. 157–182.

Chebotar, Y. e. a., 2023. *RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control*. s.l., s.n., pp. 2165–2183.

Chen, Y. et al., 2024. *PRompt Optimization in Multi-Step Tasks (PROMST): Integrating Human Feedback and Heuristic-based Sampling*. Miami, Association for Computational Linguistics, p. 3859–3920.

CMMI Product Team, 2002. *Capability Maturity Model Integration (CMMI), Version 1.1*. Pittsburgh: Carnegie Mellon University.

Curtis, B., Hefley, W. & Miller, S., 1995. *Overview of the People Capability Maturity Model*. s.l.: Software Engineering Institute.

Department of Defense, 2024. *Cybersecurity Maturity Model*. s.l.: Department of Defense.

DoD, 2020. *About CMMC*. [Online]. Available at: https://dodcio.defense.gov/cmmc/About/

Fan, L. et al., 2022. *Minedojo: Building open-ended embodied agents with internet-scale knowledge*. s.l., s.n., pp. 18343–18362.

Gail Ridley, J. Y. P. C., 2004. *COBIT and its Utilization: A framework from the literature*. Hawaii, IEEE, pp. 1–8.

Gartner Inc., 2018. *Gartner*. [Online]. Available at: https://www.gartner.com/en/newsroom/press-releases/2018-02-05-gartner-survey-shows-organizations-are-slow-to-advance-in-data-and-analytics.

Gartner Inc., 2024. *Gartner Identifies the Top Trends in Data and Analytics for 2024*. [Online]. Available at: https://www.gartner.com/en/newsroom/press-releases/2024-04-25-gartner-identifies-the-top-trends-in-data-and-analytics-for-2024.

George Mangalaraj, A. T. A. S., 2014. *IT Governance Frameworks and COBIT - A Literature Review*. Savannah, s.n., pp. 1–10.

Hu, H. et al., 2024. Chain-of-symbol prompting elicits planning in large langauge models. *arXiv*.

Humphrey, W. S., 1987. *Characterizing the Software Process: A Maturity Framework (Technical Report CMU/SEI-87-TR-11),* Pittsburgh: Software Engineering Institute, Carnegie Mellon University.

Humphrey, W. S., 1988. *Characterizing the Software Process: A Maturity Framework*. Pittsburgh: IEEE.

ISACA, 1996. *COBIT - An ISACA Framework*. [Online]. Available at: https://www.isaca.org/resources/cobit

Jennings, N. R., n.d. On agent-based software engineering. *Artificial intelligence,* 117(2), pp. 277–296.

Lester, B., Al-Rfou, R. & Constant, N., 2021. *The Power of Scale for Parameter-Efficient Prompt Tuning*. s.l., s.n., pp. 3045–3059.

Li, J. & Klinger, R., 2025. *iPrOp: Interactive Prompt Optimization for Large Language Models with a Human in the Loop*. s.l., Association for Computational Linguistics, pp. 276–285.

Liang, J., Lin, M., Rao, N. & Myers, B., 2025. Prompts are Programs Too! *Understanding How Developers Build Software Containing Prompts*. Proceedings of the ACM on Software Engineerin, Volume 2, pp. 1591–1614.

Lianying, Z., Jing, H. & Xinxing, Z., 2012. *The Project Management Maturity Model and Application*. Tianjin, Elsevier, p. 3691–3697.

Lianying, Z., *Jing, H. & Xinxing, Z., 2012*. The Project Management Maturity Model and Application Based on PRINCE2. *Procedia Engineering*, Volume 29, pp. 3691–3697.

Lim, X. et al., 2024. *Prompt Optimization with Human Feedback*. Vienna, s.n.

Lin, Y.-C.et al., 2024. *Interpretable User Satisfaction Estimation for Conversational Systems with Large Language Models*. s.l., s.n., pp. 11100–11115.

Liu, J., 2001. *Agent Engineering*. s.l.: World Scientific.

Liu, V. & Chilton, L., 2021. *Design Guidelines for Prompt Engineering Text-to-Image Generative Models*. New Orleans, ACM, pp. 1–23.

Mansur, F. J. a. A. K., 2013. Project Management Maturity Models and Organizational Project Managment Maturity Model (OPM3): A Critical Morphological Evaluation. p. 4.

Miller, B., 2004. *The Pathway to OPM3*. Anaheim, Project Management Institute.

Miller, B., 2004. *The pathway to OPM3: A busy project manager's guide to advancing organizational maturity.* Anaheim, CA, Project Management Institute, p. 1.

Nupoor, R., Saravia, M. & Johri, A., 2025. Using Rhetorical Strategies to Design Prompts: A Human-in-the-loop Approach to Making AI Useful. *AI & Society,* pp. 711–732.

Pandita, D. et al., 2025. ProRefine: Inference-time Prompt Refinement with Textual Feedback. *arXiv.*

Paulk, M. C., Curtis, B. & Chrissis, M. B., 1991. *Capability Maturity Model for Software,* Pittsburg: Department of Defense.

Paulk, M. C. C. W. C. M. B. &. W. C. V., 1993. *Capability Maturity Model for Software (Version 1.1) (Technical Report CMU/SEI-93-TR-024),* Pittsburgh: Software Engineering Institute.

Peters, H. M., 2020. *Defense Acquisitions: DOD's Cybersecurity,* s.l.: Congressional Research Service.

PMI, 2003. *Organizational Project Management Maturity Model Knowledge Foundation*. Newton Square: Project Management Institute.

Pörtner, L. et al., 2025. Data Management Maturity Model—Process Dimensions *and Capabilities to Leverage Data-Driven Organizations Towards* Industry 5.0. 21 March, pp. 1–3.

Proença, D. & Borbinha, J., 2016. Maturity Models for Information Systems - A State of the Art. Lisboa, Elsevier, pp. 1042–1049.

Russell, S. & Norvig, P., 2022. *Artificial Intelligence: A Modern Approach*. s.l.: Prentice Hall.

Sahoo, P., Singh, A. K., Saha, S. & Jain, V., 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv preprint arXiv:2402.07927*.

Shah, C., 2025. From Prompt Engineering to Prompt Science with Humans in the Loop. *Communications of the ACM,* 68(6), pp. 54–61.

W3C, 2022. *W3C Accessibility Maturity Model*. [Online]. Available at: https://www.w3.org/TR/2022/DNOTE-maturity-model-20220906/

W3C, 2025. *Accessibility Maturity Model*. [Online]. Available at: https://www.w3.org/TR/maturity-model/#description-maturity-stages

Wang, B., Deng, X. & Sun, H., 2022. *Iteratively Prompt Pre-trained Language Models for Chain of Thought*. s.l., Association for Computational Linguistics, pp. 2714–2730.

Wang, X. et al., 2022. Wang, Xuezhi, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. "Self-consistency improves chain of thought reasoning in language models. *arXiv,* p. arXiv preprint arXiv:2203.11171.

Wang, X. et al., n.d. *Self-Consistency Improves Chain of Thought Reasoning in Language Models*. s.l., s.n.

Wei, J. et al., 2025. *Finetuned Language Models Are Zero-Shot Learners*. s.l., s.n.

Weisman, R. M., 2014. *Data Management Maturity Model Introduction*. Ottawa, CMMI, pp. 1–20.

Wendler, R., 2012. The maturity of maturity model research: A systematic mapping study. In: *Information and Software Technology*. s.l.: Elsevier, pp. 1317–1339.

Wooldridgey, M. & Ciancarini, P., 2000. *Agent-oriented software engineering: The state of the art*. Berlin, Springer Berlin Heidelberg, pp. 1–28.

Xhao, X. et al., 2023. Enhancing Zero-Shot Chain-of-Thought Reasoning in Large Language Models through Logic. *arXiv*.

Yang, J. et al., 2024. *SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering*. s.l., s.n., pp. 50528–50652.

Yao, S., Chen, H., Yang, J. & Narasimhan, K., 2022. *WebShop: Towardds Scalable Real-World Web Interaction with Grounded Language Agents*. s.l., s.n., pp. 20744–20757.

Yao, S. et al., 2023. Tree of thoughts: Deliberate problem solving with large language models.

Zhang, Z., Zhang, A., Li, M. & Smola, A., 2022. Automatic chain of thought prompting in large language models. *arXiv*.

Zheng, K. et al., 2024. *What Makes Large Language Models Reason in (Multi-Turn) Code Generation?*. Singapore, s.n.

Zhou, D. et al., 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv*.

Zitoun, C., Belghith, O., Ferjaoui, S. & Gabouje, S. S. D., 2021. *DMMM: Data Management Maturity Model*. St. Petersburg, IEEE, p. 2.