Modular OODA-Based Agents for Scalable and Adaptive AI in System-of-Systems Modelling and Simulation

Jorge L. LOVACO[1, a, *] and Karl ZUBER[1, b, *]Raghu Chaitanya MUNJULURY[1, 2, c]Christopher JOUANNET[1, 2, d]Petter KRUS[1, e]

[1]Department of Management and Engineering, Linköping University, Olaus Magnus väg, Linköping, Sweden
[2]Saab Aeronautics, Bröderna Ugglas gata, Linköping, Sweden
[a]jorge.lovaco@liu.se, [b]karl.zuber@liu.se, [c]raghu.munjulury@liu.se, [d]christopher.jouannet@liu.se, [e]petter.krus@liu.se

Keywords: *System of Systems; Agent-Based Modelling and Simulation; OODA loop; Wildfire Suppression; Collaborative AI*

Abstract Agent-Based Modelling and Simulation (ABMS) is widely used in System-of-Systems (SoS) studies to represent constituent systems operating in dynamic environments. However, the absence of standardised agent architectures hinders scalability, behaviour traceability, and comparative evaluation of emergence, coordination, and operational performance. This work introduces a modular agent design based on *Observe–Orient–Decide–Act (OODA)* loops as a framework for SoS simulations, enabling transparent information flow and Artificial Intelligence (AI) integration at the decision layer. A wildfire suppression scenario is used for the study, modelling firefighting crews, aircraft, helicopters, bulldozers, and an incident commander as OODA-driven agents. Results show that OODA-based agents exhibit coherent and traceable team behaviours, adaptive task switching, and communication-driven coordination, supporting the study of emergence and interoperability in SoS operations. The *Decision* stage is designed for future improvements in the form of learning-based AI and Large Language Models to enhance autonomy and operational fidelity.

Introduction

The advancements of Artificial Intelligence (AI) are opening new possibilities in the field of product development. A proper understanding of the complexity of a new product and all its subsystems could greatly benefit from the new AI-enabled technologies. Systems Engineering is strongly correlated with the product development process as it provides standardised methods and tools to handle the complexity of the new system in a traceable manner [1].

For the proper generation of requirements, it is important to have a clear idea and understanding of how the new system will operate and interact with other systems.

Background
System of Systems (SoS) analysis is a valuable tool for finding needs and capabilities that will be translated into requirements for new systems [2]. Communication, interoperability and performance are some of the several key aspects analysed for the collection of constituent systems that form the SoS. For these analyses, ABMS is typically used, as the resources needed to evaluate a complete operation with several constituent systems in a complex and dynamic environment can become exorbitant [3]. The focus of ABMS is the evolution and interactions of large-scale complex systems, which is also the main concern of SoS analysis. The agents in the model are a representation of the constituent systems within an SoS. The agents are implemented to mimic the capabilities and behaviours of the systems during the execution of a mission. Their behaviours will be affected by the interactions they have with one another and with the environment. This is a key part of the SoS simulation and analysis, as the interactions and behavioural changes might lead to emergent behaviours that are very difficult to predict.

Contribution
The present work considers the use of the Observe-Orient-Decide-Act (OODA) loop as the standard method for Agent-Based Modelling oriented towards SoS analysis. The OODA loop was originally proposed for military operations [4], but it provides a flexible structure for implementing agents and their behavioural rules that is easily relatable to operational guidelines. Moreover, the use of OODA loops is considered a step towards more generalised and agnostic frameworks for representing different systems. The OODA loop can be mapped into agent behaviour rules:

1. Observe: agents acquire information using sensors and external inputs.
2. Orient: agents use the observed data and compare it with the situational context.
3. Decide: decision-making logic selects optimal actions in response to the information propagated from the orientation phase.
4. Act: actions are executed, influencing the agents' state, the other agents and the system environment.

This approach is considered as a manner to ease the characterisation of SoS architectures in simulations, increases operational fidelity, and improves the scalability by allowing variable forms of AI to be included in the *Decision* phase of the loop.

Study Case
A wildfire suppression scenario is used as a case study. In this context, multiple constituent systems will be included: bulldozers, water bomber aircraft, helicopters, ground teams, and an incident commander are modelled as agents. Each of the agents will be implemented from an embedded base OODA and loop extended to their own characteristics. The agents' behaviour will change dynamically depending on the information that they receive through

communication. They will decide and act depending on the information, situation and assigned tactics. The proposed implementation structure is expected to help capture emergent effects from coordinated strategies, communication, and allow for different levels of AI sophistication to be used in the agents' decision-making processes. This generalised approach enables the flexible testing of diverse AI models, SoS architectures and their impact on system performance and operational safety. By providing an adaptable and scalable agent modelling structure oriented for SoS, the OODA-loop method helps to improve fidelity of simulations for complex missions, leaving the increment in computational resources dependent only on the sophistication of the AI used.

Implementation

The implementation of the agents attempts to follow OODA loops in such a way that, in the *Decision* step of the loop, different forms of AI could be used for extending the implementation. The steps of the loop perform were implemented as follows:

- Observe (perception/inputs): Agents gather data from the world current state such fire grid, distance fields, costs, terrain multipliers, fuel state, etc. This is done via the shared memory in terms of the software implementation.
- Orient (context building/interpretation): Agents derive context from observations: compare terrain costs, possible routes, inaccessible-gap flags, live metadata from the quadrants that split the map into a grid, or day/night thresholds for tactics. The special case is the incident commander agent world picture, who depends on the communication and information from other agents to know who is where, what is left to do, assignment counts, local completed sets, and finish-and-exit targets.
- Decide (select behaviour): The choice of what to do is mainly done by the commander:
  - Assigns targets (quadrants) and tracks completion of tasks.
  - Implemented rule tables for choosing points of interest and suppression strategy.
  - Switches between main and alternative tactics given thresholds and time-of-day.

  All agents can take decisions to some extent, and it is this step of the OODA loop that is open to the inclusion of advanced forms of AI.
- Act (executes task): Movements are done by the agents using path planning (A* search algorithm and direct-path checks) [5], and they perform tasks such as apply suppression over cells (which updates on-fire pixels) or transport other agents (this task is only realizable by helicopters). In the case of low fuel, the perform actions for returning to base and refuel.

The OODA loop shown in Fig. 1 is the most basic form of it, but there are versions were intents have been added between *Orient* and *Decide*, with continuous evaluation of knowledge and beliefs, in order to separate desires and orders [6].

Discussion

The agent-based model with the agents based on the OODA loop allowed to represent important topics for studying SoS. Collaboration and communication to reach a common goal

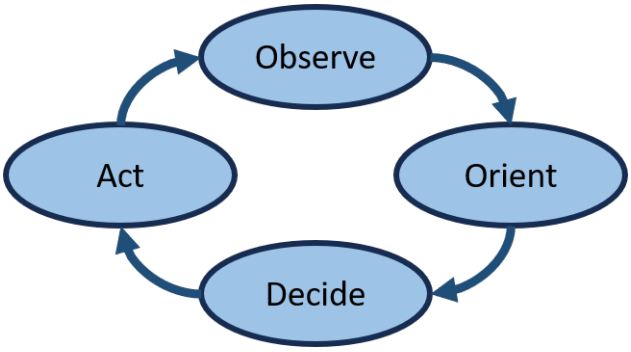

Figure 1: *Observe-Orient-Decide-Act loop used by agents.*

are particularly difficult, as any form of information exchange for taking decisions based on prior knowledge require to use some form of AI. For the present case, decision trees have been used, which limit the decision capabilities and are considered an AI in narrow form. However, the results were satisfactory since agents performing tasks are aware of their surroundings and react and communicate to events. An example of this can be seen in Fig. 2, where

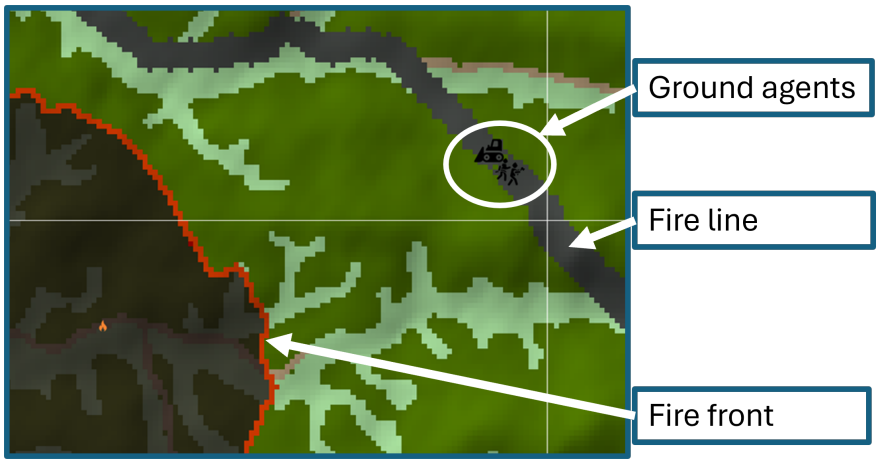

Figure 2: *Agents collaborating during a simulation of a wildfire suppression case.*

ground agents are working for closing a fire line to stop the fire front from spreading. The agents are assigned to a location by the commander agent to create the fire line in a certain direction. The simulation showed that, if they observe another agent working in the line and they close it, they stop the current action, communicate the completion and request another task. In the example of the figure the bulldozer will drive to a different location and the firefighters will request transport to their next location. The implementation, nevertheless, is still the most basic form of OODA loops with a very simple form of intelligence, which in terms of operational credibility can generate situations that would be open to debate with stakeholders and experts. For example, when many firefighters are assigned to work on a line at a certain area, they try to work in parallel, leading to situations where they become less efficient due to a lack of advanced forms of planning.

A further advantage of the OODA-based agent design is that it enables traceability of decisions and actions during the simulation. Each stage of the loop generates information

that is exchanged and recorded between agents, forming a continuous flow of observations, decisions, and actions. These communications are automatically stored in a text-based log, creating a chronological record of assignments, acknowledgements, and task completions. This allows reconstruction of how the commander's decisions propagate through the SoS and how agents respond to changing conditions. The sequence also supports post-simulation analysis of coordination efficiency, task timing, and communication reliability, which are essential for validating and improving the models.

Table 1: *Agents communication during a simulation of a wildfire suppression case.*

Time (hh:mm:ss)	**Sender**	**Receiver**	**Message Summary**
15:03:30	Commander	Hotel-1	Assigned suppress in Q53
15:03:30	Commander	Sierra-1	Assigned suppress in Q53; helper
16:05:43	Hotel-2	Commander	Reported Q53 task complete
16:10:04	Sierra-2	Commander	Reported Q53 task complete

The modular OODA structure also strengthens how communication is modelled across architectural layers of a System of Systems. By explicitly separating *Observation*, *Orientation*, *Decision*, and *Action*, it defines clear points where information must be exchanged between agents and higher-level systems. This mirrors real operational architectures, where systems contribute to and depend on shared situational awareness. In this way, OODA provides a unifying logic that links system-level design with software and agent-based implementation. Each message or data exchange can be associated with a stage of the *Decision* cycle, improving consistency between intent and execution while enhancing interpretability of communication behaviours.

Despite these advantages, some limitations remain. The conceptual OODA framework is flexible and continuous, allowing entities to operate at different speeds, but in the agent-based simulation all loops run synchronously on a shared time step. This simplifies computation yet limits realism, as agents react simultaneously rather than at their own decision rates. In real operations, commanders and field units act asynchronously, and these temporal differences shape coordination and response dynamics. Future work should introduce variable or event-driven update cycles so that agents process information and decide independently while maintaining coordinated communication within the OODA framework.

Conclusion and Future Work

The implemented model has shown that using OODA loops as framework results in agents with behaviours that are closer to those of humans. The different steps in the loop can be implemented separately, making the code easier to maintain and understand. Moreover, the information that is propagated from one step to the next of the loop can be traced, making it possible to understand if the amount of it is correct, which pieces must me communicated and how this affects the behaviour of an agent representing a constituent system. For an SoS, where there are many interactions that results in its overall behaviour, it is important to understand what the constituent systems need from the group and what they bring to the group value.

The *Decision* part of the loop is the most relevant part for including different levels of AI as future work. Starting from a narrow form of AI, different methods can be used, such as

Machine Learning, for training agents in a certain situation. However, this would be a very challenging task. Following the case of a fire suppression operation, firefighters and pilots are trained in real life both as individuals and as a group, but they also need to follow the orders of a commander. It is then important to be able to include communication into the training, so they learn by using both individual and group knowledge. With the recent advancements in AI, the OODA loops could also include image recognition or forms of Large-Language Models for including outputs similar to human communication, which brings another layer to the research about relevant communication and information in SoS, in order to avoid miscommunication or knowledge gaps that will affect the group performance and the resilience of the SoS.

ACKNOWLEDGEMENTS

The research presented in this paper has been performed in the framework of the COLOSSUS project (Collaborative System of Systems Exploration of Aviation Products, Services and Business Models) and has received funding from the European Union Horizon Europe programme under grant agreement No. 101097120. The Swiss participation in the Colossus project is supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number 22.00609. The authors would also like to acknowledge the Swedish Innovation Agency (VINNOVA) and The Swedish Defence Materiel Administration (FMV) for financial support through the grant 2019-05371. Finally, the authors want to acknowledge the insightful discussions about the topic with Kristian Amadori.

References

[1] Brusa, E., Calà, A., & Ferretto, D. (2018). Systems engineering and its application to industrial product development. Cham: Springer International Publishing.

[2] DeLaurentis, D. A., Moolchandani, K., & Guariniello, C. (2022). System of systems modeling and analysis. CRC Press.

[3] Murphy, K. J., Ciuti, S., & Kane, A. (2020). An introduction to agent-based models as an accessible surrogate to field-based research and teaching. Ecology and evolution, 10(22), 12482-12498.

[4] Enck, R. E. (2012). The OODA loop. Home Health Care Management & Practice, 24(3), 123-124.

[5] Russell, S. J., Norvig, P. (2018). Artificial intelligence a modern approach (4th ed.). Boston: Pearson.

[6] Silvander, J., & Angelin, L. (2019). Introducing intents to the OODA-loop. Procedia Computer Science, 159, 878-883.