

Synchronizing Multi-Threaded Activity under Pressure: A Simulation Study of Contingency Response in Envisioned Air Traffic Management Operations

Abhinay Paladugu

abhinaypaladugu96@gmail.com

The Ohio State University

Alicia Fernandes

Mosaic ATM (United States)

Martijn IJtsma

The Ohio State University

Research Article

Keywords: Modeling and Simulation, Computational Methods, Envisioned Systems, Concept of Operations, Reconfiguration, Discovery, Exploration, Synchronization and Validation

Posted Date: October 10th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9839654/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

1 Synchronizing Multi-Threaded Activity under
2 Pressure: A Simulation Study of Contingency
3 Response in Envisioned Air Traffic Management
4 Operations

5 Abhinay Paladugu^{1*}, Alicia Fernandes² and Martijn IJtsma¹

6 ¹Integrated Systems Engineering Department, The Ohio State
7 University, 1971 Neil Ave, Columbus, 43210, Ohio, United States of
8 America.

9 ²Mosaic ATM, Leesburg, , Virginia, United States of America.

10 *Corresponding author(s). E-mail(s): abhinaypaladugu96@gmail.com;
11 Contributing authors: afernandes@mosaicatm.com; ijtsma.1@osu.edu;

12 **Abstract**

13 This paper discusses the use of modeling and simulation during the design and
14 exploration of an envisioned aerodrome reconfiguration operation to uncover
15 synchronization issues arising from interactions between work and the work envi-
16 ronment. Large-scale, complex systems involve coordinated work that should be
17 investigated further before moving towards prototyping and deployment. Such
18 investigations typically rely on static models of work, which do not account for the
19 synchronization required in complex, dynamic work environments. Short-range
20 low-altitude air mobility operations are an example of a complex, dynamic sys-
21 tem. The paper shows how early-stage evaluation of these complex systems, using
22 work modeling to explore variables such as communication lag, wind report gen-
23 eration frequency, and the decision-making authority for rerouting, can uncover
24 requirements for aerodrome reconfiguration during a weather-related emergency.
25 The results show that the communication lag in the system can lead to unex-
26 pected outcomes, with no lag being slower to complete the mission than with
27 a lag. The exploration of vertiport reconfiguration was expanded to account

for changes in wind report generation frequency and how those changes would affect lag, revealing three distinct patterns in which communication lag would increase or decrease reconfiguration time. The paper summarizes all the synchronization requirements arising from interactions between the work and the work environment.

Keywords: Modeling and Simulation, Computational Methods, Envisioned Systems, Concept of Operations, Reconfiguration, Discovery, Exploration, Synchronization and Validation.

1 Introduction

Concepts of operations for aviation systems and air traffic management envision a shift in how aviation systems operate, including a cooperative approach to airspace management that requires multiple actors (controllers, pilots, dispatchers, etc.), both automated and human, to coordinate and synchronize activities under a wide range of operating conditions. The shift to highly automated systems and cooperative airspace management will require special consideration of how actors will coordinate during planning for nominal operations and during adaptations to address disruptions caused by contingencies or environmental dynamics. Advanced Air Mobility (AAM) is one such new operational concept that has received considerable attention from industry and government over the last few years, also under the names of Urban Air Mobility (UAM) and UAV Traffic Management (UTM) [1–4]. In efforts to enable short-range, low-altitude AAM operations, design, testing, and evaluation of operational concepts are underway, including defining workflows and procedures for cooperative air traffic management in congested airspaces, such as around major airports [5]. This includes developing procedures for responding to disruptions, such as defining partially automated flight rules for go-arounds and missed approaches ([4]).

During disruptions, such as unexpected wind shifts, multiple actors must adapt and resequence interdependent work. The literature in cognitive systems engineering and distributed work shows that the feasibility and performance of a distributed work

56 system depend on multiple actors synchronizing multiple threads of activity [6, 7]. For
57 example, reconfiguring operations around vertiports necessitates synchronized infor-
58 mation sharing for successful performance. Due to the high tempo of these operations,
59 communication lags or delayed update cadences can result in coordination break-
60 downs. Coordination breakdowns can occur when actors are unable to keep up with
61 their demands (decompensation), and actors in the system pursue local goals in ways
62 that adversely affect other actors’ local goals or even the overall system goal (locally
63 adaptive, globally maladaptive) [8, 9]. These breakdowns can create systemic synchro-
64 nization issues, such as overtaking or race conditions, caused by a lack of real-time
65 information.

66 Even though synchronization of interdependent activities is critical to the fea-
67 sibility and performance of these envisioned operations, few efforts to develop new
68 concepts consider the dynamics of how a collection of human and automated roles in
69 the joint cognitive system will synchronize activities over time. In the development of
70 new operational concepts, the roles and sequences of activities are typically described
71 textually or diagrammatically, including how new forms of automation will interact
72 and coordinate with human roles. However, because the behavior of these operational
73 systems is inherently temporal, designing for coordination requires an explicit consid-
74 eration of its temporal behavior [10]. Most distributed operations rely on a carefully
75 coordinated set of activities, synchronized to ensure proper time and resource manage-
76 ment. Maintaining this alignment across a wide range of conditions requires managing
77 the trade-off between time and coordination demands. For example, excessive time
78 spent on coordination increases the risk of exhausting the time required to handle the
79 incident. Conversely, prioritizing immediate “safing” actions without synchronization
80 risks agents acting on inconsistent information [7].

81 Computational simulations provide techniques for exploring coordination and
82 synchrony demands early in the development of new operational concepts and coordi-
83 nation mechanisms for maintaining performance when unexpected events disrupt the
84 nominal flow of work. This paper applies the computational modeling approach pre-
85 sented in [11] to evaluate the work dynamics of envisioned cooperative management
86 of congested airspace in the vicinity of a vertiport. The modeling focuses on exploring
87 design tradespaces for enabling effective reconfiguration of the traffic flow in response
88 to a disruption, which requires all actors to coordinate with each other. The study
89 highlights how the modeling approach helps to discover synchronization requirements
90 and emergent properties of envisioned operations early in the systems engineering
91 cycle.

92 2 Background

93 All operations are, at their core, coordinated work, caused by the inevitable distribu-
94 tion of tasks due to necessity, enrichment, coercion, efficiency, resilience, or collegiality
95 [12]. This distribution of work creates mutual dependence that needs to be managed
96 [13]. Maintaining performance of the distributed work system requires coordination
97 among agents to synchronize actions [12, 14]. For example, the actions of pilots, dis-
98 patchers, controllers, and traffic managers should be synchronized; without it, the
99 results can lead to inefficiency or (possibly severe) safety issues. A critical instance of
100 such mis-synchronization occurred involving Boeing 727-200 from Dallas Fort Worth
101 (DFW) to Miami [15]. In this case, the pilot, dispatcher, and the current controller
102 filed and approved a reroute due to weather near Miami International Airport. This
103 reroute was later rejected by the receiving controller, likely due to insufficient weather
104 information. The pilots accepted the rejection and returned to the original route,
105 assuming the controller had updated weather information, while the dispatcher also

missed it due to other aircraft and other situations. This placed the crew and everyone on board in danger, as they are now south of Miami International Airport with no alternative airports to land at, all while running low on fuel. Though the aircraft finally broke through the line of thunderstorms, this incident shows how mis-synchronization among agents in the system can lead to unsafe situations.

The system's different threads of activity should also be synchronized to keep pace with physical processes in the environment that drive future work demands. While it is essential to synchronize the threads of activity across agents in the system, it is also essential to synchronize those actions with the environment's pace. The complexity of synchronization can only be revealed when the different threads of work are situated in their operating environment [16].

Work dynamics are the emergent patterns of activity over time that result from the interplay of interactions between the work and work environment over time [17]. Situated work is the view of work in the context of the work environment, where work is a way of manipulating the work environment, which in turn drives and constrains subsequent activity [16]. The interactions in situated work are characterized by dependencies, such as manipulating physical artifacts or the flow of information, that determine an order to the work. For instance, if Work Element A requires a resource currently held by Work Element B, an interdependency arises in which B must wait for A to release that resource.

Early systems engineering focuses on identifying, clarifying, and documenting the system's concepts of operation, known as concept selection [18]. The concepts of operations (ConOps) are effective communication tools that outline potential areas of operation and stakeholders' ideas across the different stages of implementation [18, 19]. Concepts of operation are focused on driving exploratory research into the technological feasibility within the defined problem space. The attempt to envisioned operations

132 with a heavy focus on technological feasibility can drive ConOps into the trap of over-
133 simplification, where the operations are defined away from the complexity of work and
134 work dynamics [20, 21]. Grounding the concepts to reveal the complexities requires
135 more than a static understanding of the envisioned work; it also requires understanding
136 how the work is dynamic [11, 22].

137 The typical systems engineering approach to pre-deployment verification tends to
138 be technology-focused and resource-intensive. Existing methods for systems engineer-
139 ing decompose large complex systems into a set of discrete entities. The methods
140 identify the functions of the entities and their predicted interactions to create groups
141 in which the functions are tightly coupled, i.e., maximizing interactions within strongly
142 interacting groups [18, 23]. The underlying problem is that the evaluation process,
143 which uses a positive test strategy, inherently fails to detect other system issues, such
144 as synchronization requirements [24]. While large-scale simulators are useful for verify-
145 ing the system’s core functions under tightly controlled conditions, their strict process
146 definition and focus on positive test strategy are precisely what limit the discovery
147 of synchronization requirements. While integrating humans is recognized as a critical
148 early-stage activity, it is often reduced to an afterthought [18].

149 Design of ConOps must consider time, as synchronization is inherently temporal
150 (e.g., [25, 26]). In distributed work systems, multiple threads of activity unfold over
151 time and alter the work environment, necessitating the management of synchronization
152 in response to work dynamics. A large part of effective work is about anticipating and
153 taking action at the right time in a continuously changing context [27]. For instance,
154 lag in a critical activity can cause cascading effects on downstream activities and result
155 in saturation. Saturation is the exhaustion of the agents’ or the system’s ability to
156 handle increasing demands [8]. Saturation can further aggravate the synchronization
157 issue, causing operations to fall even further behind the system and leading to a full
158 system breakdown. On the contrary, the ability to anticipate the need for critical

159 activity can help operators by creating opportunities to address unexpected actions
160 ahead of time and reduce the risk of bottlenecks.

161 Developing ConOps without considering the work environment risks missing the
162 complexity introduced by functional coupling and physical processes. An analysis that
163 ignores the work, the work environment, or the work dynamics is bound to fail to dis-
164 cover the synchronization requirements of a distributed work system. For example, in
165 the Boeing 727-200 example, the system did not keep pace with the external processes
166 (the weather, aircraft dynamics and the fuel), the weather settled and the pilots were
167 able to land, which would otherwise have resulted in a potentially fatal situation [15].

168 To be effective in the design of envisioned operations, evaluation techniques should
169 focus on exploring work dynamics in parallel with technology-based exploration. Tra-
170 ditional modeling often serves a confirmatory role, checking if a prescribed design
171 satisfies a set of pre-defined, static requirements [18]. To overcome this, it is neces-
172 sary to better understand the system’s performance under different requirements and
173 how technology and system design can support synchronization with human users
174 [24]. This can be due to a lack of low-cost techniques that can be performed along-
175 side current system engineering efforts. An exploratory approach using modeling and
176 simulation of work allows designers to “stress” the envisioned world, uncovering hid-
177 den dependencies and “trade-offs” in parallel with the technological exploration. This
178 transition from testing for compliance to searching for requirements is essential for
179 providing information that can support designing for coordination early in the systems
180 engineering process.

181 Paladugu et al. discussed a three-phase approach to discover interactions and com-
182 pare trade-offs across different designs in an envisioned system using computational
183 modeling and simulation [11]. The paper proposed a three-lens approach to inter-
184 preting simulation output to explore the feasibility, trade-offs, and vulnerabilities of
185 different alternative designs. The paper demonstrates the technique by modeling a lost

186 command and control (C2) link contingency in the UAM environment, showcasing the
187 ability of modeling and simulation to provide insight for early-stage system designers
188 and to serve as a framework to guide them through the design cycle before creating
189 full prototypes [11]. This paper applies the approach described in [11] to uncover syn-
190 chronization requirements for vertiport reconfiguration. The concept of modeling and
191 simulation for exploring synchronization issues in a system allows systems engineers
192 to move beyond simple automation unit testing to more elaborate discovery, providing
193 a systematic assessment of what will and will not work in a procedure/task.

194 3 Method

195 The concepts of operations for low-altitude, short-range aerial operations identify a
196 broad range of contingencies [1, 3, 4, 28, 29]. This study focuses specifically on aero-
197 drome reconfiguration. In this contingency, a vertiport closes or advises closure of one
198 or multiple directions of landing unexpectedly, requiring a corresponding change in the
199 route for airborne aircraft scheduled to land at the vertiport ([1, 4, 28]). The recon-
200 figuration requires a coordinated response among pilots, fleet operators, providers of
201 services (PSUs), and vertiport operators, some of which are envisioned to be human,
202 automated, or human-automation team roles. An unplanned aerodrome reconfigura-
203 tion may be necessary for various reasons, including sudden weather changes or, if
204 vertiport operations take place in the vicinity of traditional airports, changes in aero-
205 drome traffic management. In this study, a wind shift is used as a trigger to represent
206 any event that requires a change in operations, but the procedural takeaways are
207 similar regardless of the reason.

208 This work is part of a project to support systems engineers with envisioning con-
209 cepts of operations for contingency management in future short-range, low-altitude
210 air transport. Findings from the first half of the project focused on modeling a lost

211 command and control link contingency and its impact on the system [11]. The stake-
212 holders then emphasized the importance of, and need for, further understanding of
213 aerodrome operations and synchronization requirements between vertiport operators
214 and airspace users. The authors engaged in discussions with systems engineers and
215 stakeholders to envision (1) potential ways in which the work involved in contingency
216 management would be distributed among the actors in the system, and (2) alternate
217 procedures for the actors to coordinate during the reconfiguration.

218 A key design dimension for these operations concerns the locus of control for
219 dynamic (in-flight) rerouting: will it be with a single central authority (such as an
220 airspace management service provider) similar to current air traffic control, or decen-
221 tralized, with only a passive information broker and a constraint checker? Centralized
222 control is more hierarchical, with a central service provider (e.g., PSU or vertiport
223 manager) initiating a reconfiguration, dictating the direction of approach, and assign-
224 ing new operational intents (OIs) (or intended routes) for all aircraft. In a decentralized
225 control system, there is no central authority that control operations in the vicinity
226 of the vertiport; therefore, the vertiport provides wind and weather information, and
227 pilots or onboard automation must make the final decision on the landing direction
228 and negotiate new OIs as needed to safely approach and land at the vertiport. Dis-
229 tributing control raises questions about the interactions required and how the different
230 actors will need to synchronize with each other and with environmental processes and
231 constraints to maintain safe and efficient operations. The key design question about
232 control, developed through review of concept documentation and discussion with sub-
233 ject matter experts, aims to provide data that supports designers and engineers in
234 answering important questions about the vertiport vicinity operations, such as:

- 235 1. If the vertiport is uncontrolled (i.e., no ATC services), what cooperative operating
236 practices are needed to manage operations in the vicinity of the vertiport? Which

237 actors will be responsible for decision-making and information-sharing? What syn-
238 chronization requirements are created to maintain performance in uncontrolled
239 operations?

240 2. What are the implications of the use of highly automated software services to sup-
241 port the cooperative traffic management environment? How must information be
242 propagated through the system? Is the necessary information to support system
243 responses to disturbances available to decision-makers? What coordination mecha-
244 nisms are required to maintain synchronization between highly automated services
245 and humans in the system?

246 3.1 Scenario Description

247 In the baseline Aerodrome Reconfiguration scenario, five aircraft are en route to a
248 (notional) vertiport located at Dallas Fort-Worth (DFW), approaching from the south.
249 As one of the busiest airports in the nation, DFW presents a uniquely challenging envi-
250 ronment for low-altitude air operations. Specifically, the notional vertiport is squeezed
251 between two active, parallel commercial runways, creating a highly restricted UAM
252 corridor with strict altitude rules and little room for maneuver [5]. Our early simu-
253 lations revealed quickly that the aircraft need to go far north through the corridor
254 before having any room to maneuver or turn around. Therefore, the team decided to
255 explore this by using a scenario in which the winds shift as the aircraft approaches
256 the vertiport.

257 Agents in the scenario include:

- 258 • The pilots of the aircraft (formally defined as (Remote) Pilots in Command, or
259 (R)PICs, where the modeling did not distinguish between whether a pilot is onboard
260 the aircraft or flying it remotely). In the simulations, every aircraft has a human
261 pilot and onboard automation (autopilot). The onboard automation performs some
262 of the control of flight functions as specified in [11].

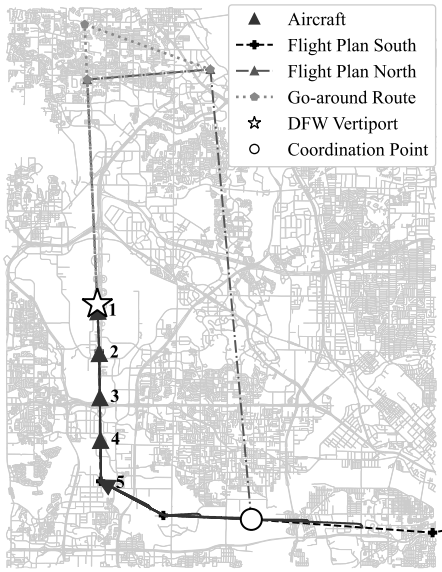
- 263 • A Provider of Services for UAM (PSU), whose involvement differs between central-
264 ized and decentralized architectures, from, respectively, an active role where they
265 have control authority over the vertiport’s approach direction and setting opera-
266 tional intent (OI) for each aircraft to a more passive role of information exchanger,
267 weather information providers, and conformance monitor. In the simulation, the
268 PSU is assumed to be a fully automated service.
- 269 • Vertiport (VP), which represents an automated service that performs constraint
270 checking. In the simulation, the vertiport represents a primarily automated human-
271 machine team implying that the automation handles most of the tasks and the
272 human will only intervene when required.
- 273 • Fleet Operator (FO), whose role in the modeled operations is to support pilot
274 decision-making in support of larger corporate fleet-related goals. In the simulated
275 model, the FO is responsible for sharing data about current vertiport and airspace
276 operations with pilots associated with their operation. In a decentralized architec-
277 ture, the FO also creates the new OI when needed by an automated pilot that
278 satisfies both aircraft and corporate constraints. In the simulation, the FO repre-
279 sents a primarily automated human-machine team implying that the automation
280 handles most of the tasks and the human will only intervene when required.

281 Figure 1 shows simulated flight paths to illustrate the scenario. In Figure 1(a),
282 aircraft 1 and 2 denote the two incoming aircraft that follow their expected approach
283 path from the south and land. The winds shift unexpectedly, while the remaining
284 aircrafts 3, 4, and 5 are on their final approach leg, see Figure 1(b), landing from the
285 south becomes impossible. The other aircrafts decide to abandon their approach from
286 the south. Aircraft 3 approaches the vertiport and then executes a go-around upon
287 detecting the unstable winds. Aircraft 4 and 5 receive communication from aircraft 3
288 describing the pilot’s decision and intent, and also decide to go around. Figure 1(c)
289 and Figure 1(d) show the paths these three aircraft take to fly to the north of the

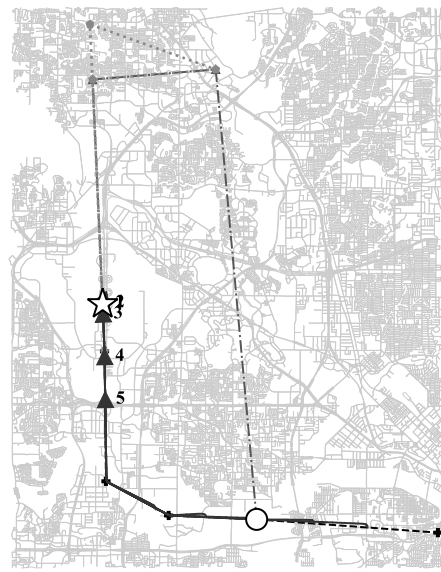
290 airport, circle until the approach path is clear, and then re-approach the vertiport
291 from the north.

292 A vertiport reconfiguration scenario may begin with an (R)PIC deciding it is
293 unsafe to land, such as due to changing wind conditions, and executing a go-around.
294 To reconfigure the vertiport operations requires coordination among various actors,
295 including other (R)PICs and service providers. The coordination starts with other
296 agents becoming aware of the change in conditions, such as the change in wind direc-
297 tion in this scenario. There are two pathways through which this information can
298 reach other agents: an update in automated weather reporting from the PSU or from
299 the aircraft operator who executed the go-around. Importantly, the aircraft operator
300 is likely to report both their intention to go around and their reason for doing so - i.e.,
301 the change in wind conditions. Communicating the reason for the go-around provides
302 the other agents with important context for their own decision-making. The time it
303 takes for the aircraft operator's intent and reasoning to reach other actors, including
304 automation, can affect outcomes, drawing attention to our second design question.
305 Thus, the time taken by a pilot after deciding to execute the go-around to complete
306 procedural steps, including communicating the reason for the go-around and sharing
307 a new OIs, is modeled as a variable in the simulation. It is assumed that the reason
308 for the go-around is communicated immediately via a voice channel, but that a delay
309 or lag may occur in this information becoming available in digital format and hence
310 known to any automated agents. The simulation is used to explore the effects of this
311 lag on the coordination of aircraft responses and the overall reconfiguration of ongoing
312 operations.

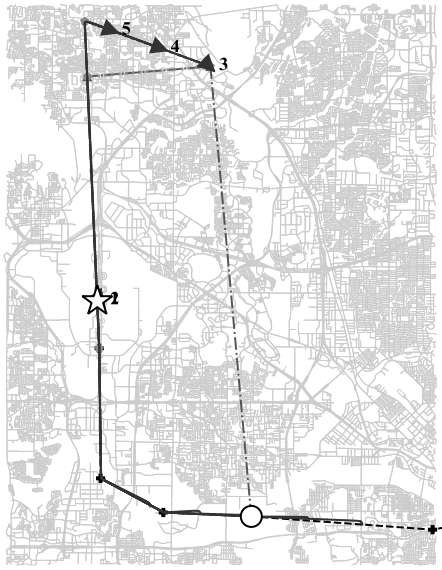
313 A functional work model is developed for this scenario by adapting and expanding
314 the models presented in [11]. The specific actions added or modified for the vertiport
315 reconfiguration are detailed in Table 1, which also describes the roles of agents within
316 both centralized and decentralized work architectures. Some of the related logic and
317 actions are described in more detail below.



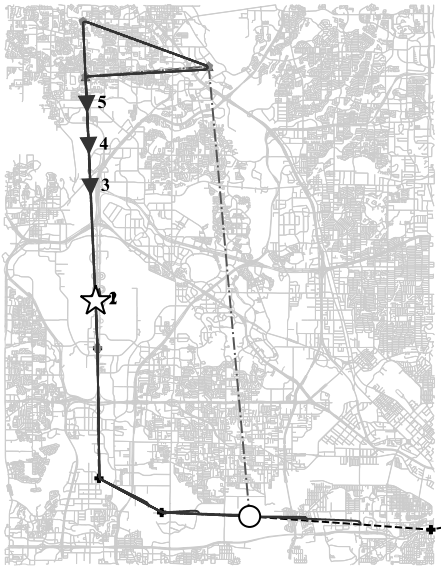
(a) Aircrafts 1 and 2 land from the south



(b) Winds change and landing from the south is not advisable



(c) Aircrafts 3, 4, and 5 execute a go-around and propose an approach from the north



(d) Aircrafts 3, 4, and 5 approach from the north

Fig. 1 Scenario description

Table 1 Key actions in the work model. * indicates agents assumed to be automated. Cent. = Centralized

Action Name	Cent.	Decent.	Description
Continue or Go-Around Decision Making	(R)PIC	(R)PIC	Takes as input current route information, current location, wind information, and previous aircraft state to decide whether to execute a go-around or land. This action is scheduled when a change in conditions, such as the wind shift, is detected during final approach. The frequency of checks for changes in landing conditions increases as the aircraft approaches landing. If a go-around is decided, it sets a temporary flight plan for the aircraft to proceed north of the vertiport, circle as needed, and then return to land from the north.
Reporting Local Wind Conditions	PSU*	PSU*	Automated action that updates the wind report every few seconds. Its timing and frequency are independent variables. It is assumed that the change in wind conditions only affects landing and not flight mechanics during the go-around. Note that the PSU is assumed to be relaying the wind information from a third-party supplemental data service provider (SDSP).
Deciding Vertiport Configuration (Direction for Approach)	PSU*	VP*	Takes as input the current wind conditions and the reason for the flight that goes around. Sets or suggests the direction of landing for the vertiport in the centralized or decentralized architecture, respectively. In this scenario, the decision is triggered by an aircraft being out of conformance with the OI shared with the distributed system as reported by the conformance monitoring function. A change in the direction of approach triggers airspace reconfiguration and consequent re-evaluation of the OIs of all other aircraft approaching (or preparing to depart from) the vertiport. The evaluation is dependent on the role of the agent executing and the access to the reason in the form of data or voice. Maintains the current order of landing to establish a priority order on who to reschedule to avoid dangerous overtaking situations. ^a
Decide Aircraft Direction for Landing	(R)PIC	(R)PIC	Takes as input the wind, intent, and reason for a previous aircraft's not landing (if available) to evaluate the feasibility of all possible landing directions. Scheduled when the aircraft is close to a point of coordination to reevaluate the intended approach, or due to an unexpected change in conditions, such as an aircraft executing a go-around (outcome of "Continue or Go-Around Decision Making"), or a system-wide reconfiguration (outcome of "Deciding Vertiport Configuration"). In an abnormal condition, if the aircraft is between 150 and 300 seconds from the scheduled landing site, the agent waits and retries this action when the aircraft is 150 seconds from landing. This logic is further expanded below, along with a visual representation as Fig. 2.
Conformance Monitoring	PSU*	PSU*	Conformance Monitoring runs every $x = 10$ seconds to check if the aircraft is where it is expected to be based on its OI. If the aircraft is out of conformance per NASA's X4 simulation rules [30], where non-conformance is defined as a 10-second telemetry loss or exceeding 1,500 feet lateral, 250 feet vertical, and 60 seconds of the estimated time of arrival within the OI, it outputs the Conformance Update. This triggers the Decide Vertiport Configuration action for the airspace and subsequent reevaluation of OIs for aircraft affected by wind change or non-conformance.
Transcribing Voice to Data	FO*	FO*	If the reason for non-conformance is communicated via voice, transcribe and provide the reason to non-voice agents. The envisioning of PSU as an automated agent highlighted the need for a function that can transcribe the knowledge shared in voice channels into data for the automated PSU to use.
Create New Operational Intent	PSU*	FO*	Takes as input all the possible waypoints and the waypoint schedules (i.e., reservations for their use by other aircraft). The action then filters the current aircraft's schedule and generates a potential new route and landing time based on the aircraft's current location and landing approach. If the PSU is creating the new OI (centralized architecture), the routes and waypoint schedules are updated immediately. If the (R)PIC creates the new OI (decentralized architecture), the updates are made only after the PSU verifies with airspace constraints and confirms the OI. If the waypoint schedules are inaccessible due to another aircraft adding its reservation, the action will be delayed by 2 seconds. ^b
Validating OI Against Vertiport Schedule	VP*	VP*	Check the proposed OI (OI1) for conformance to constraints. Gets the new OI information and verifies with the available landing times. If the vertiport is unavailable for the proposed landing time, this action returns the next available landing time slot. In the current simulation conditions, the vertiport constraints check never rejects any aircraft. This can be attributed to the airspace check that supersedes the vertiport check.
Validating OI Against Airspace Constraints	PSU*	PSU*	Gets information about the proposed OI (OI1) and verifies it against other OIs traversing the indicated waypoints. If the new OI meets the minimum buffer specified for all other scheduled crossings at the waypoints, then the new OI is confirmed. Otherwise, the OI is not confirmed, and the FO must continue the negotiation process. If there is more than one request, a first-in, first-out (FIFO) method is used.
Approve or Deny Operational Intent	(R)PIC	(R)PIC	The OI (OI2) is checked for feasibility by the PIC/RPIC/aircraft automation. If feasible, the OI (OI2) is accepted. If not feasible, a new OI (OI3) is created and sent to check the vertiport constraints. Note: The fleet operator or other actor may compute the new OI. The current simulation implementation is simplified since trajectory negotiation is not the main research interest.

¹Note that the constraint of maintaining landing order is an example of a potential cooperative operating practice, which all participants using the vertiport would need to agree to ahead of time [1].

²Waypoint schedules enforce time between aircraft to maintain safe separation. The required time between aircraft is another example of a potential cooperative operating practice.

Figure 2 describes the model of a pilot’s decision-making process based on their distance from landing, assuming $t = 0$ (the rightmost point on the timeline) represents the landing time. This model is primarily represented by the “Decide Aircraft Direction for Landing” action and its interactions with the “Land” action.

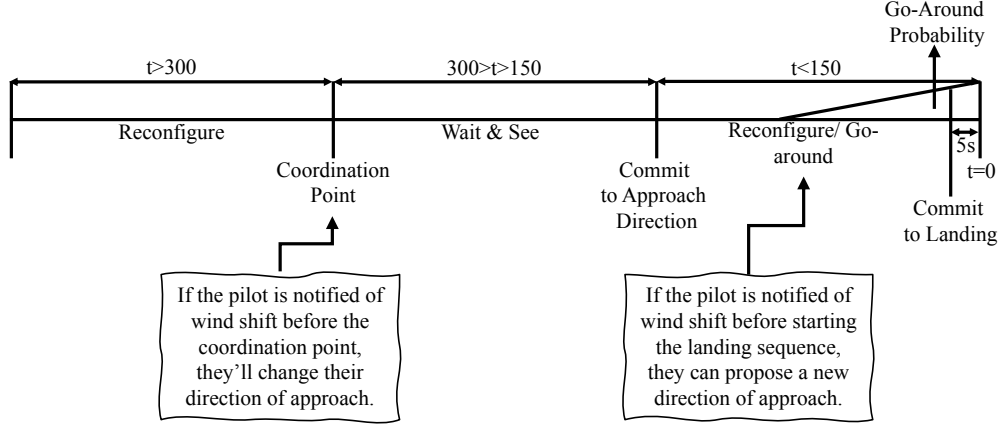


Fig. 2 Logic for the “Decide Aircraft Direction for Landing” Action as Modeled

- If the aircraft is more than 300 seconds before landing when the PIC receives the updated wind information, it is assumed the aircraft has not yet reached the coordination point (denoted by a hollow circle in Figure 1) and the PIC can still choose to approach the vertiport either from the south or from the north. In the model implementation, upon winds changing, the PIC opts to approach from the north. Note that in the current scenario, this rarely occurs, as the maximum time difference between Aircraft 1 and Aircraft 5 is 180s.
- The second option occurs if the aircraft is between 150 and 300 seconds from landing. In this range, the pilot will choose to wait and observe whether the winds stabilize. In this condition, the pilots will again reevaluate the condition when they are approximately 150 seconds from landing.

333 • The third option is when the aircraft is less than 150 seconds away from landing.
 334 In this situation, the pilots can either execute a go-around or change their route to
 335 approach the vertiport from the north. The slanted line in this interval in Figure
 336 2 represents the increasing probability of a go-around as the aircraft gets closer
 337 to landing as “Land” action is running continuously and can immediately sched-
 338 ule “Continue or Go-around Decision Making” based on the wind conditions. This
 339 probability relation arises mathematically from the interaction between the “Decide
 340 Aircraft Direction for Landing” and “Continue or Go-around Decision Making”
 341 actions, in which the pilots execute a go-around if the OI negotiation process does
 342 not complete in time. The simulation design prevents an aircraft from executing a
 343 go-around more than 120 seconds before landing.

344 3.2 Trade Space for Exploration

345 The experiment matrix for the aerodrome reconfiguration simulation uses the decision-
 346 making architecture, timing of the scenario events, pace of activity, and temporal
 347 constraints as the key independent variables. Conceptually, these variables map to the
 348 fundamental challenges of synchronizing multiple threads of activity in a distributed
 349 system. The following parameters are varied for the simulation study:

350 • Decision-making authority: who makes the decisions on dynamic rerouting as part
 351 of the reconfiguration. The main difference between the centralized and decentral-
 352 ized architectures is modeled based on the FO or PSU performing the “Deciding
 353 Vertiport Configuration” and “Decide Aircraft Direction for Landing” actions. One
 354 condition is where each (R)PIC decides their approach direction to an uncontrolled
 355 vertiport (simulating a decentralized decision-making architecture), and a second is
 356 where the PSU decides the direction for all the aircraft operating near the verti-
 357 port (simulating a centralized architecture for cooperative airspace management).

Columns 2 and 3 of Table 1 show the distribution of decision-making authority to create the two architectures in the simulation.

- The timing of the disruptive events. The time when the wind begins to prevent aircraft from landing, according to the current plan, is represented relative to the landing times of the aircraft. The net effect of this is, for each level, a different number of aircraft that must reroute and coordinate.
- Parameters relating to the pace of activity. These parameters directly influence when information is available to actors in the system. As part of work dynamics, there inherently is some “lag” in activity as actors need time to recognize the need for an action, perform that action, and—with it—generate information that others depend on. In this case, the delay between key physical or cognitive events and their digital representation is the time lag of interest. Specifically, the following parameters are varied:
 - Wind report generation frequency: The frequency at which new wind reports are generated and shared with the system.
 - Pilot go-around communication time lag: The time between the pilot deciding to initiate a go-around and information resulting from that decision being shared with the system, such as the reason for the go-around being communicated via the digital channel and a new OI for the aircraft being proposed to complete the coordination.
- The temporal constraints and margins of the operations, operationalized as the minimum time required between two aircraft passing through the same waypoint. This represents the buffer built into the system to absorb variances in timing without leading to cascading failures. This buffer value is used to validate new OIs against current airspace constraints, ensuring the buffers remain intact even after reconfiguration.

Table 2 shows the values for all the parameters. With all permutations of the first 4 columns, this creates a total of 160 cases to simulate. The levels for each parameter are based on discussions during stakeholder meetings. Though the stakeholder meetings revealed four different levels of the pilot go-around communication time lag 0s, 10s, 20s and 30s. Preliminary analysis showed similar results for different levels of the pilot go-around communication time lag, so the levels are reduced to only the 0s and 30s cases. This decision reduced the number of cases to 80. To apply between-aircraft intervals of 15s, 30s, 45s, and 60s, each level required different starting conditions in which the aircraft were strategically deconflicted by the separation values defined by the value of the between-aircraft interval. After finalizing the set of independent variables, 80 cases were simulated.

Pilot Reason Communica- tion Time Lag (Lag) (s)	Wind Change Timing (WC)	Between-Aircraft Interval at Waypoint (Buffer) (s)	Rerouting Decision-Making Authority (WA)
0	Before 1st aircraft landing	15 (0.5nm)	Centralized
30	Between 1st & 2nd aircraft landing	30 (1.02nm)	Decentralized
	Between 2nd & 3rd aircraft landing	45 (1.5nm)	
	Between 3rd and 4th aircraft landing	60 (2.0nm)	
	Between 4th & 5th aircraft landing		

Table 2 Simulation parameters and their levels

Results from the experiments indicated the need for further investigation into the synchrony requirements for maintaining the order of the aircraft. Thus, a follow-up set of simulations is performed to further examine the effects of lag on a decentralized architecture; the frequency of wind report generation is added as an independent variable. 160 simulations were additionally performed with wind report generation frequencies of 5, 7, 10, and 20s while varying other parameters as given in Table 3.

401 This data is presented only as changes in reconfiguration time relative to the lag and
 402 buffer, specifically in the decentralized case.

Pilot Reason Communication Time Lag (Lag) (s)	Wind Change Timing (WC)	Between-Aircraft Interval at Waypoint (Buffer) (s)	Wind Report Generation Frequency (s)
0	Before 1st aircraft landing	30 (1.02nm)	5
30	Between 1st & 2nd aircraft landing	15 (0.5nm)	7
	Between 2nd & 3rd aircraft landing	45 (1.5nm)	10
	Between 3rd and 4th aircraft landing	60 (2.0nm)	20
	Between 4th & 5th aircraft landing		

Table 3 The follow-up experiment matrix

403 4 Results

404 The simulation results are organized by topic. First, results on system reconfiguration
 405 performance are presented, followed by those on information exchanges. Second, results
 406 of a follow-up analysis of lag and mis-synchrony are presented.

407 4.1 System Reconfiguration Performance

408 For system reconfiguration performance, the analysis focused on three measures: the
 409 time to the last aircraft's landing, which ends the simulation run, the number of OI
 410 negotiations, and the number of go-arounds.

411 4.1.1 Last Aircraft Landing Time

412 Figure 3 shows box plots for the time at which the last aircraft lands, which from here
 413 on is referred to as completion time, in each condition. Note that each of the graphs
 414 represents a different numbers of simulation runs since each independent variable has

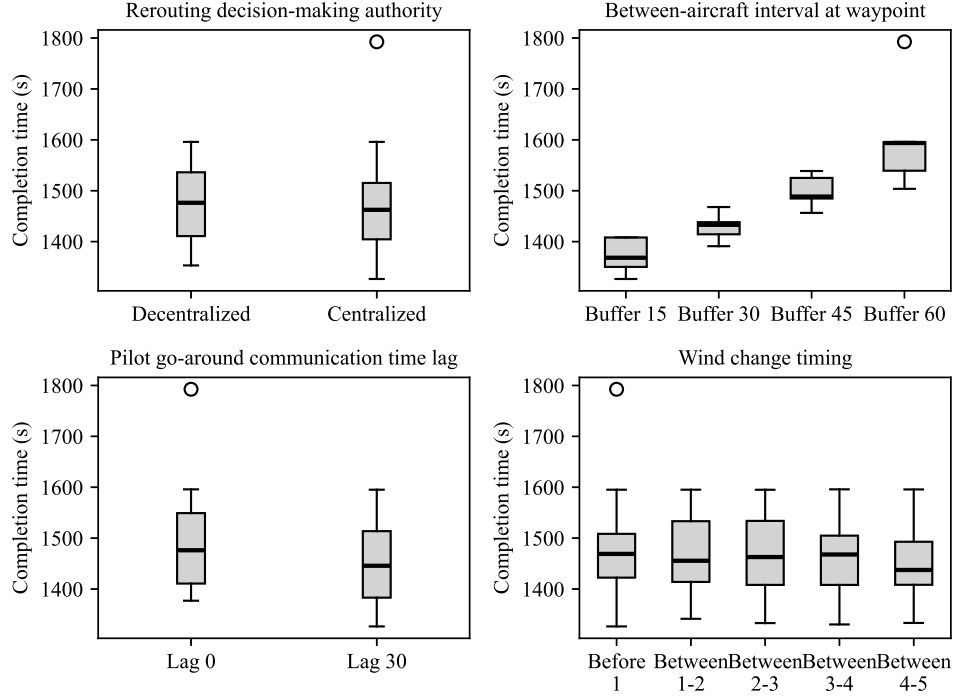


Fig. 3 Average completion time for all the simulation variables.

415 a different number of levels. For example, the top-left graph shows the completion time
 416 plot across all cases with decentralized (left bar) or centralized (right bar) architecture.
 417 Each box plot, then, represents the completion time across 40 different simulation runs.
 418 Similarly, each box plot in the top-right indicates the completion time across all cases
 419 with a between-aircraft buffer of 15s, 30s, 45s, and 60s, respectively, representing 20
 420 simulation runs. Each box plot in the bottom-left represents the average completion
 421 time across all cases with the pilot go-around communication time lag of 0s and 60s,
 422 across 40 simulation runs. Each box plot in the bottom-right represents the completion
 423 time across all cases with the wind change occurring before the first aircraft, between
 424 the first and second aircraft, between the second and third aircraft, between the third
 425 and fourth, and between the fourth and fifth aircraft, with 16 simulation runs.

426 The boxes extend from the first to the third quartile, with a thick horizontal line
427 depicting the median. The vertical black lines extending on either side of the box
428 represent the whiskers, which help determine outliers occurring beyond 1.5 times the
429 interquartile range. The outliers are represented as hollow circles outside the extended
430 whiskers. The hollow circles (outliers) in each plot represent a single simulation run in
431 which the wind change occurs early in the simulation, with a centralized architecture,
432 no communication lag, and a between-aircraft separation of 60s. This scenario results
433 in the longest completion time as the last aircraft in the sequence learns of the wind
434 change before passing the coordination point, and the PIC chooses to reroute and
435 approach the vertiport from the north instead of flying directly over the vertiport and
436 following the path of the aircraft that executed the go-around. This is the only case
437 when an aircraft chooses this longer trajectory, making it an outlier.

438 The top-left graph in Figure 3 shows that, on average, the decentralized architec-
439 ture takes a little longer to land all aircraft than the centralized architecture. The
440 between-aircraft separation buffer (Buffer) (top-right graph in Figure 3) appears to
441 have a larger impact on the time of the last flight's landing. This is because lower
442 buffer values allow aircraft to operate closer together, resulting in faster landing times.
443 Furthermore, the aircraft are initially closer to the vertiport in the 15s case than in
444 the 60s case, which also allows earlier landing times. There is some variation in the
445 last aircraft landing times across the different levels of pilot communication time lag
446 (bottom-left graph in Figure 3), but the median appears lower with lag and higher
447 without it, which is counterintuitive and warrants a need for further analysis. The wind
448 change timing (bottom-right graph in Figure 3) shows no strong pattern, although
449 a deeper analysis of the mean completion time revealed a slightly higher mean com-
450 pletion time when the wind change occurs earlier and decreases as fewer aircraft are
451 affected, thereby reducing the required coordination activities.

For further analysis of completion times for changes in between-aircraft buffer values, an additional mission duration is computed. This is an attempt to eliminate the effects of the initial spacing between the aircraft and the vertiport caused by the buffer values. The value is computed by subtracting the completion time when no wind change occurs from the average completion time for each buffer value. The additional mission durations for the aircraft separation buffers of 15s, 30s, and 45s are close to each other, 985s, 981s, and 980s, respectively, indicating minor differences due to the between-aircraft separation. The 60s separation buffer yields a higher additional mission duration of 1002. However, the 60s buffer shows an outlier. When this is removed, the additional mission duration still remains slightly higher than the others at 990s.

4.1.2 Number of OI Requests and Go-Arounds

Figure 4 and Figure 5 show the average number of OI requests excluding OI changed due to an aircraft executing a go-around and the number of go-arounds performed, respectively. As above, each condition is averaged over a different number of simulation runs: the architectures (dark blue) and between-aircraft buffer (yellow with forward diagonal stripes) each represent 20 simulation runs, the pilot communication time lag (light blue with backward diagonal stripes) represents 40 simulation runs, and the wind change time (orange with an intersecting diagonal pattern) represents 16 simulation runs. The vertical lines in Figure 4 represent the expected average number of OI requests based on scenario design, assuming no negotiation conflicts occur. Negotiation conflicts can arise when an aircraft proposes a new OI that overlaps with another aircraft's proposal, conflicts with any existing reservation that has not yet been released, or interacts with the execution of a go-around.

Centralized vs Decentralized Architecture: The centralized architecture has exactly the expected average number of OI requests. In the centralized architecture, new OIs are computed with complete knowledge of all other proposed OIs, preventing

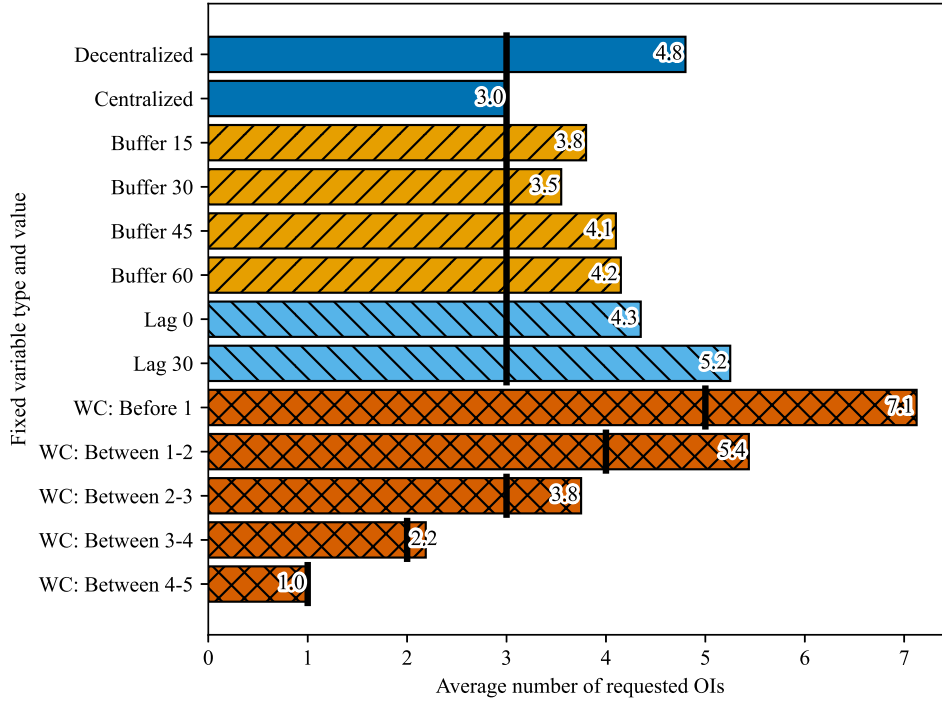


Fig. 4 Average number of OI requests (excluding go-around OI change) for all simulation variables.

conflicts (and the subsequent need for negotiation). On average, the decentralized architecture includes a greater number of OI requests than the centralized architecture, which is caused by a larger number of rejected OI requests and the need for further negotiation between aircraft in the decentralized architecture. Inspection of individual simulation runs revealed that this is caused by interactions between the initial OI requests sometimes being based on incorrect weather information and disorder in the timing of aircraft requesting new OIs, which can have ripple effects when revised OIs are proposed later, after more slots have already been taken by others. This is analyzed and explained in more detail below, as well as in sections 4.3 and 4.4.

Between Aircraft Buffer: Data for between-aircraft buffer (yellow bars with forward diagonal stripes in Figures 4 and 5) show opposite patterns for number of

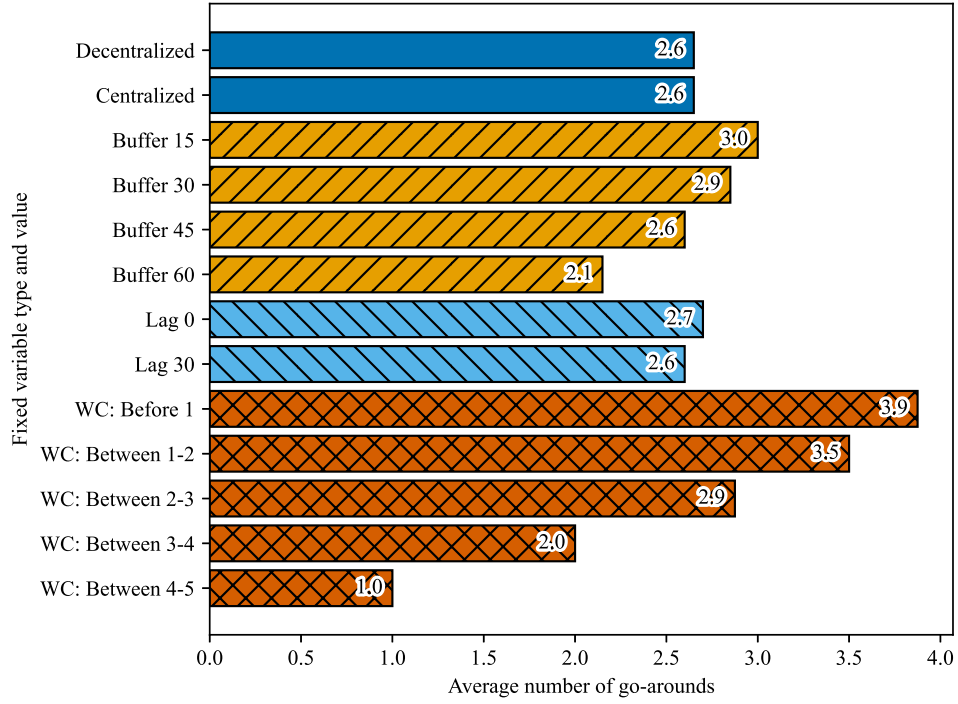


Fig. 5 Average number of go-arounds for all the simulation variables.

490 OI requests and number of go-arounds. Lower buffers show a higher number of go-
 491 arounds because the aircraft are closer together and have less time available to request
 492 and implement new OIs after learning of the wind change from the initial go-around
 493 aircraft before beginning their landing sequence. This increases the number of aircraft
 494 that have to execute a go-around. The number of OI proposals shows a decrease
 495 followed by an increase with increasing buffer. This pattern is caused by the interaction
 496 of an increased number of go-arounds combined with the waypoint availability. The
 497 OI proposal created by a go-around occurs at a time that may differ from when the
 498 others are replanning and proposing updated OIs. The probability of the difference
 499 will increase when the aircraft buffer is higher due to the increased distances. Similarly,
 500 as the buffer size increases, the number of slots decreases, making the OI negotiation
 501 more difficult. For a 15 second buffer though there is high availability and increased

502 go-arounds, the requests are still concentrated due to the close spacing compared to
503 the 30 second buffer. With the 45 and 60 second buffer there is a lack of waypoint
504 availability which increases the OI requests. This pattern is observed only in the
505 decentralized architecture, as the centralized architecture does not face the problem
506 of aircraft OI negotiations.

507 **Pilot Reason Communication Lag:** Analysis of the effects of communication
508 lag on average number of OI requests and go-arounds (light blue bars with backward
509 diagonal stripes in Figures 4 and 5) shows a slightly lower number of OI requests and
510 go-arounds without lag. The effect of lag on the number of go-arounds is negligible
511 because reroutes for other flights are not initiated immediately, regardless of whether
512 there is lag. When there is a lag, reroutes for other flights are not initiated immediately
513 because the reason (wind change (digital)) is not distributed throughout the system
514 in a timely manner, which therefore does not trigger other aircraft to propose updated
515 OIs. When there is no lag, reroutes for other flights are not initiated immediately
516 because the initial attempt to compute a new OI lacked the updated wind information.
517 This is explained in more detail in section 4.3. The increase in the number of requested
518 OIs is solely due to the decentralized architecture. The increase in the requested OIs
519 due to the communication lag is caused by ordering and race conditions, as further
520 expanded in section 4.3.

521 **Time of Wind Change:** The effects of wind change timing are shown in the
522 orange bars with intersecting diagonal pattern in Figures 4 and 5. The earlier the wind
523 change occurs, the more aircraft are in the air and affected by the change. Aircraft
524 affected by the wind change are expected to submit at least one OI request. The orange
525 bars with intersecting diagonal pattern in Figure 4 represent the average across both
526 architectures; however, this average obscures a nearly two-fold increase in OI requests
527 in the decentralized architecture when the wind change occurs before the first aircraft.
528 The number of OI requests gradually decreases with the wind change occurring later,
529 and both centralized and decentralized architectures have the same number of OI

530 requests (one) when just one aircraft is affected (wind changes between the fourth and
531 fifth aircraft). Thus, as is to be expected, in denser airspace, more OI negotiations are
532 required to reconfigure the vertiport operations. It is a noteworthy finding, though,
533 that in the decentralized architecture, the number of OI requests increases non-linearly
534 with the number of aircraft in the airspace.

535 4.2 Information Exchange Requirements

536 Figures 6 and 7 show the required information exchanges from two simulation runs
537 that differ only in architecture: centralized and decentralized, respectively, with all
538 other parameters held constant. In both networks, nodes represent agents and edges
539 denote information exchange requirements, with edge thickness indicating the num-
540 ber of information exchanges between the agents throughout the simulation run.
541 Both figures also show network analysis metrics. Both architectures show a significant
542 increase in the number of agent pairs requiring information exchange compared to 19
543 agent pairs requiring information exchange in the nominal no-wind shift condition. In
544 contrast, with a wind shift, the centralized and decentralized architectures have 33
545 and 47 agent pairs with information exchange requirements, respectively. This high-
546 lights the importance of having established communication channels available, even if
547 they are only used in off-nominal situations.

548 The high modularity in both conditions indicates that information requirements
549 are dense among agents working on a specific aircraft, while sparse with outside actors.
550 Transitivity is a measure for the amount of “information sharing triads” relative to
551 all possible triads, where an information sharing triad indicates three agents that all
552 share information with each other (as opposed to a triad where two of the agents
553 do *not* exchange information). Lower transitivity can be interpreted as information
554 sharing being more hierarchical (more agents share information only with a common
555 third agent, not with each other). On the other hand, higher transitivity implies that

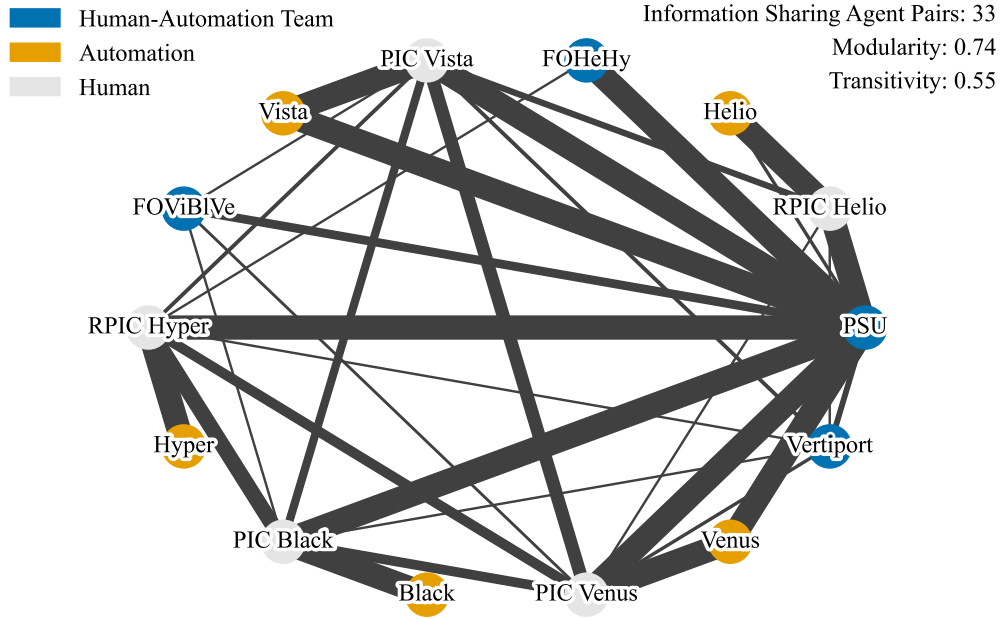


Fig. 6 Information exchange networks for simulations with wind change between Aircraft 1 and 2, including a 30s pilot communication lag and a 60s separation buffer.

each of the three agents must interact with the others. As would be expected, the centralized architecture has a relatively higher transitivity. Under higher transitivity, centralized architecture, it is expected to see a higher synchronization effort.

4.3 Further Analysis of Lag

An unexpected finding from the aggregate analysis is that—although the difference is small—a lag in communication of the pilots’ reason for a go-around and filing of new OI resulted in *lower* average completion time, see Figure 3. This finding, together with the interest in exploring the coordination and synchrony requirements between agents during a critical moment in the reconfiguration, was a motivation for a deeper analysis. Inspection of individual simulation runs revealed that in the no lag conditions, automated systems for requesting a new OI are triggered immediately (no lag), sometimes *before* the wind report is updated to reflect the new wind direction. While

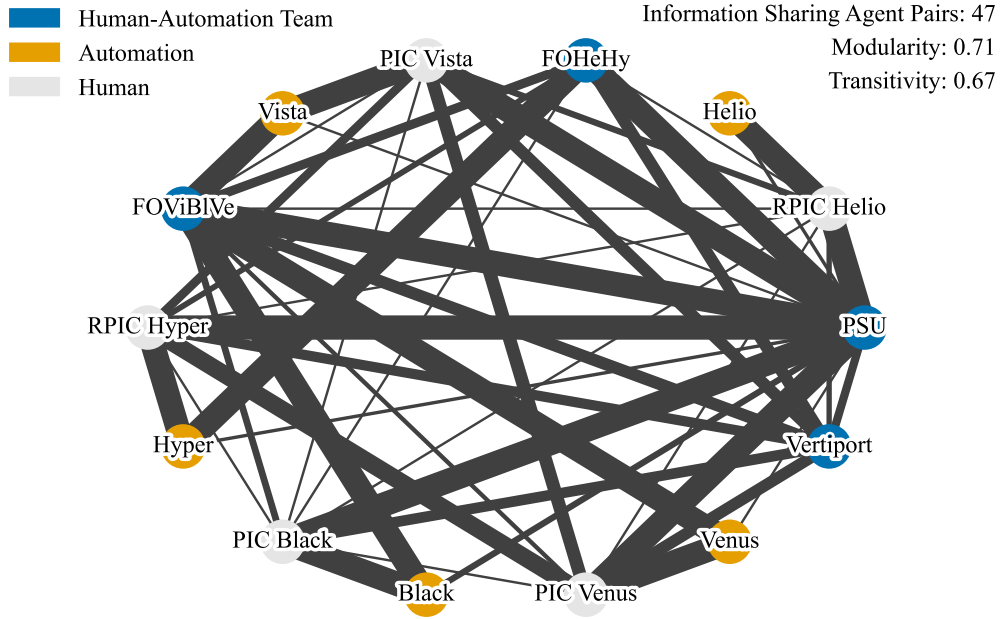


Fig. 7 Information exchange networks for simulations with wind change between Aircraft 1 and 2, including a 30s pilot communication lag and a 60s separation buffer.

568 in this condition they have access to the reason for the go-around (in digital form),
 569 they do not (yet) have access to updated weather information. They can therefore not
 570 know in what direction the wind changed, and what approach directions are still feasi-
 571 ble. The way in which the automated action of creating a new OI request is modeled,
 572 these automated services will return the original OI as still being valid.

573 As a result, even with the (R)PIC having already initiated a go-around, no new OI
 574 request is made by automated system governing the go-around aircraft. A new OI may
 575 not be proposed until *after* the nonconformance notification from the “Conformance
 576 Monitoring” action, followed by a request to revise their OI, which does not happen
 577 until several seconds later (the “Conformance Monitoring” check is assumed to occur
 578 every 10 seconds).

579 In contrast, *with* lag, a new OI is not proposed until 30 seconds after the go-
 580 around has been initiated (because it takes time for the *digital* communication of

581 the reason for the go-around and need for a new OI to reach automated systems).
582 Once automated systems then initiate a new OI request, wind reports have been
583 updated (they are assumed to update at most every 10 seconds) to reflect the new
584 wind direction. Automation then uses this information to recognize which approach
585 directions are available and successfully create a new OI.

586 This phenomenon affects both centralized and decentralized architectures similarly,
587 because both rely on wind updates to trigger the computation of a new proposed OI.
588 The key issue is not the pilot communication itself, but the fact that stale data, even
589 if not very old, leads to a mismatch between what the (R)PIC attempting to execute
590 the go-around observes and what the onboard automation believes is happening. In
591 centralized cases, this delay disrupts the PSU's OI updating strategy to account for the
592 aircraft that executed the go-around but did not propose a new OI. In decentralized
593 cases, the delay can lead pilots to not realize their new OI proposal was unsuccessful
594 until a request to revise their OI is received following a non-conformance notification,
595 sometimes making Lag 0 scenarios slower than expected due to unavailable airspace
596 slots, because by the time a new OI is proposed, other aircraft have already taken up
597 slots. The average completion time, after removing the outlier, is still higher at 1482s
598 than at 1446.6s for cases with pilot communication lag.

599 Since the PSU proposes new OIs in the centralized architecture, its proposed OIs
600 attempt to maintain the scheduled landing order. But under the decentralized archi-
601 tecture, this order is not automatically enforced, which can lead to race conditions,
602 where aircraft overtake each other to get the most optimized scheduling slots, and
603 potentially cause aircraft that do not execute a go-around to overtake those that do.
604 Race conditions occur because individual aircraft choose the most efficient routes,
605 increasing overall efficiency by reducing the last aircraft's landing time. However, this
606 compromises the minimum distance between aircraft in already busy corridors due
607 to overtaking. This results in (artificially) faster completion times, even though such

overtaking is not realistic: aircraft overtaking each other in narrow corridors represents a loss of separation and highly undesirable. In the decentralized architecture, 60% of simulations with lag result in violations of the landing order, whereas this number decreases to 40% without lag.

4.4 Patterns in (Mis-)Synchrony

To further understand the synchrony requirements for maintaining the sequencing of the aircraft, the interactions between communication lag and the wind report generation interval are simulated. The wind report generation interval ranges from 5 to 20 seconds. As maintaining synchrony is clearly the most challenging in the decentralized architecture, only the *decentralized* architecture is used in this set of simulations. This additional set of simulations revealed emergent behavior caused by interactions between the timing of wind changes, conformance monitoring, and the communication lag.

As discussed in the Lag 0s vs Lag 30s explanation for total completion time, there are two ways automated rerouting can be triggered: a pilot executing a go-around, or the conformance monitoring service flagging an aircraft as non-conforming and triggering all aircraft to re-assess the feasibility of their OIs. If the wind report generation interval exceeds the conformance monitoring interval, there is a high chance that the report of aircraft non-conformance due to the go-around will not trigger OI reassessment due to a lack of updated wind information caused by the automation miscoordination issue discussed previously. For example, if the wind change occurs at 371s, and the pilot notices it at 371.5s, and executes a go-around, and this triggers a non-conformance detection at 381s, it prompts the fleet operator to propose a new OI. However, at a wind reporting interval of 20s, the wind report is updated at 370 and 390s, respectively. Automation will therefore fail to reroute following both pilot detection of the wind shift at 371.5s and detection of aircraft non-conformance due to

the go-around at 381s. A dense (more aircraft) or spread-out (farther away) airspace helps recover from this, as later aircraft might perform a go-around and trigger another set of requests for new OI from the conformance monitoring service, again triggering automated rerouting for the aircraft for which it failed earlier.

Thus, when the wind report generation interval exceeds the conformance monitoring interval (10s), the system enters a state of non-conformance that is never resolved. Therefore, for further analysis of reconfiguration times, all simulations with a higher wind report generation interval (20s) than the conformance monitoring interval are excluded.

Figure 8 shows box plots for the reconfiguration time for different amounts of buffer and wind report generation frequency. Reconfiguration time is the time taken for all aircraft to negotiate new OIs after the unexpected wind change. Buffer values are presented in separate figures to demonstrate how synchrony issues arise differently depending on how much temporal spacing is required between aircraft. For every reconfiguration time and wind report generation frequency pair, there are two lag values, represented by separate box plots: one in blue and the other in yellow with forward-diagonal stripes. Each box plot in the graph represents five simulation runs; therefore, each buffer value has 30 simulation runs, and the overall figure represents 120 simulation runs. The graphs show outliers as round hollow markers. Most outliers occur when there is no pilot communication lag. Further analysis of the outliers and all simulations across combinations of wind report generation interval, pilot communication lag, and timing of the wind change revealed three patterns that are discussed in further detail below.

To illustrate and explain the patterns, Figures 9, 10, and 11 each depict a different simulation run in which one of these patterns occurs. Each figure shows the action traces from two different simulation runs, with all independent variables held constant while the communication lag is varied. The left side of the figures always shows the

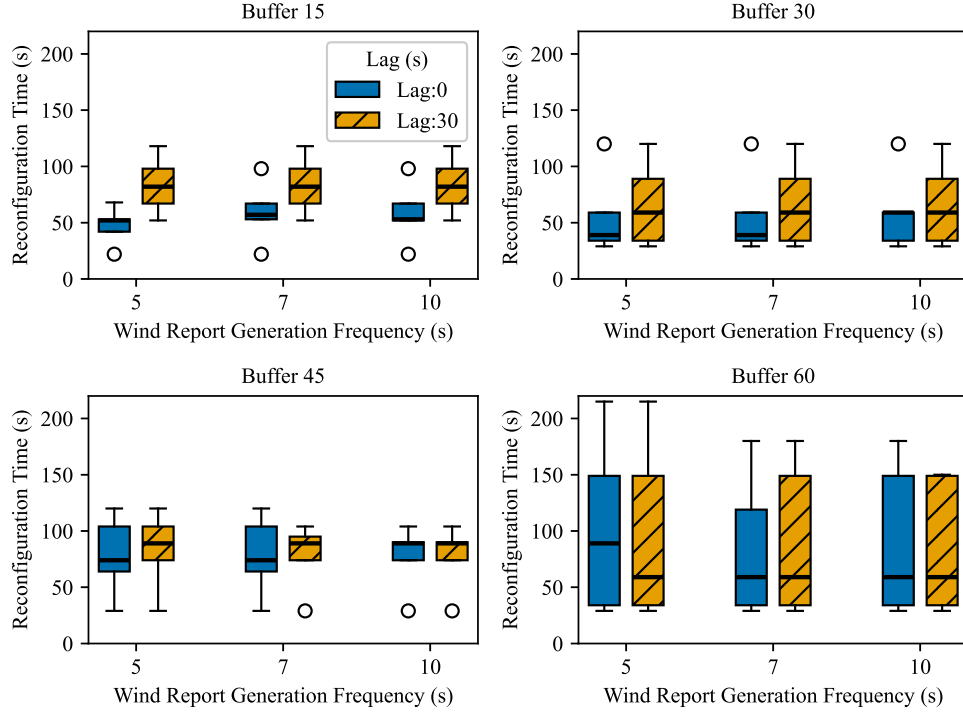


Fig. 8 Box and Whisker Plots - Reconfiguration Time vs Wind Report Generation Interval and Lag for each Buffer Value.

661 action trace from the simulation run *without* a communication lag, while the right side
 662 shows the simulation *with* a lag. A solid line indicates an action triggering or scheduling
 663 another action. A dashed line indicates scheduling of an action that eventually does not
 664 occur because another triggering pathway leads to an earlier execution of that action.
 665 An action that is scheduled but does not occur because another trigger executes it is
 666 indicated by a hollow circle. There are two notable exceptions to this representational
 667 logic: A solid line connected to a hollow circle on the far left sides of Figures 9 and 11
 668 indicates that the action is scheduled, but automated negotiation of a new OI does not
 669 complete due to the lack of an updated wind report. Table 4 outlines all simulation
 670 runs and marks those with patterns caused by the reconfiguration time, leaving the
 671 runs that do not have any difference in behavior due to a change in lag as empty cells.

Time between aircraft	15			30			45			60		
Wind report generation frequency	5	7	10	5	7	10	5	7	10	5	7	10
Before aircraft 1	1	1	1					3				3
Between 1-2	1			1	1	1					1	
Between 2-3	2	1	2	1	1		2	2		2		
Between 3-4	2											
Between 4-5	2	2	2									

Table 4 Patterns in Reconfiguration Times from the Simulation Output, illustrating three distinct patterns caused by communication lag

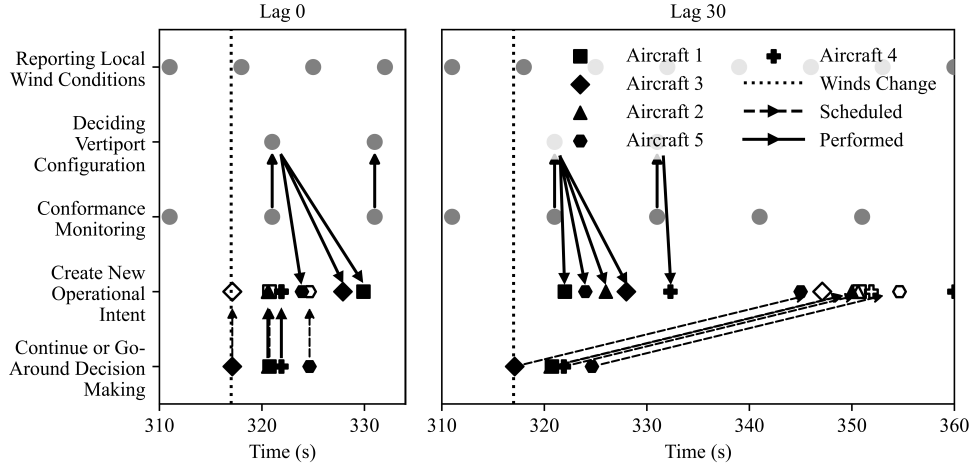


Fig. 9 Action Traces for Comparison of Reconfiguration Process between Lag 0 (Left) and Lag 30 (Right) for Pattern 1

1. **Pattern 1: When the weather report is updated with new wind information *after* the first aircraft executing its go-around and before system-level conformance monitoring, communication lag *increases* the reconfiguration time.** Figure 9 shows the difference in how the reconfiguration happens without and with communication lag. In this condition, the number of OIs being proposed by a system-level “decide direction of approach” triggered by the non-conformance notification reduces when there is no communication lag compared to when there is a communication lag. In the Lag 0s condition, some aircraft begin reconfiguration before receiving notice of a non-conforming aircraft, reducing

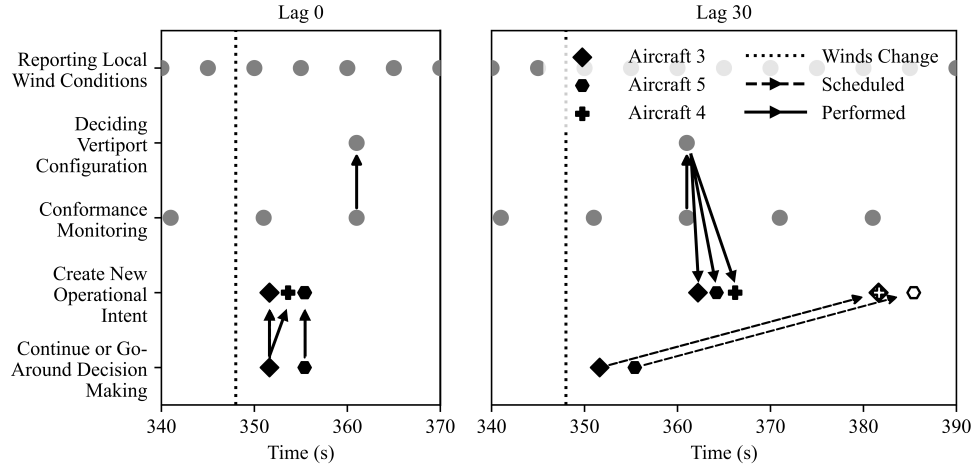


Fig. 10 Action Traces for Comparison of Reconfiguration Process between Lag 0 (Left) and Lag 30 (Right) for Pattern 2

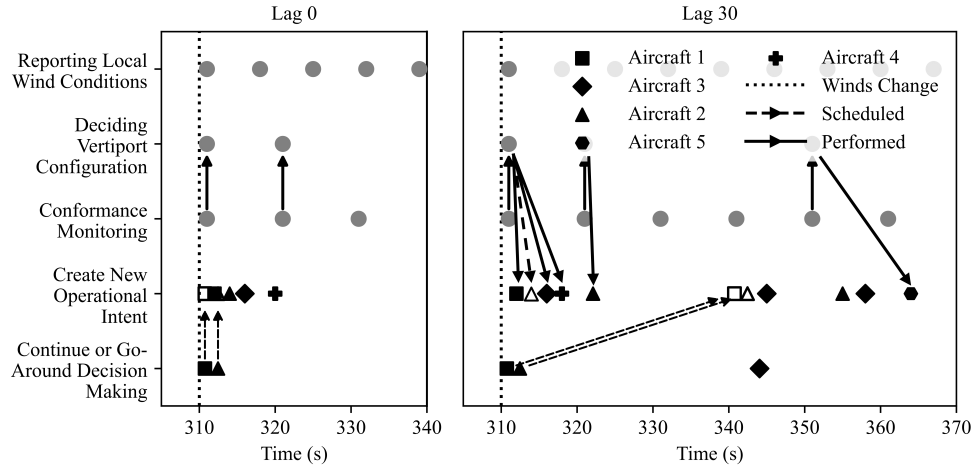


Fig. 11 Action Traces for Comparison of Reconfiguration Process between Lag 0 (Left) and Lag 30 (Right) for Pattern 3

request density and the likelihood of conflicts. This also starts the reconfiguration earlier, compared to the condition with a communication lag. The left side of Figure 9 shows the reconfiguration when there is no lag, showing that two reconfigurations are caused by aircraft executing a go-around, while the other three are initiated by

system-level actions. The right side of the figure 9 shows the reconfiguration when there is a 30 second lag, with all aircraft reconfiguration initiated by system-level actions. The simulation runs resulting in this pattern are marked as ‘1’ in Table 4.

2. **Pattern 2: When the weather report is updated with new wind information before any go-arounds, communication lag *increases* the reconfiguration time.** Receiving updated wind information before any go-arounds allows every aircraft that executes a go-around to start creating a new proposed OI, avoiding the miscoordination caused by automation acting on stale information as depicted on the right side of Figure 10. In the lag 30 condition, new proposed OIs are triggered by a system-level action, similar to pattern 1, as represented on the right side of Figure 10. This pattern of alignment results in one of the fastest reconfiguration times across all independent variable combinations. The simulation runs resulting in this pattern are marked as ‘2’ in Table 4.

3. **Pattern 3: When the wind report is updated before the FO proposes new OI due to a report of a nonconforming aircraft and before the pilot proposes a new OI due to the go-around, communication lag *decreases* the reconfiguration time.** This occurs because an aircraft executes a go-around after the previous aircraft is reported nonconforming but before the FO’s new proposed OI for the aircraft is confirmed. In the Lag 0 condition, the go-around triggers earlier OI creation, whereas in the Lag 30 condition, OI creation is delayed by the communication lag. This creates different ordering for proposing new OIs for the different aircraft due to the pilot communication lag. When the lag is 0, the go-around aircraft proposes a new OI early, whereas when the lag is 30, the go-around aircraft becomes one of the last aircraft to propose a new OI. This change in the order of OI creation in the lag 0 condition sometimes causes an additional aircraft to go-around due to the negotiation conflicts. In lag 0, the go-around aircraft OI gets approved first and now conflicts with the new OI proposed by the next

aircraft, causing an additional go-around. Figure 11 shows this change in order in both cases. The lag 0 condition, left side of the figure, shows how the go-around aircraft (represented by a triangle) creates a new OI before aircraft 4 (represented by a plus). Whereas in the lag 30 condition, right side of the figure, this order is reversed, giving non-go-around aircraft an earlier opportunity to propose a new OI as represented in the right side of Figure 11, where aircraft 4 (plus sign) creates new OI before aircraft 2 (triangle), avoiding the additional need for go-arounds. The simulation runs resulting in this pattern are marked as ‘3’ in Table 4.

While the above three patterns provide a rationale for how the communication lag affects reconfiguration time, they also explain the shifts in the median values, leading to outliers in the box plots. For example, the three outliers that have a lower reconfiguration time for a between-aircraft buffer of 15 seconds are caused by Pattern 2. The higher outliers are due to a median shift caused by changes in reconfiguration time across all patterns, making them outliers when there is no pilot communication lag.

5 Discussion and Conclusion

Work modeling and simulation can uncover the complexities of distributing decision-making control among different agents and explore the impacts of design decisions related to work architecture. Centralized and decentralized architectures offer divergent pathways for achieving system synchronization. A centralized architecture delivers better system performance metrics, including faster overall completion times, fewer OI requests, and lower synchronization costs. In contrast, a decentralized architecture creates a highly interconnected network that distributes the synchronization burden across the entire agent collective.

736 The information exchange requirements revealed that the PSU is central to both
737 centralized and decentralized architectures. This suggests that the PSU remains heav-
738 ily involved in activities in either architecture, with the nature of its tasks shifting
739 between active decision-making, monitoring, and extensive information sharing. In
740 both architectures, the information exchange requirements for the PSU remain very
741 high relative to other agents, indicating the need for careful design of the joint cognitive
742 system of the PSU, including the human role, along with automation that can scale
743 as needed to meet task demands. Further research is needed to create strategies that
744 distribute the information requirements effectively over time. Such strategies could
745 enhance the system’s adaptive capacity and better prepare it to handle unforeseen
746 conditions. These system architectures are not static and can therefore be dynami-
747 cally changed to support agents in these situations by incorporating mechanisms that
748 redirect some of the information exchange requirements to other agents in the system.
749 The design should be prepared to account for dynamic changes in architecture and to
750 support the additional interactions and synchronization required to support dynamic
751 task allocation.

752 The results of the Lag 0 and Lag 30 conditions challenge the implicit assump-
753 tion that instantaneous communication is always superior. In the Lag 30 condition,
754 the “communication delay” inadvertently provides the system with the temporal slack
755 necessary for important information—such as wind reports—to be updated and dis-
756 tributed across the system before key decisions are made. This underscores a vital
757 principle for cooperative traffic management: synchronization is not instantaneous
758 and requires agents to account for the time-costs of information propagation. Taking
759 “decisive action” too early, as seen in the Lag 0 condition, actually creates asynchrony
760 and potentially cascading coordination demands. While a static fix might involve pro-
761 gramming the automation to simply request a data refresh, a more robust, adaptive
762 solution requires introducing mechanisms that allow agents to monitor the state of

763 information sharing itself. True synchronization requires not only the timely sharing
764 of relevant information but also explicit verification that all agents are operating on a
765 consistent, up-to-date version of the truth before executing interdependent work.

766 The additional simulations, which varied the wind report generation frequency,
767 provided further insight into the synchrony requirements. The interactions between
768 the lag and the wind report generation frequency show that certain synchronization
769 strategies yield faster reconfiguration times for the system under the lag 0 condition,
770 as in patterns 1 and 2, whereas others yield faster reconfiguration times for the lag 30
771 condition, as in pattern 3. The analysis also revealed an important synchronization
772 requirement: the wind report generation frequency should be lower than the confor-
773 mance monitoring frequency, especially in a fully automated system. The absence of
774 such a requirement can cause system coordination breakdowns and result in aircraft
775 flying without any approved OI. This shows an important avenue for further exploring
776 how different timing issues might lead to emergent properties.

777 Negotiation conflicts and race conditions in OI requests are also highlighted in
778 the simulation results. Negotiation conflicts arise from unavailability due to uncer-
779 tainty about when original resource reservations should be released after a new OI
780 is proposed that creates additional reservations. Several options were considered, like
781 releasing reservations immediately after a new OI is proposed or waiting until the
782 proposed OI is confirmed by the PSU. The first option risks aircraft flying without a
783 valid OI, while the second could unnecessarily reserve resources, increasing the likeli-
784 hood of negotiation conflicts. Another potential solution is to make the requested OI
785 public, but operators may not be inclined to share it with competitors before the OI
786 is confirmed. Another issue with this is confusion about which slots an aircraft should
787 consider when creating new OI reservations. In general, when multiple aircraft need
788 to negotiate a new OI simultaneously, it may be difficult to know which resources will

789 be available and when, suggesting a need for complex negotiation approaches rather
790 than a first-come, first-served approach.

791 The race conditions, on the other hand, occur because when aircraft execute go-
792 arounds, the physical sequence is not necessarily preserved in the updated set of OIs
793 for reconfiguring the vertiport and reattempting the landing. Such resequencing could
794 occur in any situation that calls for replanning, and in today's environment, it is
795 often managed by humans, who have the authority to override the sequence proposed
796 by automation. In a decentralized highly automated cooperative environment, who
797 will be responsible for maintaining aircraft sequencing during airborne replanning?
798 For example, if automation does not maintain the physical aircraft sequence when
799 generating new OIs for go-arounds, who is responsible for correcting and maintaining
800 the physical sequencing? In the simulation scenario, the aircraft were crewed, allowing
801 the pilots to self-organize as they would at an uncontrolled airport today, but the
802 example highlights a potential requirement for a resource scheduling services in future
803 architectures.

804 Information synchronization is critical for effective and efficient tactical replan-
805 ning. Results from the simulation showed that, irrespective of the architecture used,
806 a robust architecture for information exchange is required. The information require-
807 ments emphasize the need to build coordination mechanisms that enable agents to
808 synchronize their activities with other agents they may not interact with under nom-
809 inal conditions, but are required for successfully replanning operations in off-nominal
810 situations. Not only should the coordination mechanisms be available, but it may
811 also be good to test them periodically, so agents are primed to use them effectively
812 when needed. Network metrics can be used to understand the impacts of different
813 architectures, which is inherently incomplete without simulations, both human and

814 computational. The transitivity and modularity values show how the proposed archi-
815 tectures' information requirements dictate the system structure, thereby aiding in
816 validating and verifying the assumptions made during initial system design.

817 **5.1 Summary of Coordination Requirements from Trade**

818 **Space Analysis**

819 The analysis of work dynamics revealed the complexity of the distribution of control
820 and the resulting challenge of synchronizing threads of activity among the multiple
821 agents. It showed clear differences in synchrony challenges based on how control is
822 distributed and how long it takes to disseminate key information to relevant actors.
823 In summary, for maintaining synchrony during reconfiguration, system design must
824 support the following coordination requirements:

- 825 • Before creating a new OI request, automated rerouting services should wait for an
826 updated weather report.
- 827 • Weather reports must be provided frequently enough for automated rerouting ser-
828 vices to provide timely new OI requests (and minimize the time aircraft are in
829 non-conformance).
- 830 • To reduce the likelihood of automation surprises, automated rerouting services
831 should provide feedback to pilots about the status of a new OI request.
- 832 • In a decentralized system, the agents in the system should synchronize to maintain
833 a pseudo sequence of aircraft's landing to avoid potentially dangerous race condi-
834 tions. In the centralized condition, the PSU's resource scheduling service should also
835 consider race conditions before rescheduling.
- 836 • The conformance monitoring service should wait for an updated weather report
837 before starting a new OI request.
- 838 • Pilots or FO requesting automated rerouting services to create a new OI request
839 must explicitly verify that the information is synchronized.

5.2 Role of Simulation in Discovering Potential

Synchronization Issues Early in Design

Computational modeling and simulation is a well-established tool that engineers and designers can use to understand the trade-offs associated with design decisions. This paper demonstrates the use of modeling and simulation of work for uncovering synchronization requirements while supporting continuous envisioning of new concepts of operations. While our experiments effectively compare high-level requirement variations across architectures, this model serves as a foundational first step in uncovering areas of concern. To support iterative envisioning, future iterations can adapt model fidelity as the system design matures. The aspects of the model to prioritize for refinement depend on the nature of the questions to be addressed. For instance, modeling negotiation architectures in more detail to analyze race conditions, or detailing more distributed architectures to explore more intricate changes in decision-making control.

Work-based modeling and simulation provide a discovery framework for identifying synchronization issues, helping streamline future research. This approach allows designers and engineers to prioritize research on communication lag, which might otherwise remain obscured by the assumption that instantaneous communication is superior. The results provide the grounding required to progress through the envisioned world problem using simulation. The specific synchronization failures uncovered should be further validated through iteratively higher fidelity human-in-the-loop simulation studies, followed by the design of a mechanism to mitigate them. Ultimately, by generating the quantitative data needed to inform technical trade-offs, work modeling fulfills a core mission of cognitive systems engineering: influencing the system design decisions [24, 31].

Expanding on the models established by [11], this study demonstrates how work-based modeling and simulation support continuous envisioning by enabling rapid reconfiguration and reuse. While the initial investment in creating computational work

models is significant, the ability to reuse core components—such as flight dynamics—while only reconfiguring contingency-specific functions demonstrates a sustainable path for iterative design. In this study, minimal changes to the flight models were required to explore the complex requirements of aerodrome reconfiguration, illustrating how computational work models can complement and scale with system updates. This reuse supports an iterative discovery process, uncovering the research areas most critical to building resilient, cooperative air traffic management systems.

Acknowledgements. This work was performed under the National Aeronautics and Space Administration (NASA) contract number 80NSSC23CA121, with Jason Prince serving as NASA technical monitor. The authors greatly appreciate the input and feedback from the NASA team. The authors would also like to thank the Mosaic ATM team members and other WMC developers.

Conflict of interest. The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding. The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was performed under NASA contract number 80NSSC23CA121.

Author contributions. Conceptualization, P.A., I.M., and F.A.; computational modeling and simulation, P.A.; data analysis and graph generation, P.A.; writing—original draft, P.A. and I.M.; writing—review and editing, P.A., I.M., and F.A.. All authors have read and agreed to the published version of the manuscript.

Declarations

- Ethics approval and consent to participate: Not applicable
- Data availability: The simulation data that support the findings of this study are available, but restrictions apply to the availability of these data, and so are not

publicly available. The data are, however, available upon request and with the permission of the collaborating institutions.

- Materials availability: Not applicable
- Code availability: The code that supports the findings of this study is available, but restrictions apply to the availability of these data, and so are not publicly available. The code, however, is available upon request, subject to the permission of the collaborating institutions.

References

- [1] Deloitte: National Aeronautics and Space Administration (NASA) UAM Vision Concept of Operations (ConOps) UAM Maturity Level (UML) 4. Technical report, Deloitte (2020). <https://ntrs.nasa.gov/api/citations/20205011091/downloads/UAM%20Vision%20Concept%20of%20Operations%20UML-4%20v1.0.pdf>
- [2] Federal Aviation Administration: Unmanned Aircraft System (UAS) Traffic Management (UTM). Technical report, FAA (2020). https://www.faa.gov/sites/faa.gov/files/2022-08/UTM.ConOps_v2.pdf
- [3] (FAA), F.A.A.: Urban Air Mobility Concept of Operations v2.0. FAA Technical Report v2.0, Federal Aviation Administration (2023). <https://www.faa.gov/sites/faa.gov/files/Urban%20Air%20Mobility%20%28UAM%29%20Concept%20of%20Operations%202.0.0.pdf>
- [4] Wisk Aero, The Boeing Company, SkyGrid: Concept of Operations for Automated Flight Rules (AFR). Technical report, Wisk Aero, Boeing, and SkyGrid (December 2025). https://cdn.bfldr.com/3FMO7QHS/as/3j55gvks8t92r5fv7jm2f5b/Wisk_Boeing_and_SkyGrid_ConOps

- 916 [5] Verma, S., Dulchinos, V., Wood, R.D., Farrahi, A., Mogford, R., Shyr, M., Ghatas,
917 R.: Design and analysis of corridors for uam operations. In: 2022 IEEE/AIAA
918 41st Digital Avionics Systems Conference (DASC), pp. 1–10 (2022). [https://doi.
919 org/10.1109/DASC55683.2022.9925820](https://doi.org/10.1109/DASC55683.2022.9925820)
- 920 [6] Chow, R., Christoffersen, K., Woods, D.D.: A model of communication in sup-
921 port of distributed anomaly response and replanning. Proceedings of the Human
922 Factors and Ergonomics Society Annual Meeting **44**(1), 34–37 (2000) [https:
923 //doi.org/10.1177/154193120004400110](https://doi.org/10.1177/154193120004400110)
- 924 [7] Sarter, N.B., Woods, D.D., Billings, C.E.: Automation surprises. Handbook of
925 Human Factors and Ergonomics **2**, 1926–1943 (1997)
- 926 [8] Woods, D.D.: The theory of graceful extensibility: basic rules that govern adaptive
927 systems. Environment Systems and Decisions **38**(4), 433–457 (2018) [https://doi.
928 org/10.1007/s10669-018-9708-3](https://doi.org/10.1007/s10669-018-9708-3)
- 929 [9] Branlat, M., Woods, D.: How do systems manage their adaptive capacity to suc-
930 cessfully handle disruptions? a resilience engineering perspective. In: AAAI Fall
931 Symposium - Technical Report, pp. 26–34 (2010)
- 932 [10] Woods, D.D., Sarter, N.B.: Capturing the dynamics of attention control from
933 individual to distributed systems: the shape of models to come. Theoreti-
934 cal Issues in Ergonomics Science **11**(1-2), 7–28 (2010) [https://doi.org/10.1080/
935 14639220903009896](https://doi.org/10.1080/14639220903009896)
- 936 [11] Paladugu, A., Fernandes, A., IJtsma, M.: Computational simulation of distributed
937 work as a discovery tool for envisioning future operations. Journal of Cognitive
938 Engineering and Decision Making **19**(4), 395–419 (2025) [https://doi.org/10.1177/
939 15553434251327698](https://doi.org/10.1177/15553434251327698)

- [12] Klein, G., Feltovich, P.J., Bradshaw, J.M., Woods, D.D.: Common Ground and Coordination in Joint Activity. In: Rouse, W.B., Boff, K.R. (eds.) Organizational Simulation. Wiley series in systems engineering and mangement. Wily-Interscience, Hoboken, NJ (2005). <https://doi.org/10.1002/0471739448.ch6>
- [13] Malone, T.W., Crowston, K.: The interdisciplinary study of coordination. ACM Computing Surveys **26**(1), 87–119 (1994) <https://doi.org/10.1145/174666.174668>
- [14] Maguire, L.M.D.: Managing the Hidden Costs of Coordination. Communications of the ACM **63**(4), 90–96 (2020) <https://doi.org/10.1145/3379989>
- [15] Smith, P.J., Caisse, S., Beck, C., Denning, R., Obradovich, J.H., McCoy, C.E., Orasanu, J.: Using critical incidents to understand the interactions of airline dispatchers with the traffic management system. In: Proceedings Fourth Annual Symposium on Human Interaction with Complex Systems, pp. 48–62 (1998). <https://doi.org/10.1109/HUICS.1998.659954>
- [16] Pritchett, A.R., So Young Kim, Kannan, S.K., Feigh, K.: Simulating situated work. In: 2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), pp. 66–73. IEEE, Miami Beach, FL, USA (2011). <https://doi.org/10.1109/COGSIMA.2011.5753756>
- [17] IJtsma, M.: Computational Simulation of Adaptation of Work Strategies in Human-Robot Teams. Dissertation, Georgia Institute of Technology, Atlanta, GA (2019)
- [18] International Council on Systems Engineering: Systems Engineering Handbook: a Guide for System Life Cycle Processes and Activities, Fourth edition edn. Wiley,

- 964 Hoboken, N.J (2015)
- 965 [19] Frittman, J., Edson, R.: A systems thinking based approach to writing effective
966 concepts of operations (conops) (2010)
- 967 [20] Miller, M.J., Feigh, K.M.: Addressing the envisioned world problem: a case study
968 in human spaceflight operations. *Design Science* **5**, 3 (2019) [https://doi.org/10.
969 1017/dsj.2019.2](https://doi.org/10.1017/dsj.2019.2)
- 970 [21] Woods, D.D., Dekker, S.: Anticipating the effects of technological change: A new
971 era of dynamics for human factors. *Theoretical Issues in Ergonomics Science* **1**(3),
972 272–282 (2000) <https://doi.org/10.1080/14639220110037452>
- 973 [22] IJtsma, M., Ma, L.M., Pritchett, A.R., Feigh, K.M.: Computational Methodology
974 for the Allocation of Work and Interaction in Human-Robot Teams. *Journal of
975 Cognitive Engineering and Decision Making* **13**(4), 221–241 (2019) [https://doi.
976 org/10.1177/1555343419869484](https://doi.org/10.1177/1555343419869484)
- 977 [23] Kossiakoff, A., Biemer, S.M., Seymour, S.J., Flanigan, D.A.: *Systems Engineer-
978 ing: Principles and Practice*, 3rd edn. Wiley series in systems engineering and
979 management. John Wiley & Sons, Inc, Hoboken, NJ (2020). [https://doi.org/10.
980 1002/9781119516699](https://doi.org/10.1002/9781119516699)
- 981 [24] Smith, P.J.: In: Smith, P.J., Hoffman, R.R. (eds.) *Making Brittle Technologies
982 Useful*, pp. 181–208. CRC Press, Boca Raton, FL (2017). Chap. 10. [https://doi.
983 org/10.1201/9781315572529-10](https://doi.org/10.1201/9781315572529-10)
- 984 [25] Woods, D.D., Hollnagel, E.: *Joint Cognitive Systems: Patterns in Cognitive Sys-
985 tems Engineering*, 1st edn. CRC. CRC Press, Boca Raton, FL (2006). [https:
986 //doi.org/10.1201/9781420005684](https://doi.org/10.1201/9781420005684)

- 987 [26] Feltovich, P.J., Hoffman, R.R., Woods, D., Roesler, A.: Keeping it too simple: how
 988 the reductive tendency affects cognitive engineering. *IEEE Intelligent Systems*
 989 **19**(3), 90–94 (2004) <https://doi.org/10.1109/MIS.2004.14>
- 990 [27] Gorman, J.C., Cooke, N.J., Winner, J.L.: Measuring team situation awareness in
 991 decentralized command and control environments. *Ergonomics* **49**(12-13), 1312–
 992 1325 (2006) <https://doi.org/10.1080/00140130600612788> . PMID: 17008258
- 993 [28] NUAIR: High Density Automated Vertiport Concept of Operations. Techni-
 994 cal report, NUAIR, Rome, NY (2021). [https://ntrs.nasa.gov/api/citations/
 995 20210016168/downloads/20210016168_MJohnson_VertiportAtmtnConOpsRprt_
 996 final_corrected.pdf](https://ntrs.nasa.gov/api/citations/20210016168/downloads/20210016168_MJohnson_VertiportAtmtnConOpsRprt_final_corrected.pdf)
- 997 [29] Boeing Wisk: Concept of Operations for Uncrewed Urban Air Mobility.
 998 Technical report, The Boeing Company: Arlington, VA, USA, 2022 (2023).
 999 [https://www.boeing.com/content/dam/boeing/boeingdotcom/innovation/
 1000 con-ops/docs/Concept-of-Operations-for-Uncrewed-Urban-Air-Mobility.pdf](https://www.boeing.com/content/dam/boeing/boeingdotcom/innovation/con-ops/docs/Concept-of-Operations-for-Uncrewed-Urban-Air-Mobility.pdf)
- 1001 [30] Cheng, A.W., Witzberger, K.E., Isaacson, D.R., Verma, S.A., Arneson, H.M., Lee,
 1002 H., Seah, C.H., Field, M., Levitt, I.: National Campaign (NC)-1 Strategic Conflict
 1003 Management Simulation (X4) Final Report. Technical report, National Aero-
 1004 nautics and Space Administration (2022). [https://ntrs.nasa.gov/api/citations/
 1005 20220018159/downloads/X4-Final-Report-V8.pdf](https://ntrs.nasa.gov/api/citations/20220018159/downloads/X4-Final-Report-V8.pdf)
- 1006 [31] Woods, D.D., Roth, E.M.: Cognitive engineering: Human problem solving
 1007 with tools. *Human Factors* **30**(4), 415–430 (1988) [https://doi.org/10.1177/
 1008 001872088803000404](https://doi.org/10.1177/001872088803000404)