# **Technical Research Report: An Auditing Project Manager Agent for Verifiable Task Completion**

Version: 27.0

Date: 18 June 2025

Status: Approved

## **1. Abstract**

This report details the foundational design for the fourth and final agent in the autonomous R&D pipeline: the Master Project Manager (PM) agent. The successful implementation of the modular testing framework (TRR v26.0, TDS v9.0) has created a stable environment in which the PM agent's functionality can be reliably developed and verified. The PM's initial role will not be that of a high-level strategist, but of a meticulous, automated Auditor. Its primary function is to programmatically verify that the output of an agent (e.g., the Coding Agent) precisely matches the requirements of its input task (e.g., the Design Agent's work order). By starting with a simple, verifiable audit of the "Hello, World!" unit test, we can build a PM agent on a foundation of certainty, paving the way for its future evolution into a truly autonomous orchestrator.

## **2. The Project Manager as an Auditor**

The core challenge for an autonomous PM is to reliably answer the question: "Did the last task succeed?" In a complex generative system, this is a non-trivial problem. Our modular testing framework simplifies this dramatically. The PM's role is therefore defined as follows:

* Input (Goal): The PM receives the artifact that defined the task. For the Coding Agent, this is the design\_synthesis.json "work order."
* Input (Evidence): The PM receives the artifact that represents the completed work. For the Coding Agent, this is the coding\_execution\_report.json.
* Process (Audit): The PM executes a "diff" or comparison between the Goal and the Evidence. It follows a strict set of rules to check for discrepancies.
* Output (Verdict): The PM produces a structured pm\_audit\_report.json containing a clear verdict: SUCCESS or FAILURE, along with its reasoning.

This process transforms the PM's function from subjective interpretation into a deterministic, verifiable audit.

## **3. The "Hello, World!" Audit: A Unit Test for the PM**

The "Hello, World!" test case provides the perfect first unit test for the PM agent itself. We can now test the auditor.

* Goal: We provide the PM agent the test\_design\_synthesis.json file. This file clearly states the goal is to create one new file, hello.py, with the content print("Hello, World!").
* Evidence: We provide the PM agent the coding\_execution\_report\_...json artifact that was successfully generated by the Coding Agent during its own unit test.
* Expected PM Behavior: The PM agent should perform the following logical steps:
  1. Parse the GOAL artifact and identify that the new\_files array contains one object for hello.py.
  2. Parse the EVIDENCE artifact and identify that the final\_code\_artifacts array contains one object for hello.py.
  3. Compare the content field for hello.py from both artifacts.
  4. Since the goal and the evidence match perfectly, the agent must conclude the task was a SUCCESS.
* Verification: We, the human supervisors, can verify that the PM agent correctly produced a pm\_audit\_report.json with a SUCCESS status.

## **4. The Path to Autonomy**

This simple auditing capability is the cornerstone of future autonomy. Once the PM can reliably verify task completion, its responsibilities can be expanded. A future version, for example, could be given the authority to:

* If SUCCESS, automatically trigger the next step in a sequence (e.g., run a testing agent, or merge a code branch).
* If FAILURE, automatically trigger a feedback loop, sending the audit report back to the failed agent for a retry.
* Ultimately, use a sequence of successful audit reports to determine when a high-level goal is complete, and then initiate the next research\_generation\_task to continue the development cycle.

By starting with this focused, verifiable auditing task, we ensure the PM agent is built on a robust and reliable foundation.