Computers & Industrial Engineering 137 (2019) 106041

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

The design and simulation of an autonomous system for aircraft maintenance scheduling

Yinling Liu[a,*], Tao Wang[b], Haiqing Zhang[c], Vincent Cheutet[a], Guohua Shen[d]

[a] Université de Lyon, INSA Lyon, Laboratoire DISP (EA4570), France
[b] Université de Lyon, Université Jean Monnet, Laboratoire DISP (EA4570), France
[c] School of Software Engineering, Chengdu University Information Technology, China
[d] School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China

ARTICLE INFO

Keywords:
Maintenance system
Systems Engineering
Architecture design
Agent-based modelling and simulation

ABSTRACT

Operational support is a key issue for aircraft maintenance, which aims to improve operational efficiency and reduce operating costs under the premise of ensuring flight safety. Although many works have emerged to achieve this aim, they mostly address the concept of maintenance systems, the relationship between stakeholders and the loop of maintenance information separately. Hence, the cooperation between stakeholders could be impeded especially when urgent decisions should be made, relying on historical data and real-time data. In this paper, we propose an innovative design of an autonomous system supporting the automatic decision-making for maintenance scheduling. The design starts from the proposition of the analysis framework, to concept formulation of the system, to information transitional level interface, and ends with an instance of system module interactions. The underlying architecture illustrates the high-level fusion of technical and business drives; optimizes strategies and plans with regard to maintenance costs, service level and reliability. An agent-based simulation system is developed as a proof to illustrate the feasibility of the system principle and algorithms. Furthermore, the simulation experiment analyzing the impact of maintenance sequence strategies on maintenance costs and service level has demonstrated the algorithm functionality and the feasibility of the proposed approach.

1. Introduction

Maintenance businesses for aircraft involve flight scheduling, maintenance strategies and planning, repair, part supply and a number of stakeholders including: OEM (Original Equipment Manufacturing), suppliers, airline, MRO (Maintenance, Repair and Overhaul), and airworthiness authority, etc. Stakeholders cooperate with each other in a distributed, synchronized and ephemeral way. For example, suppliers independently purchase parts when they are out of stock. When necessary parts and auxiliary equipment for repairing some faulty components are ready, MRO operations will be scheduled. Maintenance managers will define strategies as soon as maintenance is requested. The major difference between general plant and machinery maintenance, and aircraft maintenance is that the latter is mandated and monitored by regulatory authorities like FAA (Federal Aviation Administration), CAA (Civil Aeronautics Authority), etc. (Sahay, 2012). Thus, they are the individuality of stakeholders, the complex maintenance regulations, and distributed, synchronized and ephemeral cooperation between stakeholders that clearly make the system on the operational support for aircraft maintenance rather complex.

Recently, aircraft maintenance still largely depends on the economical time-based PM (Preventive Maintenance) practice and the manual analysis (Sahay, 2012). The deterioration of components and unplanned maintenance activities make the maintenance further complex. PM is programmed to prevent failures on aircraft from taking place, in order to minimize air transport service disruption. Less effective CM (Corrective Maintenance) is easier to cause “chain reaction” in the network of air transport service. Additionally, delayed flights, incidents, broken components, cost, requested parts and safety recommendations, etc., are generated every day. Clearly, the operational support for aircraft maintenance is becoming a data-rich issue with a considerable number of data streams and operating databases. So, the real challenge is about how to provide efficient operational support for aircraft maintenance under this complex environment.

* Corresponding author.
E-mail addresses: yinling.liu@insa-lyon.fr (Y. Liu), tao.wang@univ-st-etienne.fr (T. Wang), haiqing.zhang.zhq@gmail.com (H. Zhang), vincent.cheutet@insa-lyon.fr (V. Cheutet), ghshen@nuaa.edu.cn (G. Shen).

https://doi.org/10.1016/j.cie.2019.106041
Received 26 September 2018; Received in revised form 17 June 2019; Accepted 2 September 2019
Available online 06 September 2019

Many works have emerged to tackle this issue. They mostly address the concept of maintenance systems (Duffuaa, Ben-Daya, Al-Sultan, & Andidjani, 2001; Durazo-Cardenas et al., 2018), the relationship between stakeholders (Ward, McDonald, Morrison, Gaynor, & Nugent, 2010) and the loop of maintenance information (MacKenzie, Miller, & Hill, 2012; Zhang, Liu, Jiang, & Chen, 2015). However, to the best of our knowledge, no one combines these three aspects together, in order to build a new autonomous maintenance system. This may give rise to some deficiencies to the system. Developing a system without its architecture design makes it difficult to truly understand its essence and key properties, which in turn affects concerns such as the feasibility, utility, completeness, and maintainability of the system (ISO, 2011). Designing an architecture model of systems without an executable system (or simulation system) makes it hard to demonstrate the feasibility of the architecture design. Therefore, an innovative methodology is highly demanded to build such a system starting from the requirement analysis, to the architecture design, to the analysis of system modules involving stakeholders, and ending with a concrete simulation system. As a result, this system is capable of more efficiently transforming the historic and real-time data into global decisions for maintenance scheduling, which eventually effectively decreases the maintenance cost and improves maintenance efficiency.

SE (Systems Engineering) is an interdisciplinary field of engineering and engineering management that focuses on how to design and manage complex systems over their life cycles (Blanchard, 2004). The *waterfall approach* is a common approach to develop complex systems, which uses a sequence of design, implementation, testing, and evaluation (Royce, 1987). It enables us to build large systems by decomposing them into small, manageable, and testable units (Waltz & Hall, 2001). Therefore, SE is a good choice for building the system, in order to have a holistic point of view on the system at the design phase.

Implementing a real autonomous system of aircraft maintenance is definitely difficult because a number of information systems are hardly accessible and the evidence for convincing the operators to follow system instructions is rather limited. However, simulation systems enable us to gain important insights into future systems in an inexpensive way, especially when the costs, risks or logistics of manipulating the real system-of-interest are prohibitive.

ABMS (Agent-Based Modelling and Simulation) is a relatively new approach to modelling the dynamics of complex systems and complex adaptive systems (Macal & North, 2005). It is generally employed when the complexity of the system being modelled is beyond what static models or other techniques can fully present (Helbing, 2012). The complexity of the autonomous system of aircraft maintenance mainly lies in system uncertainty and dynamics. The uncertainty is reflected not only by random mechanisms but by unknown behaviours resulting from agents' interactions. For example, the occurrence of faults on components of aircraft is conformed to a certain probability. The bounds on these uncertainty issues and the implications of potential outcomes should be understood and evaluated. Agent-based simulations provide a flexible and useful mechanism to capture these uncertainties (Heppenstall, Crooks, See, & Batty, 2011; Monostori, Váncza, & Kumara, 2006). In terms of system dynamics, maintenance tasks change over time, including maintenance sequences for aircraft and maintenance resources scheduling. Different environments will lead to different system behaviours. Agent-based decentralization takes this into account by letting each agent continuously coordinate its actions with other agents, instead of making this agent apply a behaviour prescribed at design-time (Moyaux, Chaib-Draa, & D'Amours, 2006). Additionally, ABMS allows us to model real-world systems of interest in ways that beyond the capabilities of traditional modeling techniques, such as discrete event system or system dynamics (Ali, Doolan, Wernick, & Wakelam, 2018; Siebers, Macal, Garnett, Buxton, & Pidd, 2010). Hence, ABMS is a better fit for realizing the autonomous system design.

Furthermore, a framework for building an autonomous system is firstly provided, in order to explain how to combine SE and ABMS. Based on this framework, a requirement analysis, an autonomous system design, and a simulation system will then be accomplished successively. In the end, simulation experiments will be performed to validate the feasibility of the proposed approach. The main contributions of this paper are illustrated as follows:

- Proposing a new framework that fully supports the building of an autonomous system on the operational support for aircraft maintenance;
- Proposing an architecture model for the autonomous system based on which the detailed development of components is achieved and a UML (Unified Model Language) use case diagram involving stakeholders is provided;
- Developing a concrete agent-based simulation system to enable us to have a more detailed description of maintenance strategy and scheduling, a more in-depth analysis of service level and cost, and a loop of more complete maintenance information.

Based on the simulation system, maintenance engineers can run different maintenance scenarios to investigate the impact of key factors on maintenance costs and service level. For example, the impact of the number and the distribution of maintenance technicians on maintenance efficiency can be analyzed. Predicting maintenance issues is also possible based on part of real data and the data generated from simulation experiments. Finally, the virtualization of the maintenance process allows us to have an insight into any maintenance details.

The remainder of this paper is structured as follows. Section 2 reviews the main related works. Section 3 represents the whole design process for the autonomous system of aircraft maintenance. Section 4 builds an agent-based simulation system based on the system design. Section 5 conducts a simulation experiment on the impact of the maintenance sequence strategy on maintenance costs and service level. Section 6 concludes the paper with future perspectives.

2. Related researches and limitations

This section provides a comprehensive literature review on modelling maintenance systems. Since this paper focuses on the design of the autonomous maintenance system to improve the operational support for aircraft maintenance, it excludes the papers without a system's perspective. For example, the topics like utilizing mathematical approaches (DeBruecker, Beliën, Van den Bergh, & Demeulemeester, 2018; Keysan, Nemhauser, & Savelsbergh, 2010) or only introducing algorithms (Ahmed & Azadian, 2017; Diaz, Huertas, & Trigos, 2014) to improve maintenance scheduling and planning are out of our scope. The synthesis analysis of reviewed papers is concluded in Table 1. The papers are analyzed from six aspects: proposes conceptual models, proposes simulation models, involved stakeholders, maintenance-related information, model purposes and the relevance to our research. Furthermore, the relevance is evaluated with different levels: low, medium and high.

According to this table, researchers rarely propose both conceptual models and simulation models at the same time. Fortunately, Chang, Ni, Bandyopadhyay, Biller, and Xiao (2007) gather these two models together. Nevertheless, they only concentrate on the issue of maintenance labours. The maintenance process is rather simple. As for involved stakeholders, researchers have almost neglected this aspect. Although Durazo-Cardenas et al. (2018) have involved stakeholders, they do not provide a simulation model. The majority of maintenance information has been involved by researchers. Most of the reviewed papers consider parts states, resource availability, planning and scheduling information and labours, etc., but the planning and scheduling information is just associated with maintenance activities. The scheduling information for the system-of-interest is still missing. In this paper, it is the information about the scheduling of aircraft. The lack of this information will bring

Table 1
An overview of related papers – A (proposes conceptual models), B (proposes simulation models), and C(involved stakeholders).

Article	A	B	C	Maintenance-related info.	Model purposes	Relevance to our research
1. Durazo-Cardenas et al. (2018)	Yes	No	Y-es	Parts degradation state, planning and scheduling info., cost info.	Autonomous scheduling maintenance jobs	Medium, similar model purposes, similar aspects of architecture designs, no simulation system is involved
2. Gary et al. (2018)	No	Yes	N-o	Equip. status, labours, maint. type, planning and scheduling info., stock, maint. performance, maint. cost	Presents a system dynamic model for the study on dynamic behaviours of the maintenance performance and costs	Medium, similar in scope, explicit model, only enables an aggregated level analysis
3. Sahnoun et al. (2015)	No	Yes	N-o	Main. duration, resource availability, degradation level, weather condition, manpower	Develops a multi-agent system for selecting the optimal maint. strategy for offshore wind turbines	Medium, similar in scope, explicit model, simple maintenance strategy, overly simple algorithm for maint. scheduling
4. Zhang et al. (2015)	Yes	No	N-o	Material, parts, knowledge, labours, equipment, failures,	Supports the continuous improvement of productivity and reduces the MRO cost	Low, the proposed framework is too simple. The model description is quite abstract. No strategies or scheduling is involved
5. Sama and Kiridena (2012)	Yes	No	N-o	Material, activities, resources, suppliers, labours	Develops an integrated framework for the planning and scheduling of aircraft heavy maint.	Low, similar objectives, a detailed discussion about planning and scheduling of aircraft heavy maint., only part of maint. processes are involved
6. MacKenzie et al. (2012)	No	Yes	N-o	Average sortie duration, the number of aircraft, manpower, break rates, abort rates, fix rates, planning and scheduling	Designs an agent-based simulation model for providing directions about dynamic task priorities and resource assignments	High, complete maintenance process, part of planning and scheduling logic is shared. However, no dynamic strategies for repairing parts are provided. The model is less of reality and no costs are involved
7. Ward et al. (2010)	Yes	No	N-o	Location, direct outcome of maint., consequences, staff, suggestion, actions	Develops a comprehensive and ecological model of the operation system	Low, similar model purpose, overly abstract model, principally based on the manual analysis, hard to transfer to our research
8. Yang et al. (2008)	No	partly	N-o	Machine degradation level, maint. cost, spare parts, maint. resources	Proposes an approach for maint. scheduling based on the predicted equipment degradation	Medium, similar model purposes, abstract description of the simulation model, hard to reuse
9. Chang et al. (2007)	Yes	Yes	N-o	Total cost, labours, machine failures	Investigates the trade-off between maintenance personal staffing level and the throughput of a production line	Low, similar in scope, very limited maintenance processes, little applicable details to support model building
10. Duffuaa et al. (2001)	Yes	No	N-o	Input info., maint. type, planning and scheduling info., spares and equipment info., labours, quality and performance info.	Delivers a generic conceptual model for developing a realistic simulation model	High, similar in scope, explicit model, shares the concepts of the model, partly reuses the flow charts, however, no real simulation system is built

about the consequence that the impact of the delay for one aircraft on other aircraft cannot be investigated. Besides, the papers with low relevance mainly suffer from limited maintenance processes or overly simple model details to support our research. Half of the papers own a medium level of relevance. However, their contributions to our research are limited. Only one paper (Durazo-Cardenas et al., 2018) proposes a detailed conceptual model and discusses stakeholders but it does not involve simulation models. Sahnoun et al. (2015) provide a simulation model, whereas they just deliver a simple maintenance strategy and the design of turbine agents is less of reasonableness. Gary, Amos, and Tehseen (2018) present a model where only an analysis of an aggregated level can be performed. However, this model hardly describes the cooperation between stakeholders, which is less of reality. Yang, Djurdjanovic, and Ni (2008) accomplish a discrete-event model and employ the Genetic Algorithm to optimize maintenance schedules. Unfortunately, the details of simulation models cannot be observed and only cost factor is considered for optimizing the maintenance schedules.

Duffuaa et al. (2001) and MacKenzie et al. (2012) have a high level of relevance. However, they treat conceptual models and simulation models separately. MacKenzie et al. (2012) provides us with a proper "prototype" of our model. Since no conceptual models and cooperation between stakeholders are analyzed, the simulation model is hardly involved with important insights into the maintenance process for aircraft. On the other hand, Duffuaa et al. (2001) gives us a detailed analysis of maintenance concepts and maintenance flows, however, it can only serve as the theoretic support for the autonomous system design because of no simulation models.

To conclude, the existing simulation models are prone to tackling maintenance issues without a comprehensive understanding of maintenance concepts. They are not able to offer an insight into the maintenance process, which results in simple maintenance strategies (MacKenzie et al., 2012), overly simple algorithms (Sahnoun et al., 2015), only an aggregated level analysis (Gary et al., 2018), etc. While the papers of conceptual models tend to lack a concrete simulation demonstrator to illustrate the feasibility of the system design. Thus, we are motivated to build an autonomous system starting from requirement analysis and ending with a simulation system. Our simulation system can fully support the analysis of the complete maintenance process, enable a more in-depth analysis of maintenance issues and automatically optimize the maintenance scheduling and the part repairing.

3. Design of an automatic system for aircraft maintenance

The characteristics of maintenance systems for aircraft require that the systems are capable of being broken down into manageable fragments. In line with these principles, the autonomous system analysis framework is initially proposed to introduce an approach to build the system. The analysis of system requirements is then carried out. In the following, an architecture model for instructing the system design is proposed. Finally, the interactions between components are illustrated by UML diagrams.

3.1. Autonomous system analysis framework

The framework for building the autonomous system is divided into three parts: requirement analysis, autonomous system design and simulation system demonstration (Fig. 1).

The first step is to analyze requirements from the operational support for aircraft maintenance. Based on these requirements, an architecture schema is designed, which derives component models and a system interactive model. This system has been equipped with the very basic elements allowing operators of aircraft maintenance to globally automatically support maintenance decisions. The last step is to validate the system design by building a simulation system and by conducting simulation experiments. The results from simulation

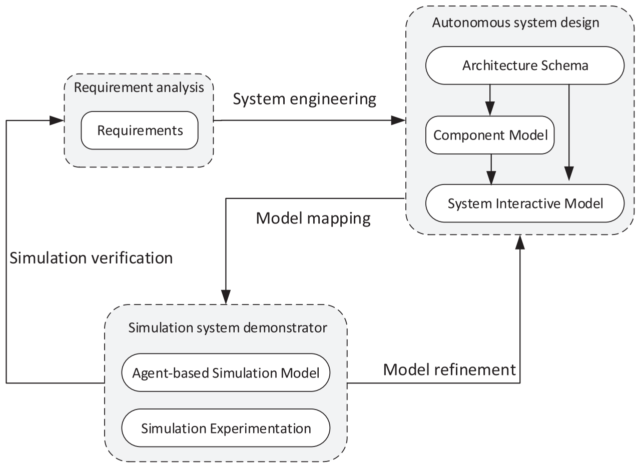

Fig. 1. Autonomous system framework for aircraft maintenance.

experiments will be applied to check requirements and to iteratively refine the system design. This framework tries to provide a methodology to build an autonomous system starting from requirement analysis, to system design, ending with simulation system demonstrator.

Our framework for developing the autonomous system on aircraft maintenance clarifies the practical value of the work. To begin with, it provides an alternative way to implement the "real system" based on the system design by building simulation systems, which enables us to gain important insights into future systems in an inexpensive way. Secondly, it offers a general approach to build an autonomous system that supports reuse and sharing. System designers in other fields are likely to provide a rigorous architecture design of the system and deliver a powerful demonstrator of the future system, which illustrates the feasibility of the design. At last, this framework proposes a methodology for building digital twins. It not only proposes a prototype system-of-interest but allows us to manipulate the prototype system with part of real data, in order to examine the reaction of the system.

3.2. Requirement analysis

The architecture of systems is a structure of components, their relationships and the principles and guidelines governing their design and evolution over time (ISO, 2011). This definition principally addresses components and relationships. Understanding the system overall aims is the first step to identify components and relationships. For example, what decisions will be made? What are the characteristics of the system?

Effective engagement with the stakeholders is vitally crucial to capture the new system requirements. The interview was conducted with a few experienced aircraft maintenance engineers in Chengdu, China. Because of confidential issues, several general questions were discussed, which are illustrated as follows:

- Q1: what are the major issues about aircraft maintenance?
- Q2: what is the level of autonomy required?
- Q3: what is the general information needed for a maintenance?
- Q4: what are the ways of accessing information?

The answers about questions 1–4 are concluded in Table 2, which is based on a 3-h interview. According to the answers, we can identify several problems, including "the data interaction of different IT systems is not sufficient", "an automatic maintenance system for aircraft maintenance is missing" and "A lack of an effective approach to make maintenance strategies and scheduling plans". As for autonomy, strategies for maintenance enables the maintenance system to adapt to real-

Table 2
The answers on questions 1–4.

No.	Answer
Q1	AMOs (Aircraft Maintenance Organizations) have multiple IT systems which are not located in the neighborhood of engineers and aircraft; The real-time progress of aircraft maintenance is highly demanded; AMOs are often not meeting the planned release dates of aircraft maintenance checks.
Q2	The high level of autonomy required spans several areas: maintenance strategy planning and updating,workers scheduling, resource response, cost and service efficiency.
Q3	The general maintenance information refers to parts, equipment, preventive scheduling, worker scheduling, inventory, task card, etc.
Q4	Information systems, telephones, emails are the major ways of communication.

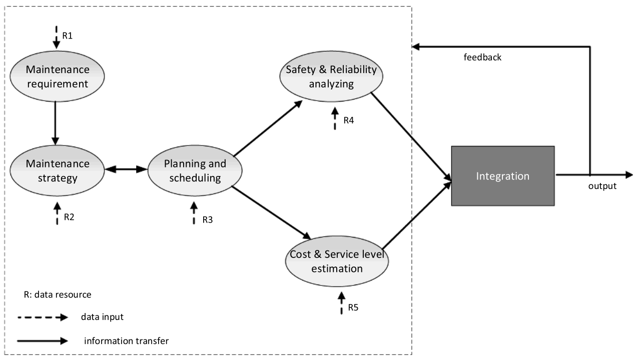

Fig. 2. The architecture model for the autonomous system.

time condition decision-making. The decision-making should be optimized with the consideration of service level and maintenance costs.

3.3. *Architecture development*

3.3.1. *Architecture model*

Based on the requirements (Table 2), an architecture model that covers most of the aspects for aircraft maintenance is highly demanded. Fig. 2 illustrates the architecture model for the autonomous system, which consists of 6 components: *Maintenance requirement*, *Maintenance strategy*, *Planning and scheduling*, *Safety* & *Reliability estimation*, *Cost* & *Service level estimation* and *Integration*. The inputs and outputs of components are synthesized in Table 3. In the following, the design of each component is discussed in detail.

Maintenance requirement. This component triggers the whole of maintenance activities. Maintenance requirements can be derived from PM, CM or third-party requests. Thus, the raw data involve the PM data, the sensor data, the alarm, and the third-party maintenance requirement reports. This component fuses the raw data to determine the basic maintenance request and severity level. Monitoring data can be from sensors, alarms, or the observation by crews. The PM for components is programmed by OEM at the design phase. It consists of not only the timing of maintenance, but maintenance activities, such as procedural instructions for maintenance tasks, procedures for recording the results of inspections, checks, tests, and other maintenance, etc. (Allen, 2012). Severity level depicts the degree of impacts of maintenance requirements (such as component faults, PM, etc.) on safety and reliability. The outputs of this component will trigger *Maintenance strategy* component.

Maintenance strategy. This component aims to analyze real-time conditions of parts in order to determine their corresponding maintenance strategies. The fusion process involves the synthesis analysis of the data from real-time inspection reports, sensors and troubleshooting databases. It is dedicated to dynamically determining maintenance strategies for parts and delivering maintenance goals and repair schema. The data of maintenance requests implies the whole description of maintenance including maintenance types, locations and priorities. The maintenance regulation refers to maintenance activities which are mandated to be executed. For example, the operator approved maintenance planning includes lots of documents like AD (Airworthiness Directive). AD provides maintenance management processes in order to ensure the airworthiness level for an aircraft. The data of the maintenance timing is used to optimize maintenance strategies via analyzing which maintenance can be delayed and dynamically updating maintenance types for parts.

Planning and scheduling. This component tries to automatically generate maintenance plans and schedules, including the optimization of the maintenance task sequence with the consideration of fundamental operating maintenance parameters such as repair time, cost, delay time and the availability of equipment and technicians, etc. Once maintenance strategies for parts are determined, the maintenance planning and scheduling will be defined accordingly. The input is used to define the task sequence by analyzing the availability of labours, parts and equipment. The remaining useful lifetime and interval are the basis for delivering repairing strategies. The maintenance historic data is utilized when some uncertain events happen during the process of repairing, in order to find a solution. The cost analysis affects ways of repairing as well. The final scheduling result of maintenance activities is presented by a Gantt chart.

Table 3
The table of components.

Component	Inputs	Outputs
Maintenance requirement	Monitoring sensor: fault info., location, etc. PM: time. Third-party request: maintenance requirement report	Maintenance request: type, location Severity level
Maintenance strategy	Maintenance request: type, location Maintenance regulation: operator approved maintenance planning Maintenance timing: remaining useful lifetime, interval, resource.	Maintenance goals and repair schema Maintenance strategies for relevant parts
Maintenance scheduling and planning	Information: remaining useful lifetime, intervals, strategies, labours, parts, equipment Maintenance historic data: trouble-shooting data Maintenance cost	Information: job completion, repair time, delay, interval Tasks scheduling: parts, labours
Safety & Reliability estimation	Airworthiness directives and service bulletins Information: the number of failed parts and corresponding time.	Safety indicator Reliability indicator
Cost & Service level estimation	Information: labour working time, parts, equipment, downtime, interval, etc. Flight information: flight No., aircraft No.	Service level indicator Cost indicator

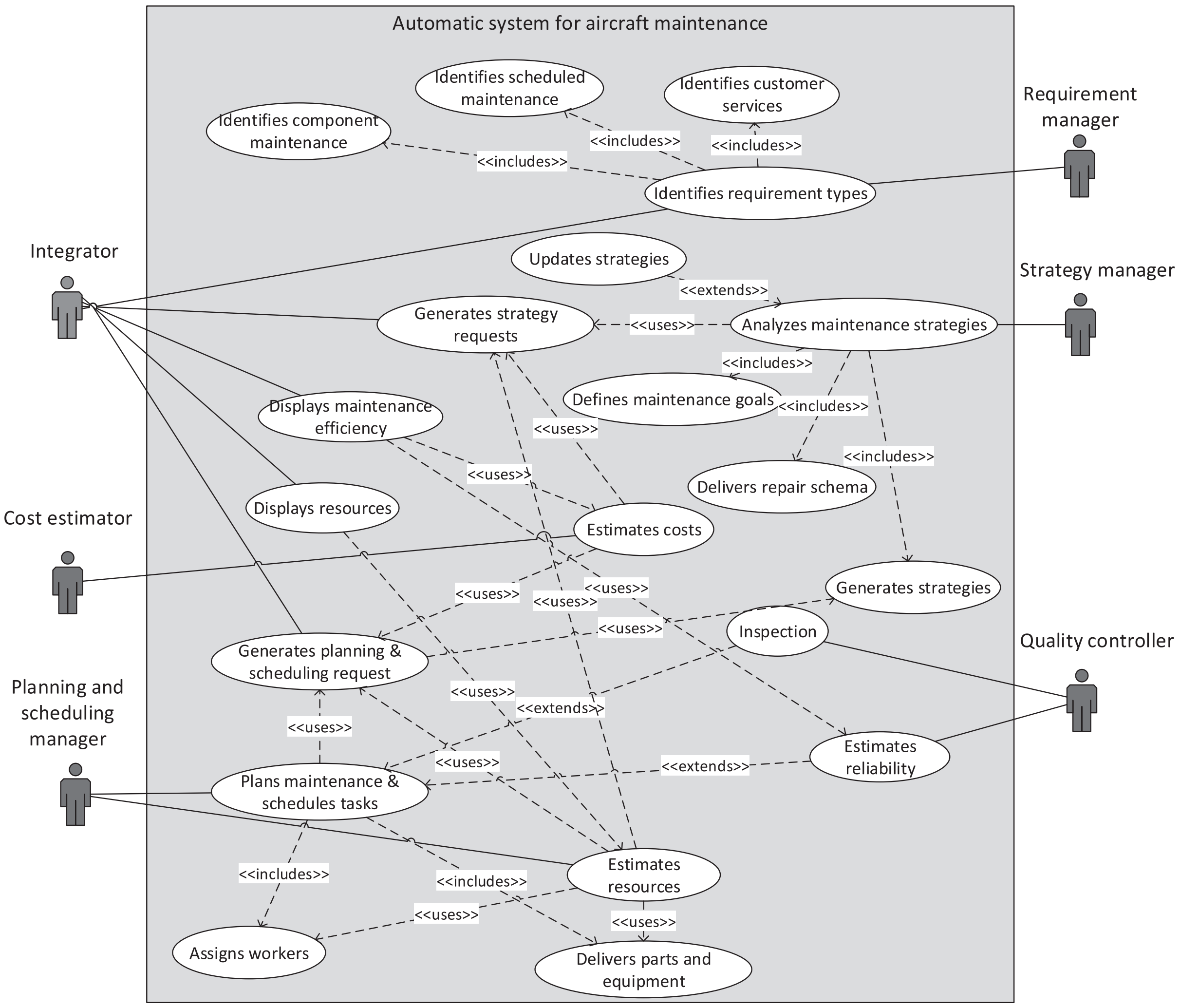

Fig. 3. Illustrative UML use case diagram for the autonomous system of aircraft maintenance.

Safety & Reliability estimation. The objective of this component is to collect the component failure data, current use time, changeover time, in order to accomplish the safety and reliability estimation. The result of the estimation will be the data for integration. Checking safety issues principally depends on AD and SB (Service Bulletin). These two documents describe the maintenance management process and the repair management process. If one of the steps is not compliant with standard processes, it will cause a safety risk. The system reliability analysis is conducted by analyzing the performance of parts. This component should ensure the safety of aircraft and provide information for product design.

Cost & Service level estimation. This component is employed to perform the estimation of maintenance costs and service level. The analysis of cost is the fusion process for the repair cost, part cost, logistic cost, staff's salary, the caused downtime cost, etc. The analysis of service level involves the assessment of delayed flights. The input is to determine the cost of labours, parts, equipment, downtime, etc. The delayed time for flights refers to the total repair time and interval. The efficiency of maintenance is evaluated by the synthesis analysis of cost and service level.

Integration. The aim of this component is to converge the estimation of safety & airworthiness and cost & service level, aviation authority regulation, resource availability into a global fused system output.

From Fig. 2, the integration model output delivers a global estimation for impacts of maintenance strategy, maintenance scheduling, resource, cost, service level and safety & airworthiness on the maintenance efficiency. The feedback provides recommendations to support better decisions and iteratively refine the maintenance process.

3.3.2. System module interaction

The system module interaction for the autonomous system of aircraft maintenance is presented by UML UCD (Use Case Diagram), which is shown in Fig. 3. UML UCDs have been widely used in the system module interaction (Skersys, Danenas, & Butleris, 2018). The complex maintenance process is illustrated by the interaction between actors in the way of use cases. It should be noted that:

- The integrator determines aircraft candidates by the synthesis analysis of resources and severity with the help of "Generate strategy requests";
- The "Updates strategies" implies the management of different maintenance strategies on components. Thus, strategy managers extend maintenance strategies when the interactions amongst strategies are considered;
- The planning and scheduling managers estimate the availability of resources, in order to provide a sequence of aircraft candidates. The plans and scheduling tasks will then be carried out.

4. Simulation system demonstrator

4.1. The architecture of the agent-based simulation system

In order to derive agents from component models, the relationship between components and agents are presented in Table 4. Combining component models and maintenance-related issues (flight, plane,

Table 4
The relationship between components and agents.

Agent	Component
Main	Integration
CRA	Maintenance requirement component
SSA	Maintenance strategy component
STA	Maintenance scheduling planning component
	Cost and service level estimation component
QCA	Safety and reliability estimation

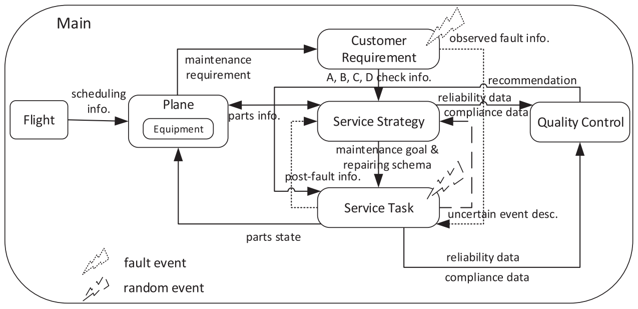

Fig. 4. The architecture model of the agent-based simulation system.

equipment, etc.), eight agents have been created, which can be divided into two groups: basic agents and maintenance agents. The basic agents refer to the Flight, Plane and Equipment. The maintenance agents consist of the Main, CRA (Customer Requirement Agent), SSA (Service Strategy Agent), STA (Service Task Agent) and QCA (Quality Control Agent). Fig. 4 provides an architecture model of the agent-based simulation system. In this figure, different maintenance information is delivered to different agents. The details of agents are explained in the following.

4.2. The development of agents

The development of agents has been implemented as the following steps: the inputs of agents, the processes of agents and the outputs of agents. All the relevant data about agents are summarized in Table 5.

4.2.1. Main and customer requirement agent

Main is designed as the integrator. It not only links with all the other agents but displays the key information of maintenance. This agent provides a graphical display of flight routes based on a GIS (Geographic Information System) map, where cities are really located and the flying time can be easily managed. The information on real-time flights scheduling table is stored in the database. In addition, maintenance states of each plane can be observed from this agent, which enables managers to control maintenance actions from the global point of view. Since this agent is the basis for the other agents, there is no input and output for it.

The maintenance sequence optimization is another key issue for Main. In our simulation environment, it is possible for more than two planes to be maintained at the same airport. The *FIFO* (First In First Out) rule is often applied in flight control (Atdelzater, Atkins, & Shin, 2000). However, only using the *FIFO* rule to deal with waiting planes cannot be always a good strategy, especially when the high efficiency of maintenance is exceptionally requested. In fact, this agent allows maintenance designers to investigate the performance of different strategies. For example, some other strategies like *FIFO-Optimized* strategy for waiting planes can be proposed. On the one hand, the least important PM for planes may be delayed to the next destination; on the other hand, the rest of maintenance will be scheduled with the *FIFO-Optimized* strategy. The *FIFO* rule is utilized to avoid the starvation of processes. Optimization operations are based on the *FIFO* rule. Each time when the temp message list is empty, it will copy all the messages

Table 5
The data of agents - DT (Data Type), I (Input data), and O (Output data).

Agent	DT	Functionality	Data			
Flight	I	Scheduling	flightNumber:String	aircraftNumber:int	flightTime:double	arrivalTime:Date
			destination:String	departureTime:Date	airport:String	interval:double
	O	Scheduling	scheduling table	cancelledFlights:int	delayedFlights:int	ontimeFlights:int
		States	CRA:int;SSA:int;STA:int	QC:int;type:String		
Plane	I	Planes	aircraftNumber:int	airport:String	aircraftType:AirPlaneType	
		Flights	flightNumber:String	flightTime:double	destination:String	
	O	Maintenance	maintenanceType:String	severity:int	parts:ArrayList<Equipment>	
Equipment	I	Equipment	type:EquipmentType	changeoverTime:double	serialNumber:int	repairTime:double
			repairingTime:double	PMFrequency:int	maintenanceStrategy:String	
			equipmentState:EquipmentState			
	O	Maintenance	repairType:String	countRepair:int	repairingTime:double	currentUseTime:double
			timeForFirstRepair:double			
CustomerRequ.	I/O	Parts	parts:ArrayList<Equipment>			
		Fault info.	faultDescription:String	faultLocation:String		
ServiceStrategy	I	Planes	aircraftNumber:int	airport:String	aircraftType:AirPlaneType	
			parts:ArrayList<Equipment>		interval:double	
	O	Parts	parts:ArrayList<Equipment>			
		Time	faultAnalysisTime:double		exceptionalEventSSATime:double	
		Strategy	maintenanceStrategy:String			
ServiceTask	I	Planes	aircraftNumber:int	airport:String	aircraftType:AirplaneType	
			partCheckingList:ArrayList<Equipment>		interval:double	
		Technicians	aircraftTechnicians:ArrayList<Person>			
	O	Cost	labourCost:double	partCost:double	equipmentCost:double	downtimeCost:double
		Time	planningTime:double	executionTime:double	downtime:double	partPreparingTime:double
			faultIdentificationTime:double		exceptionalEventSTATime:double	
QualityControl	I	Repairing	countRepair:int	repairingTime:double	currentUseTime:double	
			partCheckingList:ArrayList<Equipment>		timeForFirstRepair:double	
	O	Reliability	MTTR:double	MTTF:double	MTBF:double	Recommendation:String

from the message list. At the same time, the elements in the message list should be freed. It should be noted that the temp message list is used for optimizing the maintenance sequence for waiting planes and the message list is just employed to collect the messages. The *FIFO-Optimized* strategy has been implemented in Algo. 1. The time complexity of this algorithm is $O(n^3)$. In this algorithm, the estimation of severity and resource and the delay of the least important maintenance are conducted regarding airports. Because different airports are equipped with different capacities of the resource. This may result in changing maintenance sequences significantly. It should be noted that Algorithm 1 is different from Algorithm 3. The former focuses on scheduling faulty aircraft to repair (i.e. the sequence of "faulty" aircraft needed to be repaired); the latter concentrates on programming maintenance resources for maintenance tasks (i.e. the sequence of maintenance tasks needed to be performed).

Algorithm 1. Maintenance sequence optimization

```
input  : airportsList, messagesList, tempMessagesList
output: index.
// The index represents the number of the scheduled aircraft to be maintained.
// Firstly, the FIFO rule is used to deal with messages.
int threshold, count,index1; double max, value, min, severity, resource, interval;
max = min = severity = resource = value= count = index1 = interval = 0;
String maintenanceType = null;
// The threshold indicates the maximum number of aircraft to be maintained each time.
threshold=5;
if Main.receiveMessagesFromPlanes == true then
    messagesList.add(msg);
end
if tempMessagesList.IsEmpty() and !messagesList.IsEmpty() then
    for $i = 0$ to messagesList.size() do
        tempMessagesList.add(messagesList.get(i));
    end
    messagesList.clear();
end
// Then, based on the FIFO rule, prioritize maintenance sequences.
for $j = 0$ to airports.size() do
    max =0;
    for $k = 0$ to tempMessagesList.size() do
        if tempMessagesList.get(k).equals(airports.get(j)) then
            severity =tempMessagesList.get(k).severity;
            interval =tempMessagesList.get(k).interval;
            maintenanceType = tempMessagesList.get(k).maintenanceType;
            resource = resourceEstimate(airports.get(j));
            value = synthesize(severity, resource, interval, maintenanceType);
            if  $max < value$ then
                max = value; index = k;
            end
            count++;
        end
        // The least important PM may be delayed to the next destination.
        // min(messagesList,airport) means getting the minimum value of aircraft number from messages list at
           airport j
        while $(count - -) > threshold$ do
            for $l = 0$; tempMessagesList.get(l)==airports.get(j) to tempMessagesList.size() do
                if tempMessagesList.get(l).maintenanceType=="Corrective" or tempMessagesList.get(l).severity==2 then
                    break;
                end
                index1=min(tempMessagesList,airport.get(j));
                tempMessagesList.get(index1).severity++;
                tempMessagesList.remove(tempMessagesList.get(index1));
            end
        end
        count=0;
        send("Scheduling",tempMessagesList.get(index));
        tempMessagesList.remove(tempMessagesList.get(index));
    end
end
```

CRA sends maintenance requests to relevant agents after receiving signals from Main. If it is a PM, checking information will be sent to SSA. If it is a CM, fault information will be delivered to STA. As this agent principally manages requests and triggers maintenance activities, the input and output are the same.

4.2.2. Flight

Flight is a controller for scheduling planes. It schedules both on-time flights and delayed flights. The information on flights is stored in the database.

The scheduling strategy is implemented by two events: *scheduling* and *schedulingDynamical*. The *scheduling* event is a cyclic event where the first occurrence time is *departureTime* and the recurrence time is one day. The *schedulingDynamical* is a dynamic event. The occurrence of this event is totally determined by users via indicating the occurrence time. The scheduling strategy is described as the flowchart shown in Fig. 5.

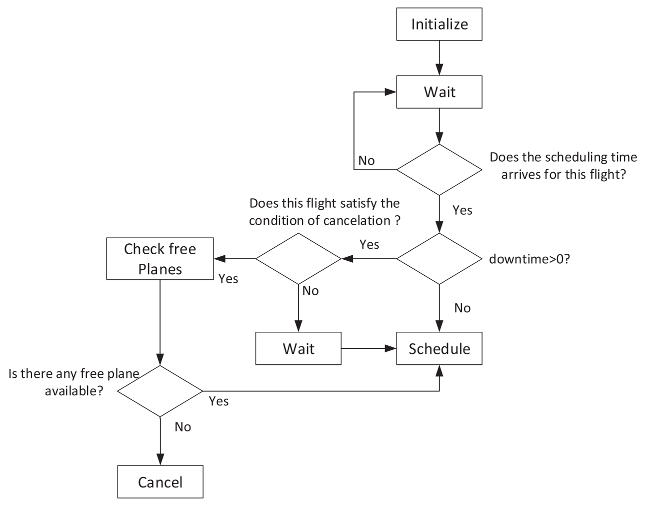
Initialize
Wait
No
Does the scheduling time arrives for this flight?
Yes
Does this flight satisfy the condition of cancelation ?
Yes
downtime>0?
Check free Planes
Yes
No
No
Wait
Schedule
Is there any free plane available?
Yes
No
Cancel

Fig. 5. The flowchart of flight scheduling strategy.

Consequently, the final state of one flight will be scheduled, delayed or cancelled. Since the scheduled information is determined, the "chain reaction" can happen due to some delays of flights. This phenomenon can significantly increase the complexity of maintenance and clearly show the high-level relevance of real-world maintenance issues of the simulation system. This agent will deliver the scheduling information and maintenance state information to Plane and Main respectively.

4.2.3. Plane and Equipment

Plane is the major carrier for maintenance. It is a composite agent, which contains a number of Equipment. When all the aircraft are loaded from the database, they will be ready at their corresponding airports. As soon as they receive the messages from Flight, they will fly to their destinations. When they reach their corresponding destinations, the maintenance will take place with a certain probability. According to the report of air transportation organization of Canada from 2007 to 2016 (TSBC, 2019), landing can be considered as the period when most of the incidents happen. Therefore, in this paper, maintenance is assumed to be started from the landing period. After the reparation, they are programmed to fly to next destinations. This cycle repeats every day.

Fig. 6 shows equipment states regarding the location of equipment. Failure events transfer the equipment state from *Use* to *Failure*. The state transition from *Use* to *Inspecting* is triggered by PM. If maintenance actions can be done on the plane, the state will pass from *Repairing* to *Use*. On the other hand, the equipment will be repaired in the workshop. Moreover, the stock will deliver any available equipment needed. It should be noted that *Repairing* and *Repair* states depict that maintenance activities take place on the spot and at workshops respectively.

During the life-cycle of equipment, recording the current use time of equipment is very important. This time will be the key factor to make a repairing decision. In this simulation environment, as soon as the equipment is loaded from the database, the current use time of this equipment starts. Only when it is placed in the stock, the current use time stops. The definition of the *currentUseTime* is illustrated as follows:

$$CUT_a = CT - ST; \tag{1}$$

$$CUT_b = CT - ST_i + LUT_{i-1}; \tag{2}$$

$$LUT_{i-1} = ET_{i-1} - ST_{i-1} + LUT_{i-2} \tag{3}$$

where, $i >= 1; LUT_0 = 0; CUT_b$ and CUT_a mean parts have been stayed at the stock or not respectively; the indices of equations are shown as follows:

CT (Current Time)	the current simulation running time;
CUT (Current Use Time)	the current use time of equipment;
ST (Start Time)	the time when equipment is leaving from stocks;
ET (End Time)	the time when equipment is put into stocks;
LUT (Last Use Time)	the last use time of equipment.

Algorithm 2. Repairing strategy analysis

```
input  : lastUseTime, changeoverTime
output: repairType
1  if $this.endTime == 0$ then // this condition means this
     equipment has never been in stock. The $time$ function
     gets the real time of simulation environment.
2      currentUseTime = time(MINUTE);
3  end
4  else
5      currentUseTime = lastUseTime + time(MINUTE) -
         StartTime;
6  end
7  if $currentUseTime > changeoverTime$ then
8      repairType = "replace";
9  end
10 else
11     if $equipmentState$ is Broken and $repairingTime >$
         $interval$ then
12         if Inventory is enough then
13             repairType = "replace& repair";
14         end
15         else
16             repairType = "repair";
17         end
18     end
19     else if $equipmentState$ is Broken and $repairingTime$
         $<= interval$ then
20         if $changeoverTime - currentUseTime <= 1200$
             then // the useful remaining life is less than 20h
21             repairType = "replace";
22         end
23         else
24             repairType = "repair";
25         end
26     end
27     else // The maintenance type is PM
28         repairType = "inspect";
29     end
30 end
```

The analysis of repairing strategies is realized in Algorithm 2. This algorithm deals with PM and CM. The repairing strategies include "replace", "replace& repair", "repair" and "inspect". The first three strategies are associated with CM. The "inspect" is used for PM. This algorithm is designed for comparing the *currentUseTime* and *changeoverTime* of equipment with respect to *equipmentState*, in order to find the optimal repairing strategy. In addition, the time complexity of this algorithm is $O(1)$ because no loop is involved.

4.2.4. Service strategy agent

SSA is intended for dynamically determining maintenance strategies for equipment and delivering maintenance goals and repair schema.

PM and CM are analyzed in this agent. If the maintenance type is PM, maintenance actions will be carried out on PM due parts. If the maintenance type is CM, this agent will automatically generate faulty part lists. The selection of parts takes place in a random way. On the other hand, we consider the influence of maintenance strategies on the same equipment. If PM occurs at equipment *i*, the probability of the fault occurrence at this equipment will be decreased. If CM happens at equipment *i*, the PM of this equipment should be rescheduled.

4.2.5. Service task agent

This agent is dedicated to programming maintenance tasks. The maintenance planning and task scheduling are based on the estimation of the resource availability and the real-time condition.

Algorithm 3 employs the same strategy as Algorithm 1. The time complexity of Algo. 3 is $O(n)$. The major differences between them are the maintenance location and key indicators. In Algorithm 1, maintenance objects (planes) are from airports all around the world and key indicators are relatively simple. While, in Algorithm 3, maintenance objects are at the same airport. As the most of details of maintenance tasks have been determined, the maintenance priority can be estimated by multiple indicators.

Algorithm 3. Maintenance tasks sequence optimization

```
input  : messagesList, tempMessagesList,
         aircraftTechniciansList, constraint[], interval.
// The index represents the number of scheduled
   aircraft to be maintained.
output: index.
// The FIFO rule in Algo.1 is reused to deal with
   messages from SSA. The details can be seen at
   lines 4 - 13, part 1 of Algo.1.
// Under the FIFO rule, prioritize the maintenance
   task sequence.
// XXEstimate(partsList) means synthesis estimation
   of parts, according to the service level, risk,
   delay, etc., based on historic data and real-time
   data.
double max, value, risk, delay, resource, serviceLevel;
max = value = serviceLevel = risk = delay = resource =0;
// The city means the airport of current aircraft.
String city;
for $i = 0$ to tempMessagesList.size() do
    partsList=tempMessagesList.get(i).partsCheckingList;
    risk = riskEstimate(partsList);
    serviceLevel = serviceLevelEstimate(partsList);
    delay = delayEstimate(partsList);
    availability = resourceEstimate(partsList);
    value = synthesize(serviceLevel, risk, delay,
      availability);
    if $max < value$ then
        max=value;
        index = i;
    end
end
// Assign resource to the selected plane's parts
   list
ArrayList<Person> workingTechnicians =new
  ArrayList<Person>();
city = tempMessagesList.get(index).city;
for $j = 0$ to aircraftTechnicians.size() do
    Person p= aircraftTechnicians.get(j);
    if p.airport.equals(city) and p.schedule == 1 and
      p.free == true then
        p.free = false;
        workingTechnicians.add(p);
    end
end
tempMessagesList.remove(tempMessagesList.get(index));
```

4.2.6. Quality control agent

QCA analyzes the reliability of equipment and checks safety issues depending on AD and SB. Kullstam (1981) proposed a few reliability indicators like MTTF (Mean Time To Failure), MTBF (Mean Time Between Failure) and MTTR (Mean Time To Repair). MTTF is a general reliability index for estimating mean time to failure. MTTR is a basic measure of the maintainability of repairable items. It represents the average time required to repair a failed component or device. MTBF is the predicted elapsed time between inherent failures of a mechanical or electronic system, during the normal process of the system operation. During the process of simulation, all the relevant information will be collected to perform the reliability analysis. The results are compared with target repair time, target MTBF, and target MTTR respectively, in order to get the corresponding performance.

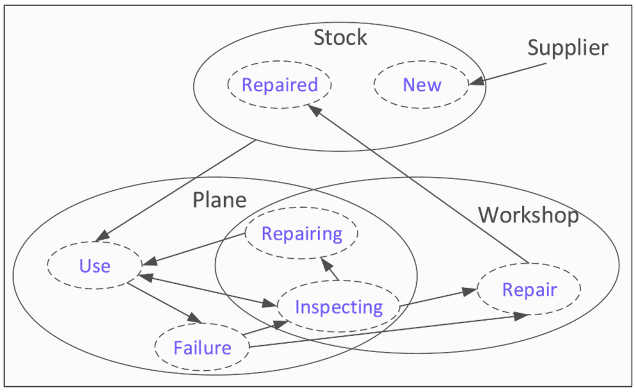

Fig. 6. The state transition of equipment.

Safety analysis is achieved by utilizing AD and SB. The maintenance management process and repair management process are supervised. Once these processes are violated, warnings will be sent out.

4.3. Cost and service level analysis

In terms of Maintenance Cost (MC), Downtime Cost (DC), Labour Cost (LC), Stock Cost (SC) and Parts Purchase Cost (PPC) are taken into consideration. The definitions of each cost (Eqs. (4)–(8)) are provided, where *totalRepairTime* (shown in Table 6) and *interval* are counted by seconds, and $unitPrice_{DC}$ means the unit price for downtime; *salary(i)* and *time(i)* represent the *i*th aircraft technician's salary counted by hours and the corresponding working time; *number(i)* implies the number of the *i*th kind of parts, $unitPrice_{SC}$ means the unit price for stock and *days* means the number of storing days; and *cost(j)* means the cost of the *j*th kind of parts. In addition, all the costs are counted by dollars.

$$MC = DC + LC + SC + PPC; \tag{4}$$

$$DC = (totalRepairTime - interval) * unitPrice_{DC}; \tag{5}$$

$$LC = \sum_{(i=1)}^{n} salary(i) * time(i); \tag{6}$$

Table 6
The compositions of total repair time on maintenance scenarios - UE (Uncertain Events).

Scenario	Total repair time
PM	informationDelivery, maintenancePlanning, inventoryPrepare, repairing,
PM + UE	**exceptionalEventSTA(eSTA)**, scheduledTime **exceptionalEventSSA(eSSA)**.
CM	**faultIdentification, faultAnalysis**, scheduledTime.
CM + UE	**faultIdentification, faultAnalysis**, **eSTA**, **eSSA**, scheduledTime.

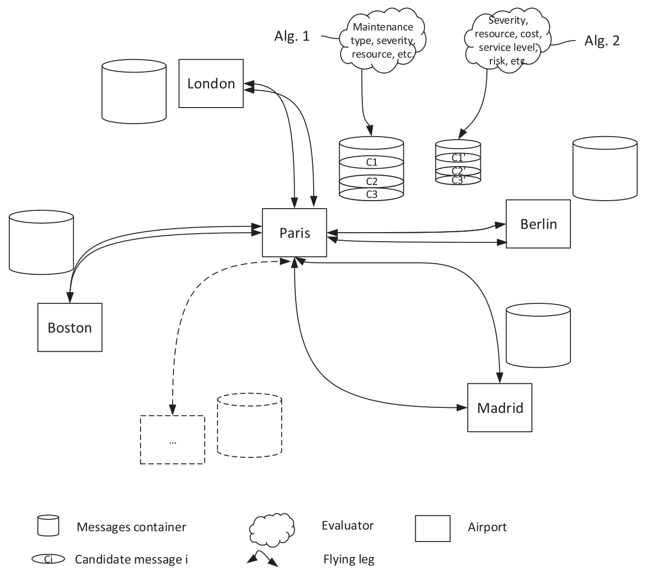

Fig. 7. The simulation experiment scenario.

$$SC = \sum_{(i=1)}^{n} number(i) * unitPrice_{SC} * days; \tag{7}$$

$$PPC = \sum_{(j=1)}^{n} cost(j); \tag{8}$$

The complexity of MC depends on maintenance scenarios. As shown in Table 6, fault events and uncertain events are considered in maintenance scenarios. The analysis of fault events aims to deliver the causes of faults on components. The uncertain event analysis tries to simulate the process of the trouble-shooting during the process of repairing when exceptional issues happen. Fault and uncertain events take place with a user-defined probability. The elements of *totalRepairTime* will be discussed in detail at the last paragraph of Section 5.2. Additionally, *scheduledTime* in Table 6 means the total repair time of PM.

The definition of service level is shown as follow:

$$ServiceLevel = totalRepairTime - interval. \tag{9}$$

Service level reflects the degree of the delay time of passengers. It is another way to evaluate maintenance efficiency. Obviously, decreasing the maintenance cost all the time is not definitively a good idea. Because it may produce numerous delays or even unnecessary cancels of flights. In this paper, when the difference between *totalRepairTime* and *interval* is more than 4 h for one flight, it is supposed that this flight will be cancelled.

5. Simulation experiment

Simulation experiments are carried out to evaluate the impacts of maintenance strategies of *FIFO, Optimized*, and *FIFO + Optimized* (supported by Algorithm 1) on maintenance costs and service level.

5.1. Scenario

The simulation starts from the flight scheduling. The aircraft stay at airports waiting for scheduling signals. Flight legs (A flight leg is a trip of an aircraft, from take-off to landing) are from the website of *AirFrance*.[1] Twelve airports are concerned, including *Paris, Lyon, London, Berlin, Madrid, Amsterdam, Boston, Marseille, Barcelona, Prague, Manchester* and *Munich*. There are 50 aircraft distributed at different airports. Fig. 7 shows the general process of aircraft maintenance for the simulation. At each airport, maintenance sequences and maintenance tasks sequence will be analyzed. In this figure, an example is given for *Paris*, where the maintenance sequence optimization and maintenance tasks sequence optimization are illustrated. Message containers for the maintenance sequence and maintenance tasks sequence are used to collect signals from aircraft for requesting maintenance and for requesting resources respectively. Optimization processes are explained in Sections 4.2.1 and 4.2.5.

Each aircraft performs routes given by flight plans. Different types of aircraft own different equipment. The deterioration of each equipment conforms to its degradation curve. Each airport is capable of undertaking all kinds of maintenance tasks. Not all resources at airports are the same. The data for the flight plan, the available person-power for airports, and the maintenance-related time are provided in the appendix.

5.2. Hypothesis

The major experiment hypotheses consist of:

- fault events can happen only when aircraft reach “Landing” state;
- each faulty part only occupies one technician who is randomly distributed and has matched skills;
- maintenance business processes are included in the combination of PM/CM and with/without uncertain events;
- for *departure* event, if the downtime for departure is over 4 h, this flight will be cancelled;
- no degradation for components happens when components are stored in the warehouse;
- the items of total repair time (shown in Table 6) are assigned with random values or are determined by the database.

As for the last hypothesis, the industrial relevance of values should be discussed. Assigning these data with values helps the simulation system understand the relevant degree of time-consuming for parameters. We try our best to provide the values with high industrial relevance. For example, the values for the data *inventoryPrepare* (1–3 h) (when the relevant part is out of stock), *repairingTime* (20–120 min) (the faulty parts that can be repaired on the spot) and *repairTime* (5–30 h) (the faulty parts that have to be repaired at the workshop) are of a certain level of industrial relevance. However, in terms of the data *informationDelivery* and *maintenancePlanning*, the values for them are assumed as 2–5 s, which are of less of industrial relevance. The data of *faultIdentification, exceptionalEventSSA* and *exceptionalEventSTA* are determined, based on the knowledge stored in the database. If the solution has been stored in the database, they will take 0 min. If not, they will take either 30 min or one hour. Also, the unit prices (Eqs. (5) and (7)) are preassigned as well. The unit price of downtime cost per second is set to be 2.8 dollars (Pohl, 2019). The unit price of stock cost is generally supposed to be 10 dollars per day, which is less of relevance.

5.3. Experiment results

This simulation experiment is implemented by Anylogic PLE 8.2.3.[2] The configuration of the running laptop is 8G memory and 4 processors (i7-6500U CPU). The run length is set to be three months, which takes around three hours. PM and CM, involving changing parts, purchasing parts, inspecting parts, shifting workers, delayed flights, cancelled flights, etc., will happen within this run length.

The warm-up period is set to be five days to avoid initialization bias since the simulation system starts with new planes.

Moving average analysis is carried out to set the required number of

[1] www.airfrance.fr (accessed in January 2019)

[2] www.anylogic.com (accessed in January 2019).

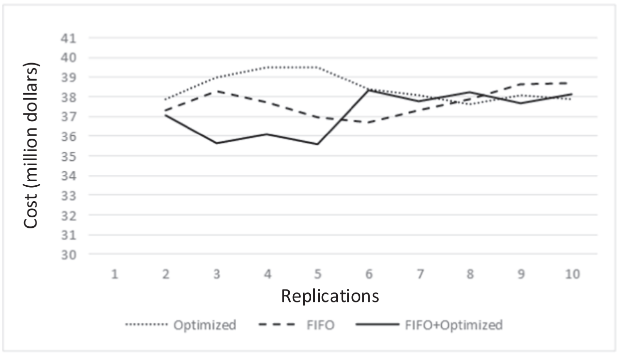

Fig. 8. The moving averages of three strategies.

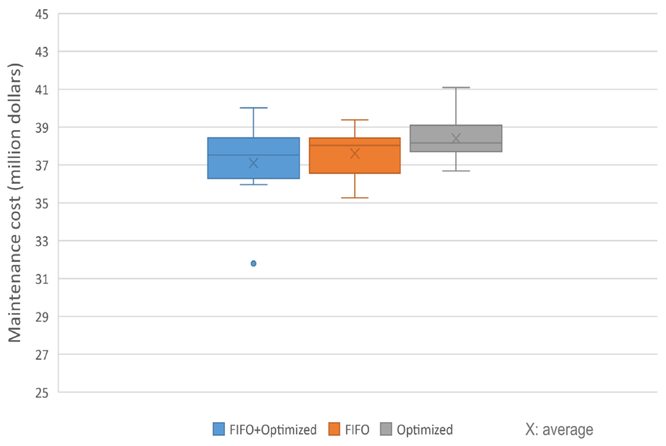

Fig. 9. The impact of maintenance sequence strategies on maintenance costs.

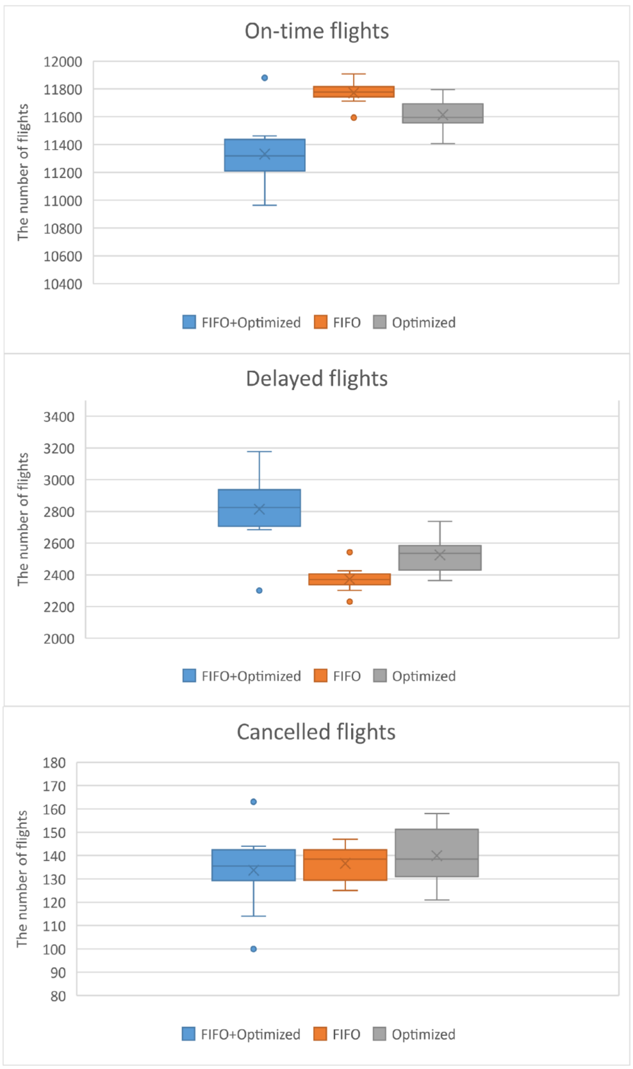

Fig. 10. The impacts of maintenance sequence strategies on service level.

replications. According to the result (shown in Fig. 8), after the 8*th* replication, the moving average lines tend to be more stable. Since the moving average lines fluctuate with a minimum amplitude around the 10*th* replication, the number of replications will be set to be ten to ensure we obtain a better estimate of average. The box plots are used to show the distribution of the results. The impact of different strategies on maintenance costs is illustrated in Fig. 9. This figure depicts that three strategies share similar ranges of change on maintenance costs except for the minimum of the strategy of *FIFO* + *Optimized*. The strategy of *Optimized* does not seem to have any advantages on maintenance costs, regarding all the indicators. Comparing with the strategy of *FIFO*, *FIFO* + *Optimized* strategy saves the maintenance cost more in general. However, it is possible for *FIFO* strategy to outperform *FIFO* + *Optimized* strategy in maintenance costs.

In terms of service level, the results about on-time, delayed and cancelled flights can be observed in Fig. 10 in detail. Overall, the strategy of *FIFO* outperforms the others. While the strategy of *FIFO* + *Optimized* has the lowest average of the number of cancelled flights. Comparing with these three strategies, *FIFO* is the most stable strategy in every aspect, which is followed by *Optimized* strategy. The least stable strategy is *FIFO* + *Optimized*.

From our results, we can conclude that the best maintenance sequence strategy will vary in the priority of minimizing maintenance costs or maximizing the number of on-time flights. If minimizing maintenance costs is the priority then *FIFO* + *Optimized* strategy would be the best. This strategy aims to always handle the most urgent maintenance issues to some extent. No "starvation phenomenon" will happen. It always keeps the priority and sequence of maintenance requests balanced, so this strategy will save maintenance costs most. If maximizing the number of on-time flights is the priority then *FIFO* strategy would be the best. This strategy handles the requests from aircraft depending on the order of the request arriving time. Therefore, the phenomenon of "the least urgent request cannot obtain resources" can be avoided. The number of on-time flights will be increased. So, this strategy results in the largest number of on-time flights. As a result, the efficiency of maintenance strategies should be investigated from different aspects. Different strategies should be adapted to different aims.

5.4. *Discussion*

An autonomous maintenance system has been designed and demonstrated by utilizing SE and ABMS respectively, which aims to improve the operational support for aircraft maintenance.

The architecture model for the system provides a global view about components and their relationships in the system. It guides us to implement the lower development of components, which involves the inputs, outputs and functions of components. Hence, the maintenance-related data can be installed at the corresponding components properly, which makes it possible to accomplish the transformation from real-time data to global maintenance decisions.

The simulation system is the demonstrator to simulate the whole process for aircraft maintenance. Many factors, including strategies, scheduling plans, the cost analysis and the service level analysis, have been addressed in this system.

The analysis of strategies consists of the maintenance sequence optimization strategy (Algorithm 1), the repairing strategy (Algorithm 2), and the maintenance task sequence optimization strategy (Algorithm 3). These strategies improve the efficiency of aircraft maintenance. The maintenance sequence optimization strategy states that the requests from aircraft should be answered in an optimized way. From the experiment results, we can see that the judge of the best strategy relies on the chosen priority and the synthesis of real-time conditions.

The planning and scheduling task scheduler uses Algorithm 3 to enable the allocation of maintenance jobs to groups of MRO workers, providing cost-effective plans for dealing with maintenance issues in a simple and timely way.

The cost analysis uses a simple cost-breakdown structure (Eqs. (4)–(8)) to estimate the maintenance cost; but because of the complexity of the effect of repair time (shown in Table 6), multiple values are possible for any given task.

The analysis of service level is another way to evaluate maintenance efficiency. Even though the definition of service level is quite simple, it could make the analysis of maintenance much more complicated. The concept of service level allows aircraft to be delayed or cancelled. Since flying routes of aircraft have already planned, it may cause the “chain reaction”. For instance, as the resource of one airport is not well evaluated, it may lead to the resource competition between CM and loose PM. In other words, the loose PM could be delayed to some other airports (details see Algorithm 1). Consequently, more aircraft will have to wait for resources. It will then cause the delay of flights. Hence, the “chain reaction” will happen between Flight and Plane. So, a delay of an aircraft may cause multiple delays of other aircraft. The real-time condition varies from time to time. The planned scheduling could be useless. As a result, quick response to real-time condition becomes vitally important to improve maintenance efficiency.

As above, our simulation system is a successful proof of demonstrating the design of the autonomous system of aircraft maintenance.

6. Conclusion and research perspectives

Aiming at improving the operational support for aircraft maintenance, an integrated approach that fuses aircraft’s condition, strategy, planning and cost has been proposed. First of all, an autonomous system analysis framework is provided to illustrate the approach from the global point of view. A high-level architecture for the autonomous system is then developed to put forward the fundamental components and the relationship between them. The black box approach is employed to derive the underlying inputs and outputs of each component with respect to the component functionality. The UML use case diagram is used to illustrate the system module interaction. This helps deal with the complexity of the system design and enables smoother code development.

In order to illustrate the feasibility of the design of the autonomous system, a solid simulation system demonstrator has been accomplished by using *Anylogic*. The autonomous system is implemented in the way of the agent-based simulation system where components are instantiated as agents. The strategies, the planning and scheduling, the cost analysis and service level analysis have all been achieved in the simulation system. The impacts of strategies on maintenance costs and service level have been investigated from experiment results.

Even though experiments have been performed to validate the functionality of the proposed algorithms the rationality of design for the autonomous system, the feasibility of interactions between agents and behaviours of the simulation system can hardly be assured in the long run. Therefore, the study of the formal verification on the logics of the simulation system will be our future work.

Acknowledgements

This work was supported by the China Scholarship Council.

Appendix A. Maintenance-related data

Tables A.7–A.9.

Table A.7
The distribution of labours, where P1 = (00:00–05:00), P2 = (05:00–07:00), P3 = (07:00–14:00), P4 = (14:00–17:00), P5 = (17:00–20:00) and P6 = (20:00–00:00).

Airport	P1	P2	P3	P4	P5	P6
Paris	8	1	8	8	10	5
Lyon	2	1	3	3	3	2
London	2	1	3	3	3	2
Berlin	2	1	3	3	3	2
Madrid	2	1	3	3	3	2
Amsterdam	2	1	4	4	3	1
Boston	2	1	5	5	5	2
Marseille	1	1	3	3	3	2
Barcelona	1	1	3	3	3	1
Prague	1	1	3	3	3	1
Manchester	1	1	3	3	3	2
Munich	2	1	3	3	3	1

Table A.8
Flight plan for aircraft 10001 and 10004.

Flight No.	Aircraft No.	Airport	Destination	Flight time	Departure time	Arrival time
BA0304	10001	Lyon	Paris	70	2018-1-1 07:20	2018-1-1 08:30
BF0321	10001	Paris	Lyon	70	2018-1-1 09:30	2018-1-1 10:40
BF0333	10001	Lyon	Paris	70	2018-1-1 11:40	2018-1-1 12:50
BF0334	10001	Paris	Lyon	70	2018-1-1 15:30	2018-1-1 16:40
BF0335	10001	Lyon	Paris	70	2018-1-1 19:00	2018-1-1 20:10
BF0336	10001	Paris	Lyon	70	2018-1-1 21:30	2018-1-1 22:40
BA0800	10004	Lyon	Paris	70	2018-1-1 08:20	2018-1-1 09:30
BA0213	10004	Paris	Boston	475	2018-1-1 12:25	2018-1-1 20:20
BA0143	10004	Boston	Paris	475	2018-1-1 21:30	2018-1-2 05:25
BA0632	10004	Paris	Lyon	70	2018-1-2 06:00	2018-1-2 07:10

Table A.9
Equipment information.

Equipment type	Changeover time (hour)	Repair time (h)	Repairing time (min)	PM frequency (week)	Price (dollar)
motive_generator	5000	20	80	5	5000
APU_generator	5000	20	80	5	5000
fuel_tanks	1000	15	60	5	5000
fuel_pumps	1000	15	60	5	5000
chassis	800	20	80	6	3000
IRU	5000	5	20	6	1000
SG	5000	5	20	5	8000
FCC	3000	5	20	7	3500
MCP	3000	5	20	7	2000
FMC	3000	5	20	9	15,000
TCAS	5000	5	20	9	12,000
radar	6000	5	20	5	1000
CDU	3000	10	40	5	10,000
APU	44,000	20	80	5	70,000
engine	6500	20	80	12	30,000
nose_cone	69,000	70	80	10	90,000

References

Ahmed, K. A., & Azadian, F. (2017). Airline flight schedule planning under competition. *Computers & Operations Research, 87*, 20–39.

Ali, S. M., Doolan, M., Wernick, P., & Wakelam, E. (2018). Developing an agent-based simulation model of software evolution. *Information and Software Technology, 96*, 126–140.

Allen, J. M. (2012). *Air carrier maintenance programs*. U.S. Department of Transportation.

Atdelzater, T. F., Atkins, E. M., & Shin, K. G. (2000). QoS negotiation in real-time systems and its application to automated flight control. *IEEE Transactions on Computers, 49*(11), 1170–1183.

Blanchard, B. S. (2004). *System engineering management.* John Wiley & Sons.

Chang, Q., Ni, J., Bandyopadhyay, P., Biller, S., & Xiao, G. (2007). Maintenance staffing management. *Journal of Intelligent Manufacturing, 18*(3), 351–360.

DeBruecker, P., Beliën, J., Van den Bergh, J., & Demeulemeester, E. (2018). A three-stage mixed integer programming approach for optimizing the skill mix and training schedules for aircraft maintenance. *European Journal of Operational Research, 267*(2), 439–452.

Diaz, J., Huertas, J. I., & Trigos, F. (2014). Aircraft maintenance, routing, and crew scheduling planning for airlines with a single fleet and a single maintenance and crew base. *Computers & Industrial Engineering, 75*, 68–78.

Duffuaa, S. O., Ben-Daya, M., Al-Sultan, K. S., & Andidjani, A. A. (2001). A generic conceptual simulation model for maintenance systems. *Journal of Quality in Maintenance Engineering, 7*(3), 207–219.

Durazo-Cardenas, I., Starr, A., Turner, C. J., Tiwari, A., Kirkwood, L., Bevilacqua, M., ... Emmanouilidis, C. (2018). An autonomous system for maintenance scheduling data-rich complex infrastructure: Fusing the railways' condition, planning and cost. *Transportation Research Part C: Emerging Technologies, 89*, 234–253.

Gary, L., Amos, N. H. C., & Tehseen, A. (2018). Towards strategic development of maintenance and its effects on production performance by using system dynamics in the automotive industry. *International Journal of Production Economics, 200*(C), 151–169.

Helbing, D. (2012). Agent-based modeling. *Social self-organization* (pp. 25–70). Springer.

Heppenstall, A. J., Crooks, A. T., See, L. M., & Batty, M. (2011). *Agent-based models of geographical systems.* Springer Science & Business Media.

ISO. (2011). *Systems and software engineering–architecture description.* Technical Report. ISO/IEC/IEEE 42010.

Keysan, G., Nemhauser, G. L., & Savelsbergh, M. W. P. (2010). Tactical and operational planning of scheduled maintenance for per-seat, on-demand air transportation. *Transportation Science, 44*(3), 291–306.

Kullstam, P. A. (1981). Availability, MTBF and MTTR for repairable M out of N system. *IEEE Transactions on Reliability, 30*(4), 393–394.

Macal, C. M., & North, M. J. (2005). Tutorial on agent-based modeling and simulation. *Simulation conference, 2005 proceedings of the winter* (pp. 14). IEEE.

MacKenzie, A., Miller, J. O., & Hill, R. R. (2012). Application of agent based modelling to aircraft maintenance manning and sortie generation. *Simulation Modelling Practice and Theory, 20*(1), 89–98.

Monostori, L., Váncza, J., & Kumara, S. R. T. (2006). Agent-based systems for manufacturing. *CIRP Annals-Manufacturing Technology, 55*(2), 697–720.

Moyaux, T., Chaib-Draa, B., & D'Amours, S. (2006). Supply chain management and multiagent systems: An overview. *Multiagent based supply chain management* (pp. 1–27). Springer.

Pohl, Thomas (2019). Cost per hour of downtime per aircraft is 10,000 USD more. Accessed 2019–01-10 https://blogs.sap.com/2013/05/02/cost-per-hour-of-downtime-per-aircraft-is-10000-usd-more.

Royce, W. W. (1987). Managing the development of large software systems: concepts and techniques. *Proceedings of the 9th international conference on Software Engineering* (pp. 328–338). IEEE Computer Society Press.

Sahay, A. (2012). *Leveraging information technology for optimal aircraft maintenance, repair and overhaul (MRO).* Elsevier.

Sahnoun, M., Baudry, D., Mustafee, N., Louis, A., Smart, P. A., Godsiff, P., & Mazari, B. (2015). Modelling and simulation of operation and maintenance strategy for offshore wind farms based on multi-agent system. *Journal of Intelligent Manufacturing*, 1–17.

Sama, P., & Kiridena, S. (2012). Aircraft maintenance planning and scheduling: An integrated framework. *Journal of Quality in Maintenance Engineering, 18*(4), 432–453.

Siebers, P.-O., Macal, C. M., Garnett, J., Buxton, D., & Pidd, M. (2010). Discrete-event simulation is dead, long live agent-based simulation!. *Journal of Simulation, 4*(3), 204–210.

Skersys, T., Danenas, P., & Butleris, R. (2018). Extracting SBVR business vocabularies and business rules from UML use case diagrams. *Journal of Systems and Software, 141*, 111–130.

TSBC (2019). Statistical summary – aviation occurrences 2016. Accessed 2019–01-10. http://www.tsb.gc.ca/eng/stats/aviation/2016/ssea-ssao-2016.asp.

Waltz, E., & Hall, D. L. (2001). Requirements derivation for data fusion systems. *Handbook of Multisensor Data Fusion,* 15.

Ward, M., McDonald, N., Morrison, R., Gaynor, D., & Nugent, T. (2010). A performance improvement case study in aircraft maintenance and its implications for hazard identification. *Ergonomics, 53*(2), 247–267.

Yang, Z. M., Djurdjanovic, D., & Ni, J. (2008). Maintenance scheduling in manufacturing systems based on predicted machine degradation. *Journal of Intelligent Manufacturing, 19*(1), 87–98.

Zhang, Z., Liu, G., Jiang, Z., & Chen, Y. (2015). A cloud-based framework for lean maintenance, repair, and overhaul of complex equipment. *Journal of Manufacturing Science and Engineering, 137*(4), 040908.