A black background with grey leaves

AI-generated content may be incorrect.

Enhancement One: Software Design and Engineering

cs 499 milestone two: enhancement one narrative

A grey logo on a black background

AI-generated content may be incorrect.

May 25, 2025

HANNAH ROSE MORGENSTEIN

Southern New Hampshire University

Professor Gene Bryant

**Artifact Description**

The artifact I selected for enhancement is a Geographic Information System (GIS) pipeline processing tool originally developed during my IT 338 – Spatial Data Analysis course in Spring 2025. The initial version was a functional prototype built with the ArcPy library in Python. It was designed to automate spatial buffer analyses around natural gas pipelines by integrating field report data with geospatial features to support infrastructure planning.

While the tool met its basic objective, it lacked modularity, portability, and extensibility—key attributes of production-ready software. For CS 499, I chose to enhance this artifact with a focus on software design and engineering, specifically improving its architecture, usability, maintainability, and overall code quality.

These enhancements were guided by early code reviews, instructor feedback, and a formal planning process. To support clean development and testing, I also created a dedicated virtual environment (gis\_env), providing a stable, isolated workspace for managing dependencies and running tests reliably.

**Justification for Inclusion in the ePortfolio**I chose to include this artifact because it demonstrates the application of core software engineering principles to a real-world geospatial data processing challenge. The enhancements completed for Milestone Two showcase my ability to design, implement, and document a robust Python application with clear architectural structure and real-world utility.

***Key enhancements include:***

* **Modular design implemented** through a clearly structured gis\_tool/ package, with focused modules such as cli.py, buffer\_processor.py, data\_loader.py, output\_writer.py, main.py, config.py, and logger.py—supporting separation of concerns and maintainable architecture.
* **User-friendly command-line interface (CLI) built** with Python’s argparse module for flexible, intuitive configuration.
* **Integrated logging** using Python’s logging module to improve observability, streamline debugging, and track execution.
* **Robust error handling** to improve stability and user feedback across different failure scenarios.
* **Multiprocessing** support to enhance performance and enable scalability when processing large spatial datasets.
* **Comprehensive inline documentation and Python docstrings** to support long-term understanding and maintainability.
* **A dedicated testing/ package** containing unit tests for each module, ensuring correctness and supporting future code changes.
* **Use of the unittest module** to scaffold and execute tests, reinforcing a test-driven development approach.
* **Creation of a virtual environment (gis\_env)** to isolate dependencies and ensure consistent, reproducible development and testing workflows.
* **Use of Git-based version control** to track changes, manage iterative development, and maintain a history of enhancements.

Among the most impactful changes was the replacement of the proprietary ArcPy dependency with open-source libraries such as GeoPandas and Shapely. This shift eliminated the need for ArcGIS, significantly improving portability and making the tool usable across a wide range of computing environments.

Collectively, these enhancements transformed the original prototype into a robust, modular, and extensible Python package well-suited for professional use in field planning workflows. This artifact now reflects modern software engineering practices and showcases my growth in building production-quality tools grounded in real-world domain needs.

**Outcomes Alignment**

This enhancement supports the Computer Science program outcome:

***"Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals (software engineering/design/database)."***

Through this enhancement, I applied **established software engineering principles** and **modern development techniques** to transform a basic prototype into a well-structured, maintainable, and scalable geospatial processing tool. Key contributions include:

* **Modular architecture** for clean separation of concerns and long-term extensibility
* **Command-line interface design** for flexible, real-world user interaction
* **Multiprocessing** to handle large datasets efficiently
* **Integration of open-source libraries (GeoPandas and Shapely)** to replace proprietary tools and broaden usability
* **Robust testing and documentation** to support reliability and collaboration

These enhancements reflect both **well-founded engineering practices** and **innovative improvements** that align with industry expectations for quality, maintainability, and usability. They also address real-world challenges in spatial data processing, delivering a tool that adds value to infrastructure planning workflows.

All planned outcomes for this enhancement, as detailed in Module One, have been fully achieved without the need for changes. This process has deepened my practical understanding of software engineering while demonstrating my ability to implement targeted improvements that meet both academic and professional standards.

**Reflection on the Enhancement Process**Reworking this artifact was both challenging and rewarding, offering a hands-on opportunity to apply advanced software design principles in a real-world context. Refactoring the tool into a modular package demanded close attention to separation of concerns, code readability, and the management of interdependencies—all of which deepened my understanding of practical software architecture. Each module was developed with clearly defined input/output interfaces, enabling independent development and streamlined testing.

Designing and implementing the command-line interface (CLI) helped me internalize the importance of user-centered design and configurability. For example, allowing users to run the tool with arguments like --input report.csv --buffer 500 --output result.txt greatly improved accessibility and flexibility.

Adding multiprocessing support was one of the most technically complex enhancements. It required deliberate handling of shared resources, process isolation, and synchronized access to output targets. This pushed me to develop a deeper understanding of concurrency and parallelism, as well as the trade-offs involved in scalable design.

Establishing a solid foundation for testing through the unittest module allowed me to think critically about testability, modularity, and reliability. I created a dedicated testing/ package with unit tests for each component, validating everything from CLI argument parsing to buffer calculations and output formatting. This was reinforced by the use of a dedicated virtual environment (gis\_env), which ensured reproducible and dependency-isolated development.

Documenting the project thoroughly using inline docstrings (in accordance with PEP 257) helped me practice clear technical communication—an essential skill for collaborative software development. Lastly, using Git for version control provided a structured approach to iteration, safe experimentation, and traceable progress throughout the enhancement process.

Overall, this process not only strengthened my coding and design capabilities but also helped me adopt a more professional mindset toward developing sustainable, maintainable, and user-friendly software tools. It has significantly prepared me for future roles where software quality, collaboration, and continuous improvement are key.

**Conclusion**

This enhancement marks a complete transformation of a basic academic prototype into a professional-grade, production-ready software tool. Through strategic modularization, the adoption of open-source libraries, implementation of robust testing, and a focus on user experience and scalability, I was able to elevate the artifact to meet real-world software engineering standards.

The process challenged me to apply foundational computer science principles—such as abstraction, separation of concerns, and test-driven development—in a practical and domain-specific context. It also gave me hands-on experience with tools and techniques that are directly applicable in industry settings, from CLI and logging design to concurrency and maintainable architecture.

By revisiting and rebuilding this GIS pipeline tool with a structured and forward-thinking approach, I not only improved its functionality but also deepened my own understanding of what it means to build scalable, maintainable, and extensible systems. This artifact serves as clear evidence of my growth as a developer and my readiness to contribute to complex, real-world software engineering projects. It is a strong representation of both my technical abilities and my commitment to thoughtful, high-quality design.