Containerized Deployment Architecture for Studying Emergent Cooperation in LLM Agents

Emily D. Carpenter
Marketing and Data Sciences
Anderson College of Business and Computing
Regis University, Denver, CO, USA
ecarpenter004@regis.edu

Background & Purpose

- Contribute to the GENESIS (General Emergent Norms, Ethics, & Societies in Silico) research project investigating whether intelligent agents (Large Language Models) develop the emergence of cooperative behaviors & moral reasoning through participation in the Iterated Prisoner's Dilemma (IPD) game theory model (Hart & Sorauf, 2025).
- Challenge: research platform is limited to a small "bare metal" compute cluster hosted at Regis University.
  - No path for deployment on other compute clusters or cloud computing solutions.
  - Agent interaction is through plain English system prompts. Question arises on how emergence occurs with prompts in different languages.
- Solution: wrap the research project code in a containerized platform for portable deployment.
  - Build research code components into Docker images orchestrated by Lightweight Kubernetes (K3s).
  - Docker Swarm was considered as an alternative solution but not pursued given time constraints, industry standard practices, and issues experienced by users of the product (Muraru, 2025; Portainer, 2026).
- Note: project builds on Practicum I work which developed and deployed a persistent storage component (PostgreSQL DB) with a Python ETL pipeline for retaining results across experiments.

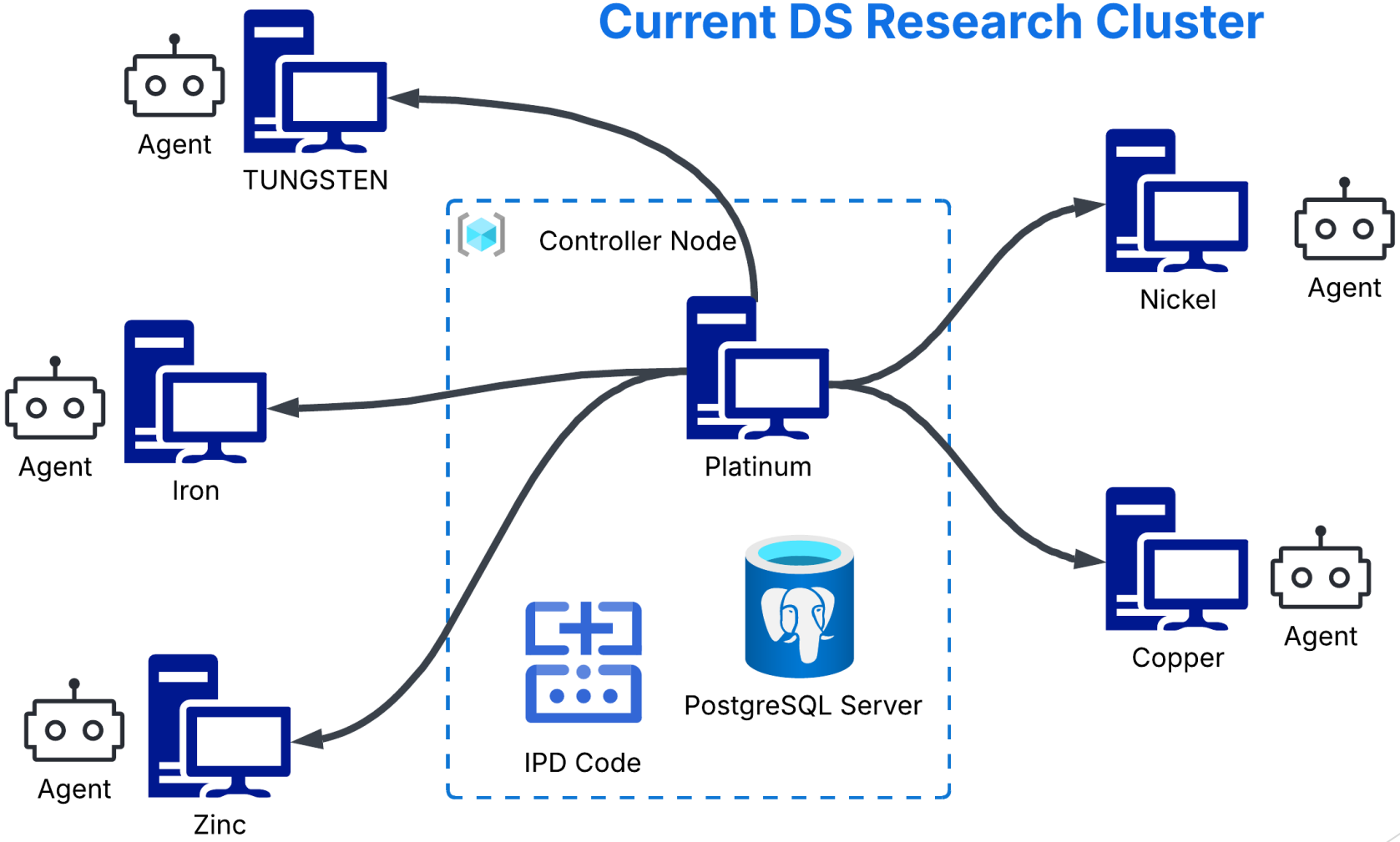
Regis University Compute Cluster

- Cluster is comprised of machines with heterogenous components. Member nodes include (by hostname):
  - **Platinum** (controller): AMD Threadripper 2950X (16-core, 3.5GHz), 126 GB RAM, NVIDIA GTX 1660 Super, 6 GB VRAM
  - **Copper**: Intel Xeon E5-2620 v4 (2.1 GHz), 126 GB RAM, NVIDIA TITAN X (Pascal) GPU, 12 GB VRAM
  - **Iron**: Intel Xeon i7-6850K (3.6 GHz), 126 GB RAM, NVIDIA TITAN X (Pascal), 12 GB VRAM
  - **Nickel**: Intel Xeon E5-2620 v4 (2.1 GHz), 252 GB RAM, 3 NVIDIA GPUs [TITAN X (Pascal); 2xGTX 1080 Ti), 34 GB total VRAM
  - **Tungsten**: AMD Threadripper Pro 9975WX (32-core, 4.0 GHz), 252 GB RAM, NVIDIA RTX 6000 Ada Generation GPU, 48 GB VRAM
  - **Zinc**: Intel i7-5930K (3.5 GHz), 126 GB RAM, 2 NVIDIA GTX 1080 Ti GPUs, 22 GB total VRAM

Compute Cluster (cont.)

- Primary research code is hosted on the Platinum node.
  - Code to run experiments is cloned and executed on node in Python virtual environments.
  - PostgreSQL database program installed and configured on Platinum.
  - Includes Ollama agent program and lightweight models (i.e. phi3:mini; phi3-mini-utility).
- Ollama agents are installed natively on each node remaining in the compute cluster.
- Cluster management requires users login remotely to each node via secure shell (SSH) to update software, download new models, and maintain operating system (Ubuntu Linux).

Current DS Research Cluster



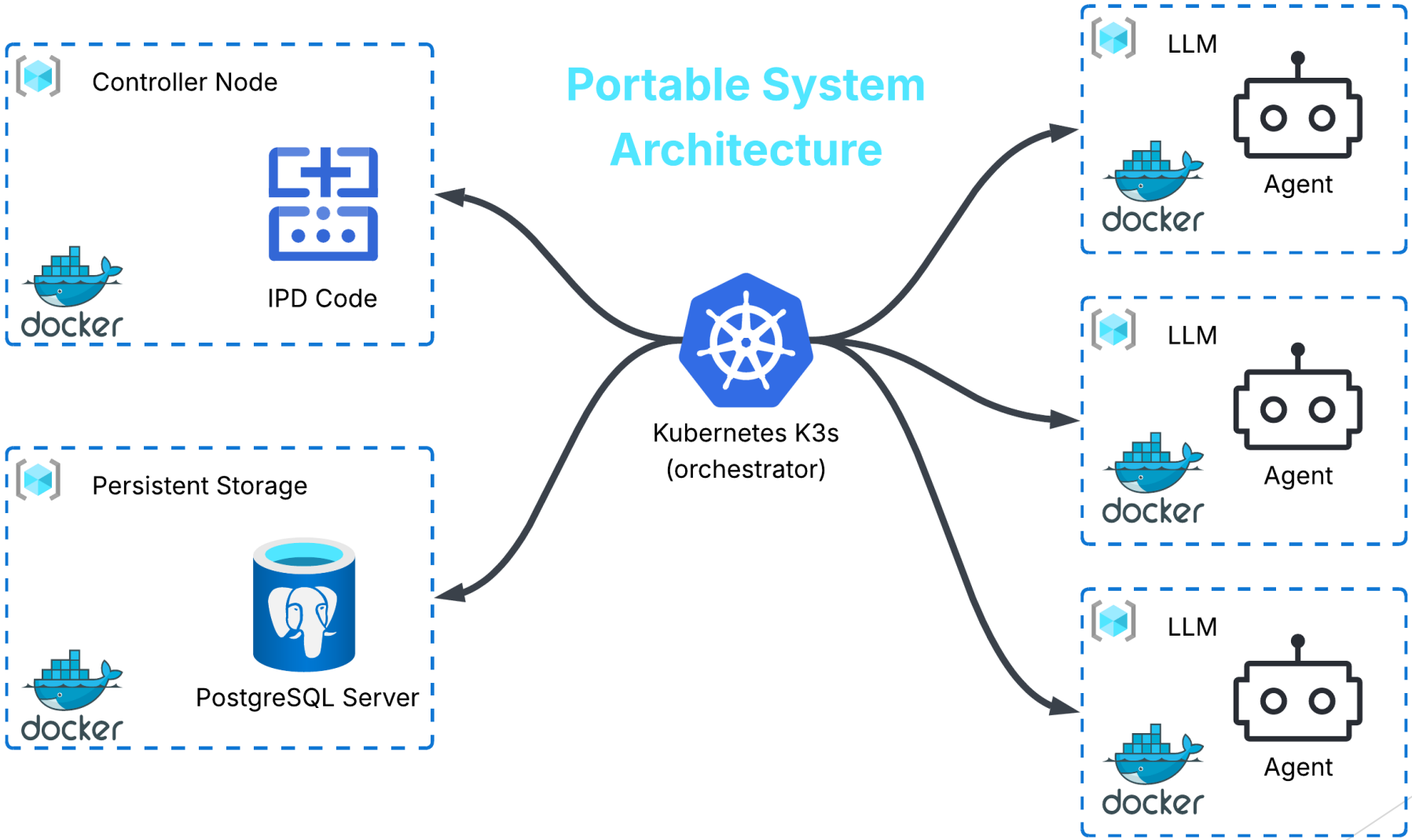
Containerized Architecture

- Docker images were built and configured to hold each component of the research project:
  - Research Python code container (forge-code)
  - PostgreSQL database storage container (forge-db)
  - Official Ollama Agent Docker image from Docker Hub (ollama/ollama)
- GitHub Actions were configured to push updates to the project GitHub repository on each push to the main branch.
  - Images hosted on repository gchr.io site through configuration of GitHub actions workflow to support Continuous Integration/Continuous Delivery (CI/CD) of software development lifecycle.
  - Allows researchers to continue developing solution while CI/CD automates process of rebuilding containers after each code commit to main branch.

Containerized Architecture (cont.)

- The Ansible application was installed and configured on the host node (platinum) to automate installation and management of the Containerized Architecture.
  - Provides ability to install (or uninstall) required applications (i.e. Docker, NVIDIA Toolkit, hosts file, etc.) across all child nodes from controller without individual SSH login required.
- Kubernetes (K3s) selected to manage deployment and execution of various components of architecture.
  - K3s is a “lightweight” version of the enterprise-scale Kubernetes (K8s) application solution designed specifically for smaller hardware systems and edge computing devices.

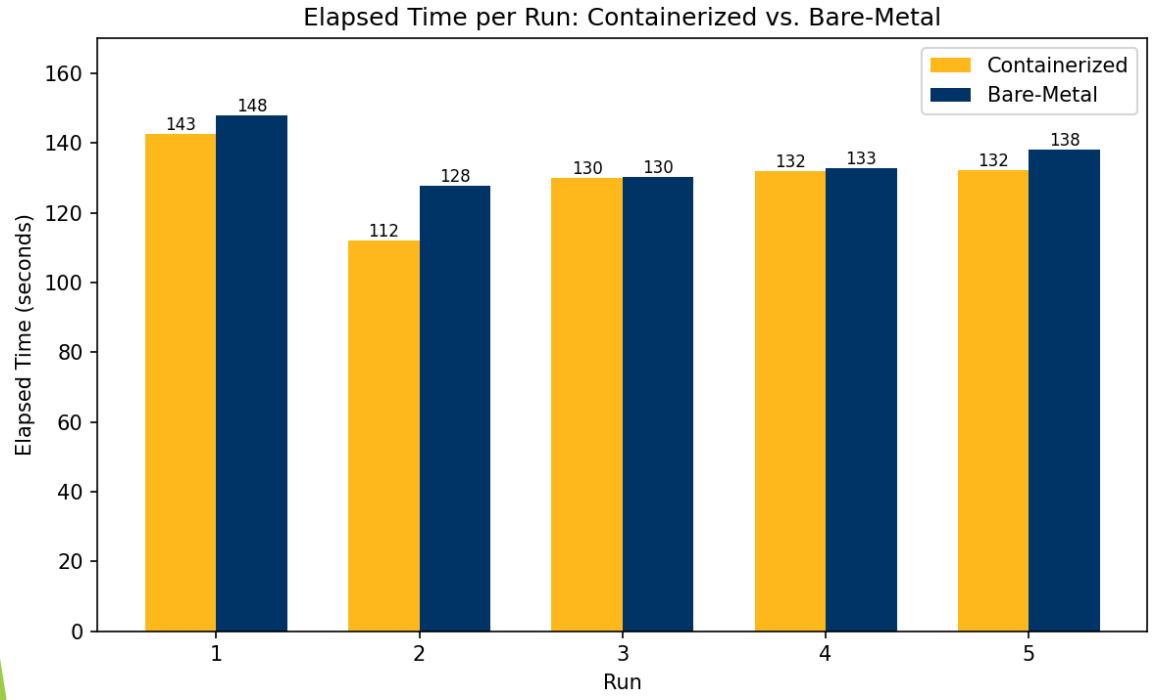
Portable System Architecture



Validation Testing

- The Containerized Architecture was built and configured to replicate the Regis University “bare-metal” compute cluster to provide a control for validation testing.
- Experiment conditions:
  - Control: bare-metal compute cluster; Experiment group: containerized architecture.
  - 5 IPD model runs, each consisting of 5 episodes of 10 rounds/episode; deterministic behavior set to lowest value (temperature=0.2) to provide consistent results across runs.
  - Ollama agents executed with the default host and model (Tungsten; Meta Llama 3 [llama3:8b-instruct-q5_K_M], 8 billion parameters).

Results



- Elapsed time to complete an experiment run was the same or better for the containerized architecture (CA) compared to the bare-metal (BM) compute cluster.
- Overall average elapsed time for containerized architecture was slightly better at 129.7 seconds vs 135.3 seconds on the bare-metal compute cluster.
- Cooperation rates for containerized architecture (0.732/0.728) were consistent with bare-metal architecture (0.776/0.764); higher values represents more cooperation.

Platform	Execution Runs	Total Rounds	Avg Elapsed Time (s)	Min Elapsed Time (s)	Max Elapsed Time (s)	Avg Cooperation Rate (Agent 0)	Avg Cooperation Rate (Agent1)
Bare-Metal	5	250	135.2689	127.6871	147.8621	0.776	0.764
Containerized	5	250	129.6929	111.937	142.5498	0.732	0.728

Conclusion

- Containerization of research code and Ollama agents did not have a negative impact to code experiment run times.
- Containerized Architecture leverages industry standard applications such as Kubernetes (lightweight version, K3s), Ansible, and GitHub actions to package the GENESIS research platform into relatively portable package hosted on GitHub.
- Research project can be cloned and quickly deployed by additional academic teams to cloud compute solutions or alternative-hardware compute clusters.
- Future work could include automation of Ollama agent models, implementation of Ansible Jinja2 templates, and large-scale testing on enterprise-level cloud compute platforms (e.g. Amazon Web Services, Google Cloud Platform, Microsoft Azure, etc…).

Architecture References

- Ansible Community. (2026). *Ansible documentation*. https://docs.ansible.com/
- Anthropic. (2025). *Claude*. https://claude.ai/
- Docker Inc. (2026). *Swarm mode*. https://docs.docker.com/engine/swarm/
- K3s Project Authors. (2025). *K3s: Lightweight Kubernetes*. https://k3s.io
- Muraru, S. (2025). *I almost ditched K3s for Docker Swarm (and why I didn't)*. https://stefanmuraru.com/blog/k3s-vs-docker-swarm/
- Portainer. (2026). *5 best Docker Swarm alternatives & why you should migrate*. https://www.portainer.io/blog/docker-swarm-alternatives
- Wahlquist, E. (2025). *K3s and K8s: Key differences and use cases explained*. https://www.suse.com/c/k3s-and-k8s-key-differences-and-use-cases-explained/

GENESIS Project References

- Axelrod, R. (1984). *The evolution of cooperation*. Basic Books. https://ee.stanford.edu/~hellman/Breakthrough/book/pdfs/axelrod.pdf
- Axelrod, R., & Hamilton, W. D. (1981). The evolution of cooperation. *Science*, 211(4489), 1390–1396.
- Haidt, J. (2012). *The righteous mind: Why good people are divided by politics and religion* (1st ed.). Vintage
- Hart, D., & Sorauf, K. (2025). *GENESIS project: From cooperation to conscience*
- Johnson, S. (2004). *Emergence: The connected lives of ants, brains, cities, and software*. Scribner
- Ostrom, E. (2015). *Governing the commons*. Cambridge University Press
- Ridley, M. (1998). *The origins of virtue: Human instincts and the evolution of cooperation*. Penguin Publishing Group

Acknowledgements

- The author wishes to thank Dr. Douglas Hart and Dr. Kellen Sorauf for the opportunity to participate in and contribute to the GENESIS research project.
- Containerized Architecture and associated documentation was developed with the assistance from Claude (Anthropic; model: Opus 4.6). All code and content reviewed, edited, and approved by the author.

Questions?