FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Impact of the Organizational Structure on Operations Management

The Airline Operations Control Centre Case Study

Nuno Gonçalo Machado

Dissertation

Master in Informatics and Computing Engineering

Supervisor: Eugénio Oliveira (PhD)

Second Supervisor: António Castro (MSc)

28^{th} June, 2010

Impact of the Organizational Structure on Operations Management

The Airline Operations Control Centre Case Study

Nuno Gonçalo Machado

Dissertation
Master in Informatics and Computing Engineering

Approved in oral examination by the committee:

Chair: António Augusto de Sousa (Professor Associado)

External Examiner: Elisabete Arsénio Guterres de Almeida (Investigadora Auxiliar)
Internal Examiner: Eugénio da Costa Oliveira (Professor Catedrático)

23^{rd} July, 2010

Abstract

Be it in a university, in a manufacturing company or in an airline business, if a set of entities are collectively collaborating and contributing toward one common goal, then we have an organizational structure. Nowadays, with the increasing complexity of goods and services, and competing in a globalized world, organizations require tuned work systems, involving human capital interwoven with the latest technological innovations.

Evolving an established organizational structure is often daunting when it is behind the core mission of a business or when it operates uninterruptedly. In these cases, software simulations are an invaluable tool to explore new work practices, information flows or even decision making processes. Modeling and simulating complete or small portions of critical workflows, makes it feasible to collect a set of metrics as well as introducing organizational transformations. Brought together, these factors allow for organizational performance assessment and evolution.

This research project is founded on such observations and aimed at proposing improvements to the operational control within a real airline company. To accomplish this, it departed from the empirical knowledge conveyed through interviews with airline operators and built an analytical infrastructure geared towards evaluating the current and hypothetical organizational structures.

At its core, this study pioneered the first simulation of a real airline operations control scenario, involving human actors, existing computerized systems, time and spatial location, operational activities and reasoning processes. Besides the faithful modeling of airline entities, it used *pre* and *post* real operational data to better reproduce workflow inception and disruption handling.

Along with operational performance assessment, the conducted research also delved into the decision making practices of the Airline Operational Control Centre specialists, performing a comparison between empirical probabilistic action and tangible solutions obtained through operational records analysis. The usage of learning techniques was demonstrated as a mean to optimize the reasoning accuracy within the simulation.

In terms of tools, *Brahms*, a human-centered multi-agent environment was used to implement and simulate the conceptual representation of the airline operational entities. The academia and research nature of this tool makes it lack a large user base, that justified the publication of educational tutorials.

By the end of the research study, the simulation of the same operational scenario across four distinct organizational structures demonstrated improvements up to 15% in disruption handling time and up to 21% in collaborator *stress*. The usage of decision tree classifiers made it possible to raise the 21% reasoning accuracy from the probabilistic empirical model to 86%.

Resumo

Seja numa universidade, na indústria, ou numa companhia aérea, quando um conjunto de entidades contribui e colabora no sentido de alcançar um objectivo comum, está-se em presença de uma estrutura organizacional. Nos dias de hoje, com a crescente complexidade de bens e serviços e enfrentando competição a nível internacional, as organizações necessitam de sistemas de trabalho eficientes, envolvendo capital humano interligado com as últimas inovações tecnológicas.

As simulações através de *software* assumem especial relevância na exploração de novas práticas de trabalho, fluxos de informação ou processos de decisão. A modelação e simulação de fluxos de trabalho torna possível a recolha de um conjunto de métricas bem como introduzir diversas transformações à estrutura organizacional vigente. Em conjunto, estes factores permitem avaliar e optimizar o desempenho de uma organização.

O presente projecto de investigação surge das observações acima e tem como objectivo último propor melhorias ao nível do controlo operacional de um companhia aérea real. Para o conseguir, parte do conhecimento empírico transmitido através de entrevistas com os principais colaboradores da companhia aérea e constrói uma infra-estrutura analítica destinada à avaliação da actual e hipotéticas estruturas organizacionais.

Este estudo protagoniza a primeira simulação de um cenário de controlo operacional no interior de uma companhia aérea, envolvendo colaboradores, sistemas computacionais, localização espaço-temporal, actividades operacionais e processos de raciocínio. Para além da fiel modelação das entidades aeroportuárias, são utilizados dados *pre* e *pós*-operação reais com vista a reproduzir o arranque de fluxos de trabalho e a gestão de perturbações de forma mais fidedigna.

No seguimento da avaliação operacional, a investigação conduzida aborda também as práticas de tomada de decisão ao nível do Centro de Controlo Operacional da companhia aérea. Aqui estabelece-se a comparação entre acções probabilísticas empíricas e soluções tangíveis obtidas pela análise dos registo da operação. A utilização de técnicas de aprendizagem supervisionada é demonstrada como veículo de optimização dos processos de raciocínio embebidos na simulação.

Em termos de ferramentas, o ambiente multi-agente *Brahms* foi utilizado para modelar e simular as entidades associadas à companhia aérea. A natureza académica e científica desta ferramenta justifica uma restrita e parca comunidade de utilizadores, facto que motivou a publicação de diversos artigos educacionais.

No final deste estudo, a simulação do mesmo cenário operacional tendo por base quatro estruturas organizacionais distintas demonstrou melhorias até 15% no tempo de gestão de perturbações aéreas e até 21% ao nível do *stress* dos colaboradores. A utilização de classificadores baseados em árvores de decisão tornou possível aumentar a precisão dos processos de raciocínio de 21%, associado ao modelo probabilístico empírico, para 86%.

Acknowledgements

First and foremost, I am sincerely and deeply grateful to Professors Eugénio Oliveira and António Castro for the supervision of this research study. The former, coordinator of a research group on distributed artificial intelligence in the LIACC Lab and a doctoral programme in informatics, always shared the scientific expertise accumulated through the years, providing invaluable hints that led to a more substantiated MSc dissertation. The latter, also researcher in the LIACC Lab and software engineer at TAP, was the link between academia and the airline company, conveying the otherwise inaccessible business knowledge and establishing the foundations of this applied study.

Another word of appreciation goes to TAP, not only to some of its collaborators, who actively took part on this study by providing interviews and other forms of guidance, but also to the whole infrastructure, which allowed us use real operational data on our simulations.

Into a more technical arena, it is fundamental to recognize the importance played by the *Brahms* Discussion Group in the implementation stage of our simulations, namely through Maarten Sierhuis and Ron van Hoof. The former, researcher at the NASA Ames Research Center and Professor at the Carnegie Mellon University, shall be given credit for the creation of *Brahms* as part of his PhD thesis. The latter, software engineer at NASA, is responsible for the continuous development of the multi-agent platform. They were both very supportive, always showing interest in the ongoing project and not losing patience with beginners questions.

Finally, I am also thankful to the *MASDIMA* research and development team in the LIACC Lab for the help in comprehending TAP operational database fields and David Humphrey, top contributor to the *Processing.js* project, for the guidance provided through the IRC Channel of the project.

Nuno Machado

Contents

List of Figures

List of Tables

List of Listings

Abbreviations

ACMI	Aircraft, Crew, Maintenance and Insurance
AMS	Aircraft Movement System
AOCC	Airline Operations Control Centre
AOG	Aircraft on Ground
ATC	Air Traffic Control
ATFM	Air Traffic Flow Management
BDI	Beliefs, Desires, Intentions Architecture
BRAHMS	Business Redesign Agent-Based Holistic Modeling System
CN	Connection Network
CSV	Comma-separated values
DSS	Decision Support System
HCC	Hub Control Centre
IATA	International Air Transport Association
GPU	Ground Power Unit
MAS	Multi-Agent System
MEL	Minimum Equipment List
MER	Mars Exploration Rover
NASA	National Aeronautics and Space Administration
OCC	Operations Control Centre
TAP	*Transportes Aéreos Portugueses*
TBN	Time Band Network
TCP	Transmission Control Protocol
TLN	Time Line Network
ULD	Unit Load Device
UML	Unified Modeling Language
WEKA	Waikato Environment for Knowledge Analysis

Chapter 1

Introduction

This chapter sets the frame for the presentation of introductory and general information related to the dissertation. It starts by providing a broad picture of the research study, exposing its foundations and establishing its objectives and expected results. In the end, a brief section details the overall structure of this report.

1.1 Overview

Over the last 40 years, the miniaturization and increasing affordability of electronic and automated systems made computers ubiquitous in homes and offices. Nowadays, businesses from different fields and all over the world use silicon-based systems to record data, assist decision processes, automate tasks, among other uses.

If on one hand technological systems play an important role promoting business growth and competitiveness, on the other hand human capital still takes part in the majority of decision processes.

The previous observations are transversal, in different extents, to every business sector. The airline business is no exception and it currently relies on a number of computerized systems to plan flight activity, track aircrafts and crew members and record operational activity.

In an airline company, as in any other contemporary business, human collaborators live together with machines creating an interconnected organizational structure aiming at carry out a set of operations. While the established organizational structure might serve its purposes, it does not mean that analyzing its intrinsic aspects, such as communication flow, number of collaborators and decision making, will not allow for tuning up the working practices and optimizing the output.

The academia dissertation that will be described along this document lies in the scientific fields of Operations Management and Artificial Intelligence and departs from the observations drawn in the previous paragraphs. It starts from the empirical knowledge of the operations carried out by an airline company, then it delves into historical data analysis and work practice simulation, concluding about business processes and work practice improvement.

1.2 Motivation

Falling into different knowledge fields, Operations Management and Artificial Intelligence, provides different incentives to develop this research study.

From an Operations Management perspective, having a clear understanding of how processes take place inside a business translates a way of improving the efficiency of those processes. The study of heuristic decision making opens the door to a more structured and automated choice analysis and selection. All this analytics may be later implemented or incorporated in software leading to determinant decision support systems.

It worths explaining why an airline company was chosen as an Operations Management case study. The airline business spans far broader than the usual traveler's view of checking-in, safety procedures, flight to destination and baggage reclaim. Actually it comprises a large number of internal processes, a great network of agents and very complex information flows. All these features translate a very rich and semi-computerized Operations Management scenario where automated systems play a role in gathering, displaying and logging data but the reasoning over decision parameters is still performed by humans.

Still on the Operations Management case study, it is relevant to emphasize the achieved cooperation with TAP, the major portuguese air carrier. At first, having someone profoundly familiar with airline operations co-supervising this dissertation allows for continuous monitoring and feedback over the work performed, drastically increasing the chances of a more targeted and realistic research study. The partnership also lead to interviews with the most relevant company employees undertaking decision roles, the Aircraft Manager and Crew Manager. Last but not least, TAP kindly provided real operational data from its databases regarding scheduling, flights and delays. This data might be regarded as an important stimulus to perform real data analytics that may help the company not only reduce their costs but also provide a better service, namely, increase customer satisfaction.

As previously stated, Artificial Intelligence is also heavily related with this research study in the sense that it provides valuable knowledge to classify and simulate operational data and processes promoting the uttermost understanding over the operations management taking place in an airline company.

In this case, the challenge is to bring and apply Artificial Intelligence research topics, such as Supervised Learning techniques, Multi-Agent Systems and Simulation concepts to the Operations Management field, namely related to the airline business.

In the next section, Objectives, it will be thoroughly described how both fields of knowledge, Operations Management and Artificial Intelligence will be articulated but for now, it worths mentioning the usage of a powerful but still emerging simulation engine combining a Multi-Agent System with Production Rules geared to model work practices systems, such as an airport and associated business processes.

As mentioned, if the adoption of a purely scientific and academic simulation engine, with a short user community, requires an extra effort in order to understand new paradigms and learn new programming languages it also motivates an educational approach by trying to transmit gathered experience to community newcomers.

To conclude, a last word goes to the engineering challenge of interacting with the simulation engine, a closed and command-line software tool, through a browser window in a web oriented environment. In order to accomplish this, recently released technologies related with data visualization and browser communication will be used. Learning and applying such emerging techniques, that will play an important role in tomorrow's Software Engineering is always stimulating.

1.3 Objectives

Being an Artificial Intelligence research project focused on Operations Management analysis, it comprises a broad set of minor goals targeting the general objective summarized by the dissertation title: study the impact of the organizational structure on airline operations management.

Starting from the aforementioned general goal, it worths explaining the concepts involved. As briefly stated, the organizational structure of a company is the network of entities collaborating and contributing to serve one common aim. Following this, the organizational structure seen on TAP, the case study airline company, includes human employees and computerized systems performing activities and communicating with each other. They are distributed across several locations, occupying different facilities and their processes aim at assuring that flights depart and arrive according a previously defined schedule.

Back to the research general goal, it intends to provide a way of mimic, visualize and evaluate different hierarchical and collaborative work scenarios among entities within an organization. In the end, the research study is expected to appraise different operational models, proposing changes that lead to a more efficient organization.

In order to accomplish such thriving goal, the research will materialize a parametric simulation of a real-world organization, the major portuguese air carrier, comprising the following features:

- **Human entities** — The most relevant company employees will be modeled as agents; they all play a role in the simulation, usually regarded as activities.
- **Computerized systems** — Represent computer or other automated systems; in a similar fashion to humans, they will be also modeled as autonomous agents that react and decide upon requests.
- **Activities** — Might translate communication flow between agents, the interaction with a system or complex reasoning processes that end in decision making; they usually consume time.
- **Input/Output** — In order to simulate the airline company as faithfully as possible, real data will be used; meaning that the simulation will be fed with a real flight schedule and verified anomalies and it is expected to produce results as similar as in real life.

This initial simulation, that intends to mirror an actual airline company, will allow to quantify the amount of time required to operate the business across a given period of time. Having a model of what happens in a real world scenario and being able to gather a set of metrics to evaluate that scenario, opens the door to perform a fair comparison between different models.

Following this, the second major goal of this research study is to model second simulations by altering the current organizational structure and propose changes that prove to raise its effectiveness.

Concerning this second goal, it will require, for instance, to add or remove human employees or change the way they act. Then, after running the new simulation model and gathering its output, if the cost is inferior, the changes might be regarded as an operational management improvement.

Although the aforementioned goals are well defined, they hide some challenges that need to be addressed and, as such, might be seen as intermediate goals.

First, the usage of real operational data provided by TAP mandates a deep analysis and filtering of the supplied database dumps. Given the large number of attributes and records, it is relevant to understand the meaning of every database column, how the records relate to each other and detect redundant fields. The file cleanup and data analysis carried out on this step will lead to better data manageability at the simulation level and establish the foundations for further decision making studies.

Second, in order to better simulate an operations management scenario, it is required to have an extensive understanding on how business processes take place inside the airline

company. While the cooperation with TAP will provide interviews and supervised modeling, that is definitely insufficient when facing the complexity and multitude of anomalies taken care by airline companies. In order to overcome this issue a thoroughly analysis of the real operational data, including the use of machine learning techniques, may boost the comprehension of the decision making processes happening at some stages of the simulation.

Last but not least, some minor goals are related to the usage of certain software tools. The most prominent is the development of the simulation using Brahms, Business Redesign Agent-Based Holistic Modeling System. As the name implies, it consists in a multi-agent system that features a high-level programming language suited to model and simulate business processes. Closely related to this minor goal is the development of an accessible visualization of the simulation.

1.4 Expected Results

The last section conceptually presented the objectives of the dissertation using a *top-down* approach. First, we detailed the broader, higher, main goals and then attention was given to more specific, lower challenges that need to be overcome in order to reach the former.

This section extends the previous one, by listing the results expected to be achieved at each step and by the end of the ongoing research. In order to do so and aiming at clarify the concepts presented in Section 1.3, this section follows a *bottom-up* approach depicting, through simplistic examples, how the research will be conduced. On chapter 3, one may find an in-depth analysis and description of the full airline operations control case study.

1.4.1 Operational Data Analysis and Decision Making Improvement

As mentioned, one of the intermediate goals of the study it to understand the business and decision making processes taking place at an operational control level of an airline company. After interviewing someone in charge of taking decisions one may get the following hypothetical excerpt:

> "... if there is a problem with an aircraft that will cause a delay, we look into the flights that will depart afterwards and, if any, we change aircrafts. Otherwise we delay the departure."

By looking into the above quotation, it seems straightforward to implement such decision making procedure. Nevertheless, it worths paying attention to the database dumps with operational data provided by the airline company. Table 1.1 depicts a simplified excerpt showing some hypothetical operational data.

Table 1.1: Hypothetical operational data.

FLT_NBR	SCHD_DEP_DATE	AIRC_REG	ACTL_DEP_DATE	DEP_DLY_TYP
452	15/02/2010 14:15	CSTMU		
894	15/02/2010 14:30	CSTPB		
972	15/02/2010 15:30	CSTPG		
452	15/02/2010 14:15	CSTMU	15/02/2010 14:20	831
894	15/02/2010 14:30	CSTPG	15/02/2010 14:50	831
972	15/02/2010 15:30	CSTPB	15/02/2010 15:30	

At this point, and as mentioned early, efforts should be made in order to understand how the company logs its operational events. For instance, the column names of the table 1.1 are not self-explanatory and it requires further inspection to comprehend the meaning of those abbreviations:

- **FLT_NBR** — Flight number.
- **SCHD_DEP_DATE** — Scheduled departure date.
- **AIRC_REG** — Aircraft registry, the univocally identifying aircraft label.
- **ACTL_DEP_DATE** — Real departure date.
- **DEP_DLY_TYP** — Departure delay type code.

Table 1.1 was purposely slitted in two parts, the first three rows correspond to scheduled flights; the second half represents what really happened. For instance, the flight 452 was planned to depart at 14:15 and has the CSTMU aircraft assigned, but it left airport 5 minutes later, caused by a 831 delay.

In order to clearly present the expected results at this stage, let's assume that all the flights listed on table 1.1 depart from the same airport and consider *831* the flight delay type code corresponding to *Aircraft Defects at Home Base*.

Comparing the interview statements with the logged operational data shows different resolution methods to the same problem. Corroborating the interview is the flight 894. It suffered a 831 delay, which means that aircraft CSTPB was having problems, so it was replaced by CSTPG from flight 972, departing afterwards. Contradicting the interview is flight 452. It also suffered a 831 delay but the aircraft wasn't replaced, while it could have been.

As stated in the previous section, the logged data will be cleaned up, analyzed, preprocessed and classified using Artificial Intelligence algorithms. During this phase, one is expected to tune up the empirical decision making procedures suggested by the interviews. For instance, figure 1.1 illustrates two decision trees. The first, on the left, was obtained solely by contextually interpreting the interview excerpt; the second, on the

right, corresponds to the application of the *C4.5 algorithm* to the database records, using the delay time as an estimate.

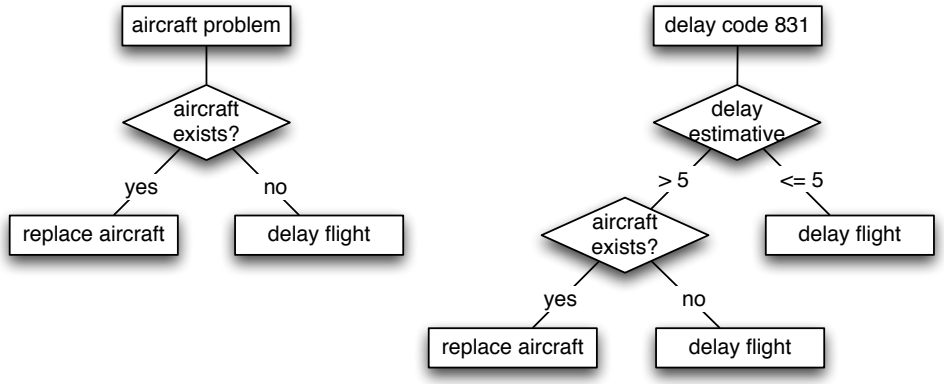

Figure 1.1: Hypothetical decision tree improvement.

To conclude, it is very important to emphasize that the example presented till now is an hypothetical and over-simplified one. While the operational data analysis to carry out along this research study assumes an extreme importance, expectation should be kept moderate because when dealing with human decisions, many random and unreported factors come into play. It worths remember that, even the same person, facing the same inputs, chooses distinct paths at different points in time.

1.4.2 Organizational Structure Performance Assessment

In the Objectives section, it was explained that one of the major goals to accomplish along the research study herein documented is to assess different organizational structures related to the airline business. The previous subsection, about operational and decision making analysis, is seamlessly interwoven with this subject as it will provide much of the knowledge required to model the simulation behind such assessment.

Continuing with the *bottom-up* approach proposed in the beginning, and using an example as a mean to clarify the goals presented beforehand, this subsection intends to show the final expectations concerning the evaluation of different operational workflows.

To determine the most relevant business processes related to the airline operations control, and in a similar fashion to what was divulged above, interviews will be used. An excerpt of an hypothetical inquiry might look like:

> "There are several employees on the ground of the airport, mechanics, cleaning personnel, catering,... They are usually split into different groups, Maintenance and Passenger Services. When something goes wrong, for instance, an aircraft didn't pass all the mandatory mechanical checks, someone from the maintenance team alerts via radio the Hub Control Centre supervisor. The

later then fills a form on the Aircraft Movement System making the Operational Control Centre personnel aware of the problem."

Even if the above quotation is fairly abbreviated it contains lots of unfamiliar information. Besides the human agents identified, such as supervisors or mechanics, it also aggregates those agents into groups, Maintenance services and Operational Control Centre personnel. Computerized systems, like the Aircraft Movement Systems are also made known, as some locations, Airport Ground, Hub Control Centre and Operational Control Centre.

More important than the entities identified in the excerpt is the set of activities and information flow ascertained. These relations between human agents, systems and work practices are the foundation of an organizational structure and will be the building blocks for the simulation.

In order to evaluate a collaborative workflow, some metrics are needed. A natural variable might be the time taken to perform the set of tasks required to complete a process. This way, inquiries will be made to gather statistics about the amount of time spend on each activity.

The better way of displaying workflows is by means of sequences and/or activity diagrams. Following this, figure 1.2 depicts the agents, systems, activities and information flow associated with the quotation above.

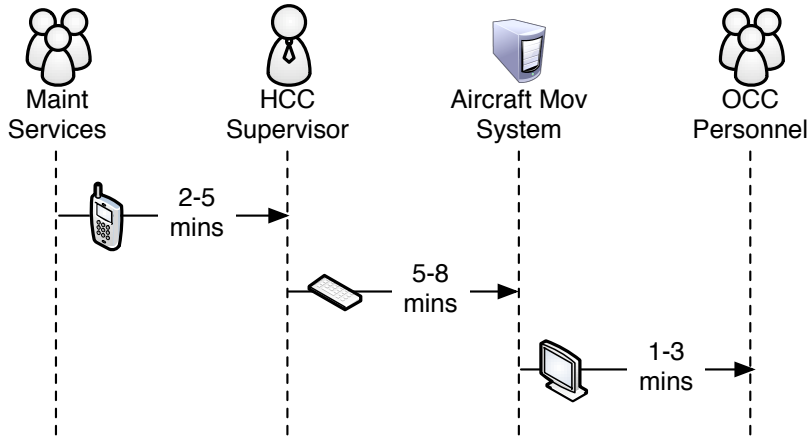

Figure 1.2: Example of an airline operational workflow.

The icons of a radio, keyboard and computer screen illustrate the activities played by the agents and by their side, is a roughly hypothetical estimation of the time intervals, in minutes, that those activities may take. Summing up the minimum and maximum values of the activities, it is possible to assert that the considered business process will consume between 8 and 16 minutes.

Without forgetting that the main goal of the research is to assess different organizational structures in order to propose changes that lead to business process improvement, figure 1.3 represents the same business process with noticeable modifications.

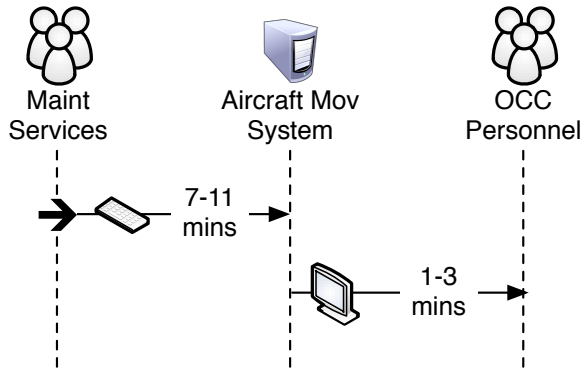

Figure 1.3: Hypothetical airline operational workflow improvement.

When reengineering business processes one should keep in mind which are the inputs and the outputs of the work flow. Failing to do that, and introducing changes at the start or at the end of the sequence, will definitely originate a different business process, something undesirable. In the example proposed, the objective is to notify the OCC personnel about an aircraft problem, so the Maintenance Services and the OCC team must remain untouched.

As it is possible to observe, the organizational structure was altered by removing the HCC Supervisor. This way, the interaction between the Maintenance Services and OCC personnel was greatly simplified by dropping the radio communication and moving the AMS terminal input to the Maintenance Services. Concerning the time spent to complete the steps, and because the Maintenance Services work at the Airport Ground, where the aircrafts are parked, while the AMS computer is inside the Hub Control Centre, it just dropped from 7-13 minutes to 7-11 minutes. This new estimation took into account the time required for the mechanics to reach the AMS terminal.

Assessing this last organizational structure would give a total operational time between 8 and 14 minutes, around 9.1% performance increase on average. It would be possible to raise this amount if, for instance, it was introduced some sort of mobile computing solution, such as a laptop or smartphone, that would allow data input on AMS anywhere.

Finally, it is important to recall that the above business process was just a fairly simple example to illustrate and clarify what is expected to achieve when speaking about organizational structure assessment and optimization. On chapter 3, where the airline operations management scenario will be profusely described, business processes will be far more complex, involving a higher number of entities.

1.5 Document Structure

Throughout this chapter it was presented a comprehensive overview of the research study. By providing simplified and hypothetical examples that better illustrate the methodologies to be used and the outcomes to be achieved, emphasis was given to the goals and expected results.

This chapter is not finished until it presents how this document will be structured and how the next chapters will be articulated with the subjects brought along this introduction, a matter to be covered in the next paragraphs.

Chapter 2, *State of the Art*, follows this introduction and, in broader terms, it will present the theoretical and foundational scientific knowledge behind the majority of methodologies and techniques to be used along the research study. To the best of our knowledge we were the first to simulate the Airline Operational Control Centre organizational structure in order to study its impact in airline disruption handling. Because of that it is difficult to compare our approach with others. Nevertheless, the *State of the Art* will be divided into three unrelated sections. The first, *Airline Operations Control*, delves into the world of airlines companies, detailing some of the major issues they face nowadays, such as the Scheduling Problem and the Disruption Management Problem. Next, *Machine Learning and Data Classification* will appear as the second section integrating Chapter 2 and will present the underlying theory behind the analysis of airline operational data and decision making improvement. Finally, the third and last section, *Work Systems Design and Simulation*, is dedicated to expose some emerging Multi-Agent Systems supporting human-centered features in order to model and simulate work practices.

Airline Decision Making Improvement and Organizational Structure Assessment, is the title of chapter 3, dedicated to present the most practical and targeted concepts involved on the research study. As the name implies, and after a brief introduction, the content will be split into three sections. First, in a section named *Airline Empirical Observation*, it will be exposed the experimental knowledge about airline operations, founded on interviews and direct inquiries. The second section, *Operational Data Analysis and Decision Making Improvement* will describe the methodologies and approaches used in analyzing the large database records provided by the airline company and how they induce systemic decision making improvements. Next, a section entitled *Organizational Structure Performance Assessment* will provide insights into airline work systems modeling using human-centered and academic multi-agent system and how it can be used to collect a set of metrics behind organizational structure assessment. This last section also briefly exposes the development of a visualization module intended to raise airline workflow understanding.

Chapter 4, *Results and Analysis*, starts with a comprehensive section entitled *Experiments* dedicated to thoroughly describe input data sets, organizational scenarios, output

appraisal metrics, among other relevant and conceptual data information. Then, two other sections come up, each one related to the major goals of the Dissertation. The first, *Operational Data Analysis and Decision Making Improvement* presents the results of the machine learning techniques, such as output decision trees, always establishing a comparison with the human classification counterpart. The second, focused on the *Organizational Structure Performance Assessment*, depicts the aggregate evaluation of different workflows using charts associated to the chosen metrics.

Finally, chapter 5, *Conclusion and Future Work*, links the objectives proposed throughout this chapter with the results obtained and listed on chapter 4. After wrapping up the work performed, it also exposes the limitations faced along the research and points out other fortuitous achievements not initially planned. To conclude, it suggests further additions to the work carried out.

Chapter 2

State of the Art

This chapter intends to describe the current level of development of the scientific fields related to the research study herein documented. To that purpose, it is divided into three distinct sections covering the knowledge areas visited along this study. The first one will present an overview about the Airline Operations Control Centre structure and the Disruption Management problem. Next, it follows a theoretical exposition on the Machine Learning discipline, mainly focused on Supervised Learning and the underlying algorithms. Finally, the third section will start by introducing some human centered computing methodologies intended to modeling and simulating work practices, a topic closely related to Multi-Agent Systems.

2.1 Airline Operations Control

When it comes to the airline business, scientific research is usually focused on a narrow range of topics mainly associated with scheduling problems, extensible to other fields, and the Disruption Management problem, a less familiar but growing on interest subject commonly connected with operational scenarios. Any literature concerned with Disruption Management in the airline field would be incomplete without defining and exposing the Airline Operational Control Centre Organization.

2.1.1 Airline Scheduling Problem

Some months before the departure date of a flight, a sequential planning approach takes place. There are two distinct stages when it comes to airline scheduling: long-term and short-term [KK04]; and this process may be seen from three different dimensions: aircraft, crew and passenger.

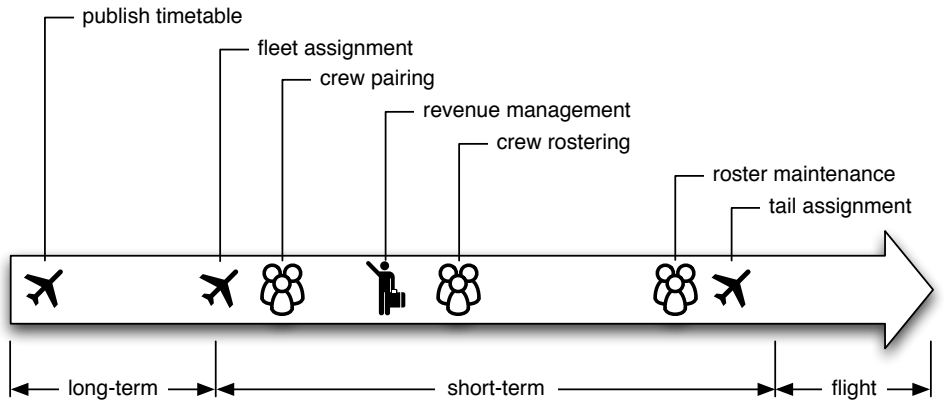

Figure 2.1: The airline scheduling process (adapted from [CO10]). Aircraft, crew and passenger views are signaled with different icons.

Figure 2.1 illustrates the Airline Scheduling Problem [Gro09]. First, the flight schedule is determined, based on forecasts of passenger demand, available slots at the airports and other relevant information. This phase determines a public timetable triggering the revenue management phase, where tickets are sold at variable prices, in order to maximize profit.

Thereafter, specific types of aircraft are assigned to individual flights in the schedule, and sequences of flights are generated within each fleet (fleet assignment and aircraft routing). This step defines the number of seats on each flight.

Next, starts the crew scheduling procedure where flight crew and cabin crew are assigned to all flights based on the already determined aircraft plan. Each pairing (crew duty periods) starts and ends at the same crew base and has a typical length of three–four days. Afterwards, pairings are grouped to form personnel rosters (assign crew members to pairings), which are lines of work typically for 14 days or one month, including rest periods, vacations and training.

Finally, physical aircrafts from a given fleet are assigned to flights in the tail assignment process.

During the Airline Scheduling Process, several restrictions and rules have to be considered. For aircrafts, there are rules on maintenance, differences between aircraft types, airport characteristics, etc. For crew, rules on flying time, off-time, etc. must be considered.

The Airline Scheduling Problem is taken care in the planning department of the airline company and its main goal it to maximize the airline operating profit. As mentioned it spans from months to some days before the flight day, time when the schedule is handed over to the operations control centre (OCC).

2.1.2 Airline Operational Control Centre Organization

The main role of the Airline Operational Control Centre (AOCC) is to monitor the conformance of flight activity according to the previously defined schedule (see subsection 2.1.1). The occurrence of some unexpected events might prevent operations to take place as planned, such as aircraft malfunction, crew delays, crew members absence, etc.

Following this, the AOCC is a human decision system composed by teams of experts specialized in solving the described problems. Teams act under the supervision of an operational control manager and their goal is to restore airline operations in the minimum frame and at a minimum cost.

According to Castro [Cas08], there are three main AOCC organizations:

- **Decision Centre**: the aircraft controllers share the same physical space. The other roles or support functions (crew control, maintenance service, etc.) are in a different physical space. In this type of Collective Organization all roles need to cooperate to achieve the common goal.
- **Integrated Centre**: all roles share the same physical space and are hierarchically dependent of a supervisor. For small companies we have a Simple Hierarchy Organization. For bigger companies we have a Multidimensional Hierarchy Organization. Figure 2.2 shows an example of this kind of AOCC organization.
- **Hub Control Centre**: most of the roles are physically separated at the airports where the airline companies operate a hub. In this case, if the aircraft controller role stays physically outside the hub we have an organization called Decision Centre with a hub. If both the aircraft controller and crew controller roles are physically outside the hub we have an organization called Integrated Centre with a hub. The main advantage of this kind of organization is to have the roles that are related with airport operations (customer service, catering, cleaning, passengers transfer, etc.) physically closer to the operation.

As mentioned, figure 2.2 shows the traditional Integrated Operational Control Centre. As previously stated, the AOCC is composed by groups of workers, each one with its own responsibilities. They must report their activity to a Supervisor, translating a two-level hierarchical system. Figure 2.2 also represents the activity time-window of the AOCC, it starts 72 to 24 hours before the day of operations and ends 12 to 24 hours after.

The roles more common in an AOCC are, according to Kohl [KK04] and Castro [Cas08]:

- **Flight Dispatch**: prepares the flight plans and requests new flight slots to the Air Traffic Control (ATC) entities (FAA in North America and EUROCONTROL in Europe, for example).

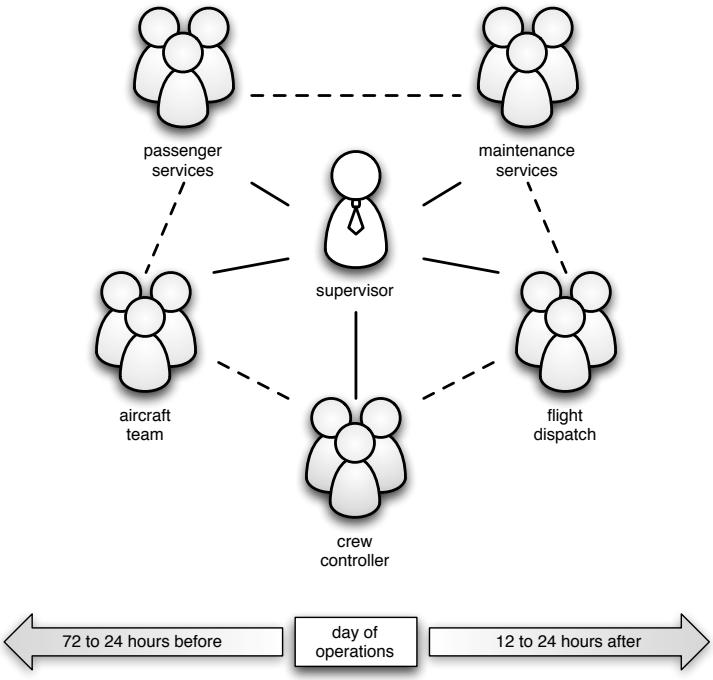

Figure 2.2: Integrated airline operational control centre (adapted from [CO10]).

- **Aircraft Control**: manages the resource aircraft. It is the central coordination role in the operational control.
- **Crew Control**: manages the resource crew. Monitors the crew check-in and check-out, updates and changes the crew roster according to the disruptions that might appear during the operation.
- **Maintenance Services**: responsible for the unplanned maintenance services and for short-term maintenance scheduling. Changes on aircraft rotations may impact the short-term maintenance (maintenance cannot be done at all stations).
- **Passenger Services**: decisions taken on the AOCC will have an impact on the passengers. The responsibility of this role is to consider and minimize the impact of the decisions on passengers. Typical this role is performed on the airports and for bigger companies is part of the HCC organization.

2.1.3 Disruption Management

Disruption Management [KK04], also known as Operations Recovery, is the process carried out by the Airline Operational Control Centre when an unexpected problem prevents a flight to operate as planned.

The first overview of the state-of-the-practice in operations control centres in the aftermath of irregular operations was provided by Clarke [Cla98]. In his study, besides an

extensive review over the subject, he proposes a decision framework that addresses how airlines can re-assign aircraft to scheduled flights after a disruptive situation.

Currently, the most thoroughly analysis of the discipline is presented by Kohl et al. [KLL+07] where their conclusions are supported by the DESCARTES project, a large-scale airline disruption management research and development study supported by European Union.

Other authors propose more general perspectives regarding disruption management. Yu and Qi [YQ04] analyze airline disruption management from different angles: crew and aircraft recovery; and applied to other fields as well: machine scheduling and supply chain coordination. Given the large scope of their work, airline operations recovery are not particularly detailed.

On the other hand, Ball et al. [BBNO07] give insight into the infrastructure and constraints of airline operations, as well as the air traffic flow management methods and actions. Simulation and optimization models for aircraft, crew and passenger recovery are also discussed. Furthermore, the authors give an excellent survey of the airline schedule robustness as a proactive alternative to recovery, including model descriptions and a literature review.

From the mentioned studies it is clear a tendency to consider the disruption management problem as twofold: aircraft recovery and crew recovery. For each type of recovery several solution approaches were proposed based on different methodologies.

Before presenting some research works related operations recovery, it is important to list and briefly describe the methodologies used on those works [CLLR10]:

- **Connection Network (CN)**

  Is is an activity-on-node network, where flight legs correspond to nodes in the network and connections between flight legs correspond to directed edges (arcs) between the nodes. A flight leg is given by its origin, destination, departure time and date and arrival time and date.

- **Time Line Network (TLN)**

  In this case, the purpose is to represent the possible schedules in a natural way from the time-and-station point of view, which is not possible when using a CN. A time-line network has a node for each event, an event being an arrival or a departure of an aircraft at a particular station. Time-line networks are activity-on-edge networks, where directed edges correspond to activities of an aircraft, and schedule information is represented explicitly by the event nodes.

- **Time Band Network (TBN)**

  Proposed by Arguello [Arg97] in order to model the aircraft schedule affected by disruptions, and is used in the context of aircraft recovery. The network can be

constructed dynamically as disruptions occur, for a certain recovery time period. There is a set of station-time nodes and a set of station-sink nodes. A station-time node represents activities at a particular airport aggregated within a certain discrete time interval, a time band. The time label of a station-time node corresponds to the availability time (the arrival time plus the turn-around time) of the first available aircraft in the time band. A station-sink nodes represent the end of the recovery period at each station. The edges in the network represent the flights.

From the methodologies described, all of them might be used on the scheduling or on disruptions management but only the last, TBN, was created to be used in the last scenario.

The pioneer researchers on aircraft recovery are Teodorovic and Guberinic [TG84], and their study is extended by Teodorovic and Stojkovic [TS90, TS95]. Given the small size of the problem instances, the solution proposed by these author, among others, are to a larger extent based on the original planning models.

Other works formulate the aircraft recovery problem as a minimum cost network flow problem and use network flow algorithms to solve it. The studies from Mathaisel [Mat96] and Yan and Yang [YY96] are examples that illustrate this approach.

While the vast majority of the publications use integer programming solution methods to solve the aircraft recovery problem, the most recent works apply some metaheuristics to the problem, such as described by Anderson [And06] and Liu et al. [LJLT06, LJC08].

Moving to crew recovery, the majority of publications formulate the crew recovery problem under assumption that the flight schedule is recovered before the crew rescheduling decisions are made, thereby following the hierarchical structure of the disruption recovery in practice. These publications include Wei et al. [WYS97], Guo [Guo05] and Nissen and Haase [NH06].

For instance, from the list of authors presented in the last paragraph, Wei et al. [WYS97] model the crew pairing repair problem as an integer multicommodity network flow problem on a connection network. The challenge is to repair the pairings that are broken and the objective is to return the entire system to the original schedule as soon as possible while minimizing the operational cost.

Something interesting about Nisses and Haase [NH06] research is its founding on European reality. They propose a duty-based formulation for the crew recovery problem, which is especially well suited for solving the crew disruption for European airlines, as these, contrary to the North American airlines, employ fixed monthly crew rates, which should be taken into consideration when solving a crew disruption.

Other approaches to crew recovery might arise when the flight schedule is fixed and the crew recovery problems can be formulated as tactical crew pairing or rostering models. Other authors extend the classical crew scheduling formulation of the recovery problem

by adding a set of decision variables, which allow to cancel flight legs. This formulation is presented in Johnson et al. [JLN+94] and Yu et al. [YAS+03].

The 1994 paper of Johnson et al. [JLN+94] is the first published work regarding the airline crew recovery. The problem is formulated as a set covering problem with decision variables allowing flight cancellations, determining the number of deadheading crew on a flight leg and forcing crew to stay at base in the recovery solution. The authors consider the recovery of pilot pairings when a single flight is delayed at a single airport.

Yu et al. [YAS+03] report a successful implementation of a crew recovery decision support system CrewSolver in Continental Airlines. The system is interconnected with other systems of the airline. The optimization engine of CrewSolver uses the depth-first search procedure developed by Wei et al. [WYS97], and can generate several solutions to give the operator a flexibility to choose the most suitable recovery solution.

Finally, Castro and Oliveira [CO10] pioneer an approach that not only accounts for the aircraft and crew perspectives but also considers passengers. An implementation of an intelligent and distributed multi-agent system (MAS) represents the operations control center of an airline. MAS includes a crew recovery agent, an aircraft recovery agent and a passenger recovery agent. They use concepts of *direct* and *qualitative* cost to determine solutions for the disruption problem.

2.2 Machine Learning and Data Classification

The concept of Supervised Learning fits into an important scientific discipline generically named Machine Learning. Given the relevance of the later topic, it worths defining its purposes and the clarifying the relation with Supervised Learning.

According to [DC96], who presents a brief review of what Machine Learning includes, there are several applications for the discipline, the most significant of which is data mining. Starting from structured or even unstructured data sets, the main goal of Machine Learning is to perform systematic analysis and complex pattern recognition that lead to knowledge extraction and decision making formulations.

Tightly connected to the above definition, is the natural meaning of the discipline name, Machine Learning, an artificial system that automatically improves with experience. In more precise terms, it is said that a machine "learns" with respect to a particular task T, performance metric P, and type of experience E, if the system reliably improves its performance P at task T, following experience E [Mit06]. In this case, Machine Learning is more concerned with computational behavior evolution by means of algorithm development and continuous improvement [MA98].

Following this, the increasing popularity of Machine Learning is due to its capability of establishing relationships between multiple features on very large data sets [Alp04]. Nowadays, Machine Learning finds applications in several fields of knowledge such as

Speech Recognition, Computer Vision, Bioinformatics, Robotics, Marketing, Economics, among many others [Mit06].

Given its broad applications and with the purpose to fulfill different data set structures, Machine Learning offers a wide variety of algorithms that might be classified into groups [WF05]. Bellow is a comprehensive list containing the most relevant Machine Learning techniques followed by a short description [BD04].

- **Supervised Learning**: its goal is to generate a function $y = f(x)$ by analyzing examples of the form (xi, yi), where $yi = f(xi)$. Each input value xi is usually an n-dimensional vector, where each dimension is either discrete or real-valued. Each output value yi is typically either a discrete value or a continuous quantity. This way, it make it possible to map a set of inputs to putatively desired outputs.

- **Reinforced Learning**: examines the current state, s, of the world and then chooses an action, a. The action causes the world to change to a new state, s'. After each state transition, the computer receives a reward, $R(s')$. The goal is to build an optimal policy for choosing actions. This approach is mostly, but not only, used in Game Theory and Robotics.

- **Unsupervised Learning**: departs from a set of unlabeled data points and attempts to find its structure, a process that often involves clustering data into classes [JMF99].

As explained on chapter 1, the airline operational data to be analyzed in the early stages of the research project herein documented, has its origin in the company databases. This implies a certain level of structuring and labeling, what motivates an in-depth exposition of Supervised Learning in the next subsection.

2.2.1 Supervised Learning

Table 2.1 shows an example of a labelled data set; *feature 1* till *feature n* name the attributes (labels) of the underlying fields. Every instance in any data set used by supervised machine learning algorithms should be represented using the same set of features.

Table 2.1: Example of data with known labels.

case	feature 1	feature 2	...	feature n	class
1	xxx	xxx	...	xxx	good
2	xxx	xxx	...	xxx	good
3	xxx	xxx	...	xxx	bad

Features or attributes in Supervised Machine Learning may be continuous, such as integers or floats; categorical, like strings of characters belonging to a finite set; or binary, usually translated by a boolean.

The process of Supervised Learning must be regarded as stepped and iterative one [Kot07]. Figure 2.3 aims at illustrating it and the next paragraphs describe all the stages from data set collection to classifier output.

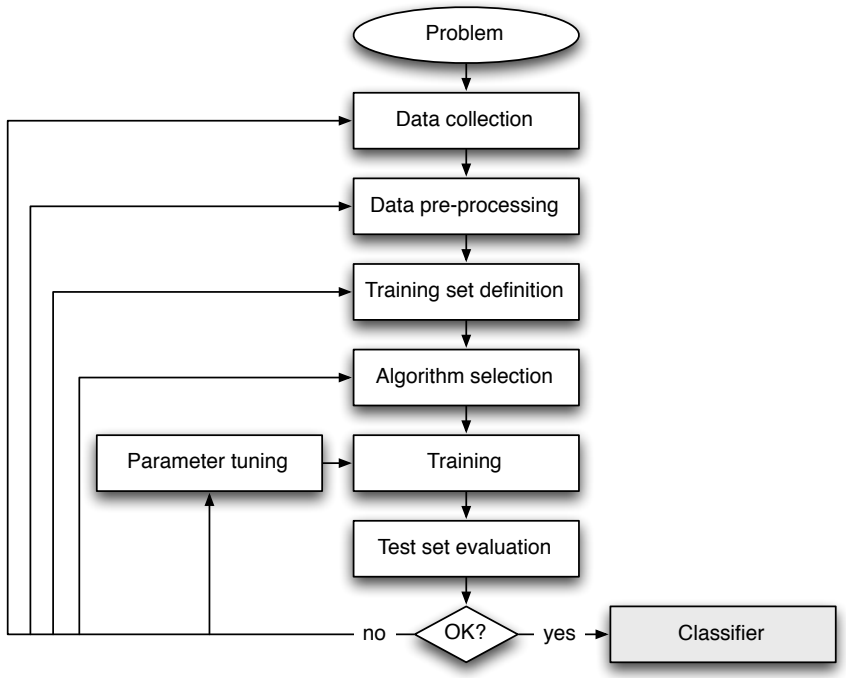

Figure 2.3: The supervised machine learning process (adapted from [Kot07]).

The first step of solving a problem by means of Supervised Learning is collecting the data set. After ensuring that it conforms to the well-structured format illustrated by table 2.1 the next step is to select the relevant features or attributes. If a requisite expert is available, then she could suggest which fields are the most informative. If not, "brute-force" may be used which means using all the information available in the hope that the right (informative, relevant) features can be isolated. Usually, having to rely on the "brute-force" workaround will introduce noise and missing feature values, making it undesirable [ZZY02].

Even with careful feature selection, the second stage of the process is data preparation and preprocessing. During this step efforts are made to handling not only missing data but also reduce noise. A number of studies have identified the techniques' advantages and drawbacks of managing missing fields, such as the research carried out by Batista and Monard [BM03], and detecting noise, like the study of Hodge and Austin [HA04]. Instance selection in the data set is an optimization problem that attempts to maintain the mining quality while minimizing the sample size [LM01]. It reduces data and enables a data mining algorithm to function and work effectively with very large data sets. A variety of procedures for sampling instances from a large data set is proposed by Reinartz

in [Rei02].

Back to figure 2.3, it worths noticing that the data preprocessing step usually overlaps with the definition of the training set, although this last step before algorithm selection and consequent training is more concerned with the effectiveness and performance of the data mining algorithm. While defining the training set, it is important to identify and remove as many irrelevant and redundant features as possible [YL04]. This reduces the dimensionality of the data and enables a faster and effective algorithm operation.

The fact that many features depend on one another often unduly influences the accuracy of Supervised Learning classification models. This problem can be addressed by constructing new features from the basic feature set, such as proposed by Markovitch and Rosenstein [MR02]. This technique is called *feature construction*. These newly generated features may lead to the creation of more concise and accurate classifiers. In addition, the discovery of meaningful features contributes to better comprehensibility of the produced classifier, and a better understanding of the learned concept.

As mentioned early, the Machine Learning discipline has many algorithms associated. This fact makes the selection of a specific learning technique a critical step. During this stage is important to keep in mind that the final classifier will map from unlabeled instances to classes, so a good way of assessing the algorithm selection is evaluating the classifier's output. Usually, this is done through prediction accuracy, the percentage of correct prediction divided by the total number of prediction.

There are several techniques used to calculate a classifier's accuracy. One technique is to split the training set by using two-thirds for training and the other third for estimating performance. In another technique, known as *cross-validation*, the training set is divided into mutually exclusive and equal-sized subsets and for each subset the classifier is trained on the union of all the other subsets. The average of the error rate of each subset is therefore an estimate of the error rate of the classifier. *Leave-one-out* validation is a special case of cross validation. All test subsets consist of a single instance. This type of validation is, of course, more expensive computationally, but useful when the most accurate estimate of a classifier's error rate is required. These and other techniques are thoroughly described in Tjen-Sien et al. [TSW00] research articles.

Recalling figure 2.3, if the error rate evaluation is unsatisfactory, we must return to a previous stage of the Supervised Learning process. Usually, a high error might translate a number of factors, such as incomplete feature selection, insufficient number of instances in the training set, a huge dimensionality of the problem, inappropriate algorithm selection or required parameter tuning [JS02].

2.2.2 Decision Trees

After dedicating some attention to the Supervised Learning process, mainly the training set definition and classifier evaluation, it's now time to focus on the classifier internals. This subsection intends to describe how a set of unlabeled instances maps to a a given class and it will expose one of the most popular classification algorithms known as C4.5.

Supervised Learning is connected to a broad range of knowledge fields, mostly with Artificial Intelligence but also with Statistics, Data Mining, Pattern Recognition, and so on. Given its multi-disciplinarily, there is a broad range of classifiers, each one relating to a different learning algorithm. This subsection is dedicated to decision trees in general, a type of classifiers that, together with rule-based classifiers, fit into the logical or symbolic learning methods. A comparison of several classifiers/algorithms might be found in Kotsiantis work[Kot07].

Decision trees are well-known as a management decision support tool that depicts a set of decisions and their possible consequences. They maybe enriched by including probabilities, resource costs, and other metrics intended to identify a set of options most likely to reach a goal.

In order to better contextualize decision trees on the Supervised Learning process, figure 2.4 is an example of a decision tree classifier for the training set of table 2.2.

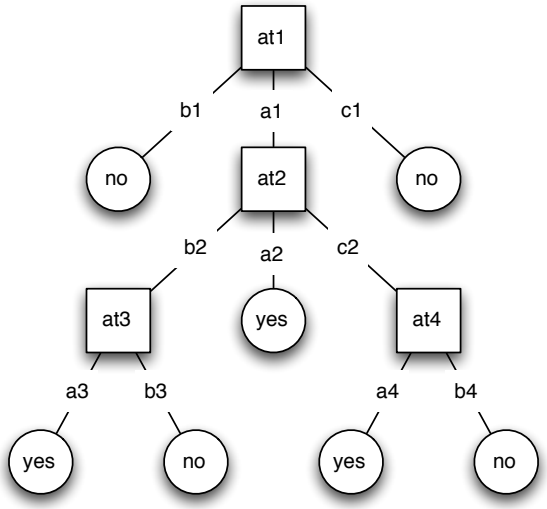

Figure 2.4: A symbolic learning decision tree.

On Machine Learning, decision trees are structures that classify instances by sorting them based on feature values. Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values [Mur98].

After applying a decision tree learning algorithm to the training set shown on table 2.2, one will get an abstract representation of the decision tree depicted on figure 2.4. Thereafter it is possible to, given a set of input attributes, achieve the corresponding conclusion. For instance, assuming the attributes: *at1* = *a1*, *at2* = *c2*, *at3* = *a3* and *at4* = *b4*; the tree would be percolated visiting the nodes *at1*, *at2* and finally *at4*, which would classify the instance as being "No". In the middle of the procedure, the node *at3* would be ignore as it was proved to be irrelevant next to the set of inputs.

Table 2.2: The decision tree training set (figure 2.4).

at1	**at2**	**at3**	**at4**	**class**
a1	a2	a3	a4	yes
a1	a2	a3	b4	yes
a1	b2	a3	a4	yes
a1	b2	b3	b4	no
a1	c2	a3	a4	yes
a1	c2	a3	b4	no
b1	b2	b3	b4	no
c1	b2	b3	b4	no

The general idea behind decision tree development is fairly straightforward. First, it is required to find the feature that promotes the highest normalized information gain. To do so, every attributed must be checked in order to determine which one best divides the training set. Once found it is possible to define a decision node, subtracting it from the feature set and recursing on the remaining sublists.

Several authors propose different methodologies for building decision trees. Since it is a procedure covering several steps, specific literature dedicated to each stage might be found. Hunt et al. [HMS66] and Breiman et al. [BFOS84] present different methods for finding the feature that best divides the training data such as the *information gain* and the *gini index*, respectively. More recent methodologies, such as the *RelieF algorithm* determine the training set division by looking into each attribute independently [Kon94].

Another topic of extensive research, mostly justified by the large size of some training sets, is decision tree optimization. There are numerous scientific articles on tree simplification an pruning. To name some comprehensive papers, Breslow & Aha [BA97] survey some methods of tree simplification to improve their comprehensibility, while a comparative study of well-known pruning methods is presented by Elomaa [ER99].

The most popular algorithm in the literature for building decision trees is the *C4.5* [Qui93]. *C4.5* is an improvement of Quinlan's earlier *ID3 algorithm* [Qui79], sporting the following characteristics:

- **Discrete and Continuous Attributes**: in order to handle continuous features, *C4.5* creates a threshold and then splits the list into those whose attribute value is above

the threshold and those that are less or equal to it [Qui96];

- **Missing Attributes**: *null* or features marked with a special "missing" character are not used in gain and entropy calculations and therefore will not take part in the respective decision tree branch;
- **Different Cost Attributes**: features might be given different costs and *C4.5* will perform the required calculations to raise the attribute gain according to its cost;
- **Pruning Trees After Creation**: *C4.5* goes back through the tree once it has been created and attempts to remove branches that do not help by replacing them with leaf nodes.

C4.5 uses the concept of information entropy which, in turn, measures the uncertainty associated with random variables. The difference in entropy corresponds to the normalized information gain, referred above as a criterion to split the training set within the same feature. This way, the *C4.5 algorithm* go after the general idea behind decision tree generation, translated by the following pseudocode [Kot07]:

1. Check for base cases (see bellow)
2. For each attribute *a*
   - Find the normalized information gain from splitting on *a*
3. Let *a_best* be the attribute with the highest normalized information gain
4. Create a decision node, *node_a*, that splits on *a_best*
5. Recur on the sublists obtained by splitting on *a_best*, add those nodes as children of *node_a*

As other algorithms, *C4.5* has a few base cases. For instance, if all the instances in the training set belong to the same class, it simply creates a leaf node for the decision tree pointing to that class. Other cases might be that none of the features provide any information gain or a previously-unseen class is encountered, on both the *C4.5* creates a decision node higher up the tree using the expected value of the class.

To conclude, although decision trees offer many advantages such as, simplicity, adaptability to several types of data and robustness on missing features, it is important to point out some of their limitations. Some decision-tree learners create over-complex trees that do not generalize the data well, an issue usually called *overfitting* [Bra07] and possibly overcome by using pruning techniques. But the greatest limitation of decision tree learning is its inability to handle XOR, parity or multiplexer problems, like medical diagnosis or protein classification. These concepts are hard to learn leading to prohibitively large

decision trees. Approaches to solve this limitation usually requires using learning algorithms based on more expressive representations, such as statistical relational learning, inductive logic programming or multi-label classification.

2.2.3 WEKA

According to the Objectives (section 1.3), the research study described along this document includes an extensive classification of real airline operational data. In order to speedup the analysis and test multiple learning approaches, third-party open-source softwares will be used. This subsection aims at present those libraries, conveniently aggregated under the project *WEKA*.

WEKA stands for "Waikato Environment for Knowledge Analysis", and it is a suite of scientific machine learning software written in Java and developed at the University of Waikato, New Zealand. Unlike other machine learning projects, the emphasis is on providing a working environment for the domain specialist rather than the machine learning expert. It provides a wealth of interactive tools for data manipulation, result visualization, database linkage and cross-validation and comparison of rule sets, to complement the basic machine learning tools.

Witten & Frank [WF05], present an exhaustive book about Machine Learning an its articulation with the *WEKA* workbench. They start by explaining how data mining algorithms work and how to evaluate the results of different techniques applied to specific problems. In the end, they cover some performance improvement approaches, including input preprocessing and output combination.

2.3 Work Systems Design

A work system involves people engaging in activities over time. Human participants might not just interact with each other, but also with machines, tools, documents, and other artifacts [Pav84]. The activities performed often produce goods, services or data.

There are two different approaches when it comes to designing or improving systems: *machine-centered* and *human-centered* [SC02].

The former is usually accomplished through a business process reengineering approach [MBCP98] based on business process flow analysis focused on work products. In this case, the functional transformations are evaluated and, in the end, the improvement will likely involve technology, such as workflow software.

The latter also takes into account how the people in the organization actually prefer to work [Ehn88, GK91]. Unlike the machine-centered approach, which neglects human

communication, collaboration, workspaces, problem solving and learning; the human-centered approach analyze human activities, work processes or tasks, comprehensively and chronologically throughout the day [Cla02].

The human-centered work system design approach is based on modeling and simulating work practices [SC02]: what people actually do, rather their outcomes. This way, it is possible to understand the effects of human behaviors in different places and times, details often omitted in a product-oriented task analysis. In the end, besides the traditional system workflow, human-centered approach might also propose some work system transformations, including different tools, resources, locations or scheduling.

Efforts to develop and formalize human-centered work system design date back to 1980 when a group of Scandinavian researchers conducted a participatory-design study involving all the stakeholders [Ehn88]. Since then, different work design approaches have been proposed, within interdisciplinary academic fields, that combine social science with a system analysis perspective [CRR91, Vic99].

The lack of theory and associated methods for developing formal models of work practice, common to the aforementioned studies, has hampered the development of a generic work system engineering approach, making the discipline like an art.

Trying to bring the human-centered work system design approach into mainstream methods for technology development, especially software engineering, Marteen Sierhuis proposed BRAHMS: a multi-agent modeling and simulation language for work system analysis and design [Sie01]. Promoting work system design as an engineering discipline would facilitate design conversations, creating a bridge between scientist, workers, managers and technology engineers.

2.3.1 *Brahms*

Brahms is a software tool to help understanding complex work system interactions, and thus providing a way for system design prototyping. It is a multi-agent modeling and simulation environment for dealing with the complex human–machine system interactions [Sie01, CSS98].

Brahms follows a holistic approach to systems modeling. By developing formal models of people's behavior at the activity level, it is possible to determine the impact of these actions on the whole system.

By relying on software models [You89], *Brahms* language describes the structure and behavior of work system defined components. The specificity of human activities, require dynamic models that shows how the system evolves over time and space.

Figure 2.5 depicts an operational method for developing a formal computational model and a simulation of a work practice for a human activity system. It also shows the relation between the four steps for using *Brahms* in a modeling effort.

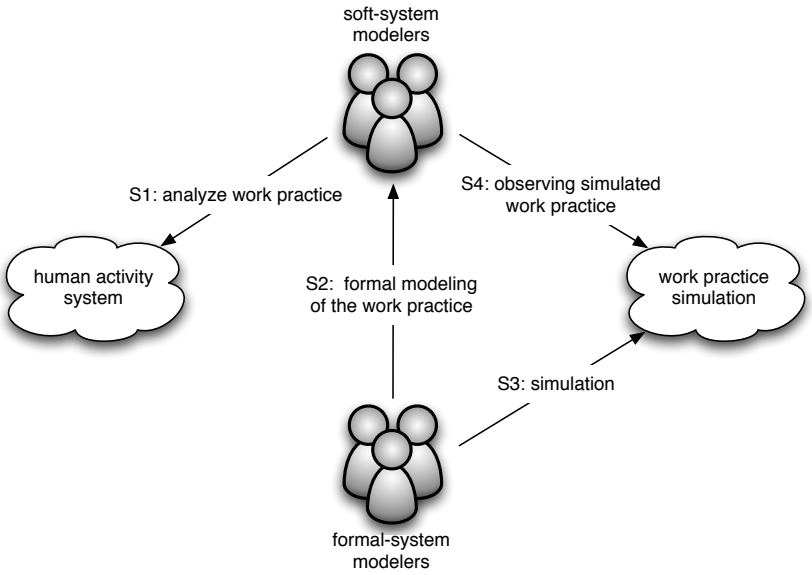

Figure 2.5: *Brahms* work practice modeling of a human activity system (adapted from [SC02]).

- **Step 1: Work practice analysis**

  Aims at observing and analyzing a human activity system [HB98, BSW93]. The goal of the analysis is to gather useful data that informally describes work practice and then to create (with the *Brahms* language) a formal model of work practice in Step 2. During this step, interviews with workers from the work system and participation of all stakeholders are recommended.

- **Step 2: Formal model of the work practice**

  Intends to formalize the informal data gathered in Step 1, conducing to *Brahms* model development. This stage is usually accomplished by people who understand the concept of agent-base modeling and simulation and often have experience in developing rule-based systems.

- **Step 3: Simulation**

  Brahms has a built-in simulator. Using the formal model developed in Step 2 as input, running a simulation outputs the modeled work practice. This stage is highly related to *Brahms* compile-simulate-debug cycle because the modeler will compile, simulate, and fix the errors in the formal model until the desired agent behavior is simulated.

- **Step 4: Observing the simulation**

  Aims at observing and investigate the work practice simulation output and compare it with the human activity system with the goal of creating a shared understanding

of the results of the work practice model and simulation. The result might suggest some changes to the formal model.

Steps 2 to 4, inclusive represent the iterative process of modeling a *Brahms* work practice simulation. After Step 4 the simulation must be reviewed with system stakeholders to verify and validate the model created in Step 2. If the model does not correctly describe the work system, then more detailed information must bet gathered in order to fix the *Brahms* model, returning to Step 2 of the process.

2.3.2 Developing a Model of Work Practice

In general, *Brahms* models represent work with much more detail than business process models but somewhat less detail (and far more broadly) than cognitive models. Considerable effort is devoted to modeling objects (for example, fax machines) and computer systems, with which people often interact to accomplish their work. The *Brahms* language and simulation engine relates several levels of detail (areas and objects, groups and agents, activities and actions) and integrates different perspectives: physical, cognitive, and social [SC02].

The following list describes the model components present in *Brahms* [vHS00].

- **Agent/Groups**: represents the group-agent membership hierarchy of the people in the work system. Agents may be grouped representing formal roles and functions or be based on location, interpersonal relations or interests.
- **Object**: is a class hierarchy of all the domain objects and artifacts (tools, desks, documents and vehicles).
- **Geography**: describes areas in which agents and objects are located, consisting of area definitions (user-defined types of areas such as buildings, rooms, and habitats) and areas (instances of area definitions).
- **Activity**: the behaviors of agents and objects are expressed in terms of the activities they perform over time [Cla02]. Agent or object activities are mostly represented at the group or class level, but they are also often specific to agents and objects. Activities are inherited and blended through a priority scheme.
- **Timing/Workframes**: constraints on when the activities in the activity model can be performed are represented as preconditions of situation action rules (*workframes*) [Kon82, Sho93]. Activities take time, as determined by the predefined duration of primitive actions. Workframes can be interrupted and resumed, making the actual length of an activity situation dependent.

- **Knowledge**: agents represent the world state internally as propositions called beliefs [Hin62]. An agent's reasoning is represented as forward-chaining production rules (*thoughtframes*) that fall at group and class levels and can be inherited. Inquiry is modeled as a combination of activities (such as detecting information, communicating, and reading or writing documents) and thoughtframes. Perception is modeled as conditions attached to workframes (called *detectables*). Thus, observation depends on what the agent is doing.

- **Communication**: describes actions by which agents and objects exchange beliefs, including telling someone something or asking a question. A conversation is modeled as an activity with communication actions, either face to face or through some device such as a telephone or email. The choice of device and how it is used are part of the work practice.

Usually, the *Brahms* modeling process starts by specifying the geography groups first. Then, common objects and activities are modeled such as telephones and phone conversations, which can be easily reused or adapted across *Brahms* environment. Finally, the grain size of the simulation clock (time per tick) might be adjusted from one second or less to five minutes or more, depending on the information available and modeling purposes.

2.3.3 Human-centered Work System Design Examples

The most visible example of work systems design using *Brahms* was conduced by Sierhuis et al. [SCS03] and involved the design of a mission operations system for a proposed NASA discovery mission to the Moon with a semiautonomous rover. The work intended to produce useful insights about vehicle power consumption and its potential impact on science objectives.

The NASA mission, code-named *Victoria*, included the exploration of the south pole region of the Moon by a robotic vehicle. Its primary mission was to verify the presence of water ice and other volatiles in the permanently shadowed regions of the Moon. The NASA team decided the most efficient approach would be to use a high-speed semiautonomous rover that could traverse a long distance (several hundreds of kilometers) for a long time period (three months to a year) while gathering the necessary geological and physics data [Cab01].

Using the *Brahms* approach, Sierhuis et al. [SCS03] developed a model of the total mission operations work system during the proposal phase of the project, including a model of people's work practice in mission operations, the rover on the Moon, the information systems and people's workspaces. With *Brahms*, they were able to quantify the impact of the human work practice on the productivity of the rover on the Moon.

Currently, the tool is being researched in context with the Mars Exploration Rover (MER) mission operations, as well as other areas of space exploration, including the International Space Station, a Mars habitat and surface exploration vehicles.

MER is a campaign employing 240 highly skilled scientists and engineers who will work around the clock to manage a new space mission in just three months. NASA has never managed a planetary rover exploration mission involving so many collaborators. The days will be tightly scheduled, requiring order among the hundreds of team members.

For the 120 scientists, each day the job requires analyzing volumes of existing and incoming data to determine a set of priorities for the rovers and build plans to achieve a goal. The same number of engineers will translate the scientists' goals into instructions for the rovers and will uplink the information.

Current research involves how the *Brahms* model can be used to develop an actual workflow system for mission operations, based on the *Brahms* agent technology and models of mission operations. The *Brahms* team plans to observe mission operations at NASA's Jet Propulsion Laboratory, and is hoping to assist in designing and implementing a surface mission operation.

Chapter 3

Airline Decision Making Improvement and Organizational Structure Assessment

After presenting the goals behind the research study documented throughout this report and exposing the most relevant underlying theoretical concepts, this chapter should be regarded as the showcase for more practical and oriented matters. Here, we will detail all the available information, data, models and approaches taken along the study.

Following the sequence of objectives and expected results presented on chapter 1, and relying on a *bottom-up* approach, this chapter will first delve into the empirical observations of a real airline operations environment. Next, a section will be dedicated to explaining the fundaments and methodologies behind the analysis of real operational data coming from the airline company databases and how this analysis relates to the decision making processes observed during airline operations control. Last but not least, the last section will wrap all the previous concepts together and build models that will allow us to provide an answer to the main concern of this research study, assessing the performance of different airline organizational structures.

3.1 Airline Empirical Observation

To begin this section, and reinforcing what was stated in the motivation (section 1.2, chapter 1), it worths emphasize the collaboration with TAP, the major Portuguese airline company, as a mean to get a higher detailed and more targeted research study. This fact, combined with the lack of other studies around the topic, makes us pioneer an in-depth

investigation about the current state of airline organizational structures and associated business processes.

Given the need to start from scratch, this section aims at organizing and systematizing the knowledge that is nowhere but on the heads of those that, working on airline facilities for many years and on a daily basis, kindly accepted to provide interviews and answer direct inquiries as a mean to share their experience.

Despite the existing literature on the problems around Airline Operations Control (section 2.1, chapter 2), such as the Airline Scheduling problem and the Disruption Management problem, and because we chose to scrutinize a particular airline company, the research study herein described, first attempted to grab a general overview on the current organizational structure, business processes and decision making practices. This step was possible trough a number of informal and unstructured conversations with the airline company personnel. Right after these interviews, efforts were made to shape and formalize all the received information, an often ungrateful task given the abstract nature of concepts or the multitude of possible scenarios.

During the first stages described above, we also received the operational data files corresponding to company database dumps. In an opposite fashion to the informal concepts gathered during personnel interviews and despite its massive extension and complexity, the database provenance conferred structure and integrity to the operational data at hands.

The next section on this chapter will be solely dedicated to the analysis of the supplied data but it is important to mention that along the analysis process our awareness around the business process and decision making practices in an airline company increased, motivating direct inquiries to airline personnel in order to corroborate or clarify arisen formulations.

Following the above explanation, this section will start by presenting a broad overview on the airline business and organizational concepts. This knowledge roughly overlaps with the simulation foundations that will be dissected on section 3.3. Next, the connection and information flow between previously described concepts will be clarified by means of sequence diagrams supported on interview excerpts. With a direct correlation to the airline operations organizational structure, there will be a section dedicated to operational anomalies description and historical classification. Finally, and given the in-depth coverage that will be carried out on section 3.2, a brief word goes to the format of the database files received and how the airline company logs its activity information.

3.1.1 Airline Business and Organizational Concepts

Trying to fully describe the airline business on this subsection would be, at least, impossible. As one of the most international, fastest growing, and revenue generating business

sectors, the aviation put on practice through airline companies certainly assumes a place among the most complex and risky businesses nowadays.

As other large companies, TAP, our case study airline company, involves a lot of departments each one dedicated to its business area. For instance, Human Resources takes care of managing the personnel required to run the company; Marketing is concerned with promoting the services and attracting consumers; Information Technology department develops and implements automated systems to assist other business areas; Administration Board decides upon long-term strategies; and so on.

The big picture given above is required to emphasize that the research study herein documented was solely concerned about the organizational structure and decision making processes related to the operations control "department" of the airline business.

Assuming that the information provided on the *Airline Operations Control* portion of the *State of the Art* (section 2.1, chapter 2) is comprehensive and general enough, attention will now be devoted to the TAP case study. From the information gathered next to the airline personnel and in order to better translate the current operations organizational structure seen on TAP, figure 3.1 depicts the most relevant facilities, collaborators and computerized systems.

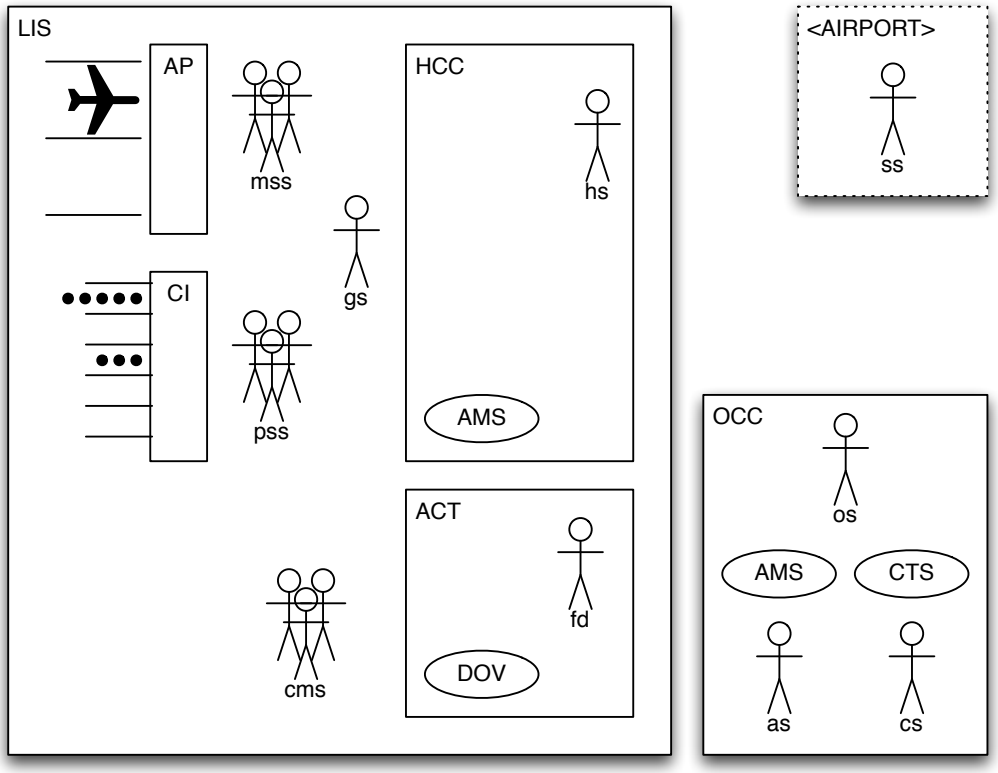

Figure 3.1: Current operations organizational structure at TAP. Extended caption on table 3.1.

When looking into the clean schemata illustrated in figure 3.1, one may find three

types of elements: rectangles, ellipses and pictorial representations of the human body. There is also an aircraft icon and small dark points symbolizing passengers, but they are both for decoration purposes only. Back to the diagram, *rectangles* represent airline facilities, usually buildings and other infrastructures; *ellipses* translate computerized systems, often a thin-client terminal; and as the name implies, the *pictorial representations of the human body* portray airline collaborators. Special attention goes for the later given their ability to form groups, corresponding to real life services or workforces.

In order to facilitate the use of language, and because of their different nature, we decided to gather facilities, collaborators and systems representations under the name "concepts". And as it is possible to observe, in figure 3.1, every concept has an associated label: facilities (rectangles), capital letters on the superior left corner; systems (ellipses), capital letters centered at the origin; and collaborators (human icons), lower-case letters beneath the representation.

Besides depicting TAP's current operations organizational structure concepts, the diagram on figure 3.1 also intends to present the spatial relations among them. For the purpose of better explaining those relations, table 3.1 lists all the labels shown on the above diagram, providing some descriptive captions for each one.

Table 3.1: Operations organizational structure concepts.

Facilities	
ACT	Areeiro Crew Terminal
AP	Aircraft Parking
CI	Passenger Check-In
HCC	Hub Control Centre
LIS	Lisbon Airport
OCC	Operational Control Centre
Computerized Systems	
AMS	Aircraft Movement System
CTS	Crew Tracking System
DOV	Flight Operations Portal
Human Collaborators	
as	Aircraft Specialist
cms	Crew Members
cs	Crew Specialist
fd	Flight Dispatcher
gs	Ground Supervisor
hs	HCC Supervisor
mss	Maintenance Services
os	OCC Supervisor
pss	Passenger Services
ss	Station Supervisor

As we will see next, the majority of the action will take place at the Lisbon Airport,

represented by the rectangle labelled “LIS”. The three-letter code “LIS” is the IATA airport code, also referred as IATA station code, designating the Lisbon International Airport. In a similar fashion, every airport around the world has an IATA code.

For a better comprehension of the schema, it worths explaining that TAP, follows the airline hub model. This means that it uses the Lisbon International Airport, as a transfer point to get passengers to their intended destination. Being part of a *hub and spoke model*[1], travelers moving between airports not served by direct flights change planes en route to their destinations. For instance, all the TAP passengers flying from Rio de Janeiro, Brazil, to Gothenburg, Sweden, will likely stop on Lisbon Airport, in order to gather other travelers moving to Gothenburg and departing from locations other than Rio de Janeiro.

The *airline hub model* adopted by TAP, makes its activity more prominent at the Lisbon Airport, allowing the company to concentrate its operations control at this airport which is generally called “base airport” or simply “base”.

Crossing the above information with the schema on figure 3.1 allows us to easily understand the dashed rectangle with the “<AIRPORT>” label. This representation intends to represent all the airports having TAP flights as departure or destination points. The “<AIRPORT>” label is just a generic placeholder for the three-letter IATA code designating the origin or destination airport. On the airports outside Lisbon, also called stations, the operations control scenario is way simpler than on the *base*, comprising just a collaborator, the Station Supervisor, whose role will be detailed soon.

Going back to the “LIS” facility, Lisbon International Airport, one may notice that a great portion of the current organizational structure stands on it. There are two distinct buildings worth reference, the first, “HCC” or Hub Control Centre is where the “hs” or HCC Supervisor plays his role, often using the “AMS”, Aircraft Movement System, terminal; the other building is the “ACT”, Areeiro Crew Terminal, where we may find the “fd” collaborator, Flight Dispatcher and the “DOV”, Flight Operations Portal, system.

At this point, it is desirable to explain that, while there was an attempt to kept the schema simple and straightforward, it somehow resembles the physical separation between different spatial locations, buildings, humans and systems. This means that the “DOV” system is directly unaccessible to any collaborator other than the Flight Dispatcher. For instance, if the HCC Supervisor needs to enter some data into the “DOV” it would have to leave the Hub Control Centre, move to the Areeiro Crew Terminal, enter the later building and then finally interact with the Flight Operations Portal.

Still inside “LIS”, one may find other facilities such as the “AP”, Airport Parking and the “CI”, Passenger Check-In. These facilities are well-known to anyone who already travelled by plane and they have two related groups of collaborators, “mss”, Maintenance

[1] Also called *hub and spoke distribution paradigm*, it is a system of connections arranged like a bicycle wheel, in which all traffic moves along spokes connected to the hub at the center.

Services, and “pss”, Passenger Services. The former is concerned with aircraft mechanical and safety checks, the later cares about passenger identification, luggage collection, security supervision, and so on. The Ground Supervisor, “gs” is somehow related to the aforementioned groups as he takes cares of other pre-flight routines, such as assure luggage and catering boarding or aircraft refueling. Last but not least, the “cms” represents the Crew Members, which are required to enter on the Areeiro Crew Terminal and report for duty through “DOV”.

Moving outside the Lisbon Airport, the TAP Operational Control Centre, “OCC” in figure 3.1, is an office somewhere in the Lisbon city, that may be regarded as a traditional Decision Centre. On the *State of the Art* (section 2.1, chapter 2) it was thoroughly described the relation between the OCC and the Disruption Management problem. Soon it will be explained how the information reaches the OCC, but for now it is sufficient to describe its concepts.

The main character found on the “OCC” is the “os”, OCC Supervisor. As the name implies, he is responsible for supervising the activity of “as” and “cs”, Aircraft and Crew Specialists, by confirming or rejecting their resolution proposals. On their turn, the Aircraft and Crew Specialists, rely on two computerized systems, namely the “AMS”, Aircraft Movement System terminal and the “CTS”, Crew Tracking System, to trigger, support and synchronize their decisions.

A final word goes to the relation between all the referred automated systems. As opposed to the human spatial limitations, the data inputted into any system flows between all the other systems, making it visible and accessible instantaneously. This means that, whenever a Crew Member reports for duty on the “DOV”, at the Areeiro Crew Terminal, this fact is immediately propagated to the “AMS” and “CTS”, making it promptly known to HCC Supervisor and Specialists (assuming they are paying attention to computer screens).

3.1.2 Operational Inputs, Outputs and in between Workflows

The previous section exposed the main concepts involved on the operational activities of TAP, our case study real airline company. These included collaborators and automated systems occupying several facilities, spread across distinct geographical locations. As any other work environment, and although a somehow static representation was provided, collaborators actively interact with each other, creating networks where information flows and collaborative decisions are made. This subsection intends to make explicit the aim of such organizational structure, presenting the most relevant business processes.

An excerpt from an interview with the HCC Supervisor follows:

> “... It’s a 24 hours job. We [the HCC personnel] spend all day here to ensure that our passengers reach their destinations on time. When a problem arrises,

> we must take action. Obviously, there are issues that are beyond our capacity and flights get delayed, people get unsatisfied, the company loses money. Fortunately, we serve around 30000 passengers a day but only a small percentage gets angry!"

The brief quotation above was the starting point for a very interesting conversation with someone that supervises TAP operational activities on Lisbon Airport. As HCC Supervisor, he leads Maintenance Services, Passenger Services and the Ground Supervisor on their roles. Grouped together they are usually referred as HCC personnel.

Following HCC Supervisor words, and matching what was written about *Airline Operations Control* on section 2.1, the goal of the whole organizational structure presented is to ensure that flights depart, and consequently arrive, according to a predefined schedule. The schedule that was advertised next to travelers, that bought airline tickets for a specific day, hour and even minute.

Taking an aircraft with hundreds of passengers to a destination several kilometers away involves a number of procedures and verifications before, during and after the flight. In the next section, a thorough list of anomalies related with those procedures and other unmanageable irregularities faced by airline companies will be presented, but for now, it is important to distinguish a couple of general cases.

- **Maintenance Procedures**: when an aircraft, coming from a foreign destination arrives into an airport, Maintenance Services, composed by mechanics, engineers and technicians perform some routine checks to the most important aircraft components, trying to assure passenger safety for the next flight.
- **Aircraft Loading**: besides the mandatory fuel, before taking off an aircraft is loaded with meals, toilet stuff, water, cabin accessories, passenger luggage, and other cargo. The collaborator responsible for monitoring proper aircraft loading is the Ground Supervisor.
- **Passenger Procedures**: passengers are required to check-in on time. Then it follows security inspection and often, customs verification, finally passengers are boarded into the aircraft, which sometimes demands airport buses or other infrastructures. Passenger Services are responsible to carry out all the aforementioned procedures.
- **En Route Verifications**: before departing, the Flight Dispatcher monitors en route problems, such as bad weather, destination airport restrictions, air-controllers capacity, natural disasters, and so on.
- **Crew Reporting**: an aircraft is only allowed to depart if a minimum number of crew members are available, a number that is related with aircraft size and seats

occupancy. In order to track Crew Members presence, they are required to report for duty on "DOV", the Flight Operations Portal.

- **Rotations**: after landing into an airport, and according to a predefined schedule, the aircraft has a limited time to reach the gates and get ready for the next flight. In a similar fashion, crew members are sometimes required to change flights following a predetermined roster. This usage of resources across subsequent flights, is called "rotation", and the "AMS" and "CTS" systems are ready to monitor its proper execution.

After the above list and keeping in mind HCC Supervisor words, some procedures might lead to problems. For instance, defects were detected in the aircraft during maintenance inspections or because of a tight schedule an aircraft had not the time to rotate and will not be able to depart on time.

As mentioned before, the "OCC", Operational Control Centre, was regarded as a Decision Centre. The reason behind this characterization lies on its responsibility to decide upon the best solutions for a given operational problem. This way, and because of its remoteness, the Aircraft and Crew Specialists must be warned to the issues arose during operations, in order to take action. On the other way around, after reaching a conclusion on which is the best way to address a problem, Lisbon Airport personnel must be notified, with the purpose of executing the decision.

In the context of this research study is therefore of essential importance to understand how the information flows, back and forth, within TAP organizational structure. This will allow us to model it with the highest fidelity, ending up with a simulation as closer to the reality as possible.

Based on the TAP personnel interviews, it were identified eight sources of anomaly reporting, and for each one a workflow diagram were depicted. Section A.1 on appendix A, *Current TAP Operational Sequence Diagrams*, contains all the eight operational scenarios triggered by the identified concepts.

The aforementioned diagrams follow the *Object Management Groups UML v2.0* directives, but taking into account their pictorial representation of actors and activities, it is important to clarify the meaning of such icons and identify the actions performed by TAP collaborators as well as the resources they use. Figure 3.2 provides a great heads-up on the interpretation of section A.1 diagrams.

Aiming at keep the figure 3.2 concise, actors were represented as general entities. The labels that appear bellow actors on appendix A diagrams match the *Computerized Systems* and *Human Collaborators* labels of table 3.1 making straightforward to track information flow. Concerning activities, they represent what collaborators actually do such as phoning to each other, reading data from systems, or reasoning over problems to reach conclusions.

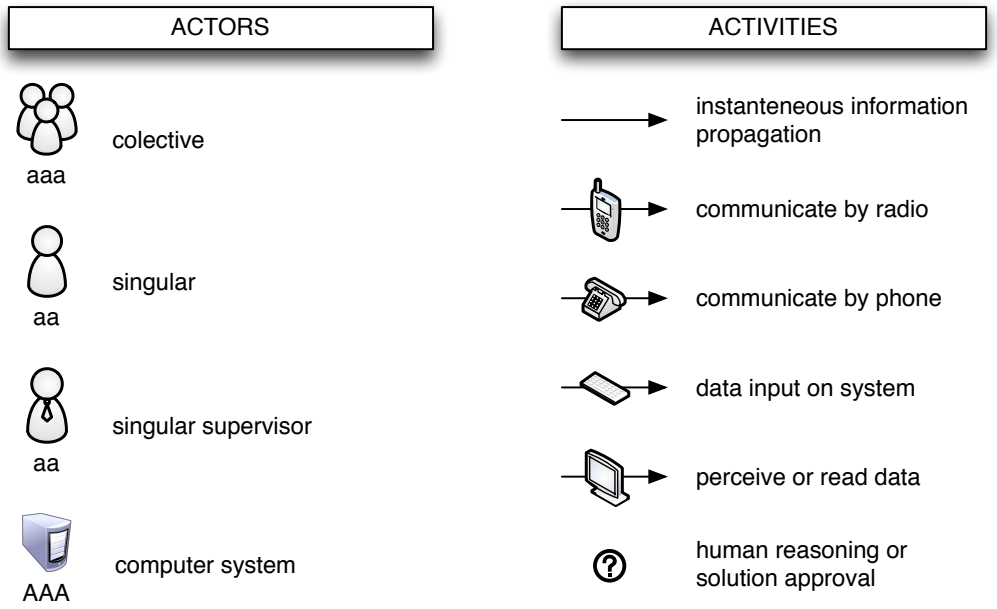

Figure 3.2: Operational workflow entities at TAP.

Describing all the eight workflow diagrams would not add value to this document as they are very similar and share the same concepts. With this in mind, we opted to choose the most complex sequence schema and thoroughly explain it, making it easier to interpret the remaining diagrams. Following this, the next paragraph is dedicated to detail the operational workflow triggered by an aircraft of crew anomaly detected by the Flight Dispatcher, figure 3.3.

As stated above, the Flight Dispatcher is responsible for monitoring en route problems, such as bad weather, destination airport restrictions, air-controllers capacity, and so on. The Flight Dispatcher is also concerned with previous flight anomalies that may threat a delay after a connection. Let's take an example given during an interview with OCC Supervisor.

> "The the aircraft CSTOL is assigned to flight 355 and flight 423. The first departs at 15:00 from Heathrow Airport and should arrive at 17:00 at Lisbon Airport. The second departs from Lisbon Airport at 18:00 and arrives on Charles de Gaulle Airport two hours later. Bad weather on United Kingdom prevented flight 355 from departing on time and it was delayed 45 minutes. Keeping the 2 hours flight duration from London to Lisbon, flight 355 would arrive at Lisbon Airport, in the best scenario, at 17:45. This would only provide 15 minutes for aircraft rotation (reach gates, maintenance, passenger boarding,...) making it impossible to depart to Paris on the scheduled time."

Knowing in advance that *flight 355* would arrive late, Flight Dispatcher should alert

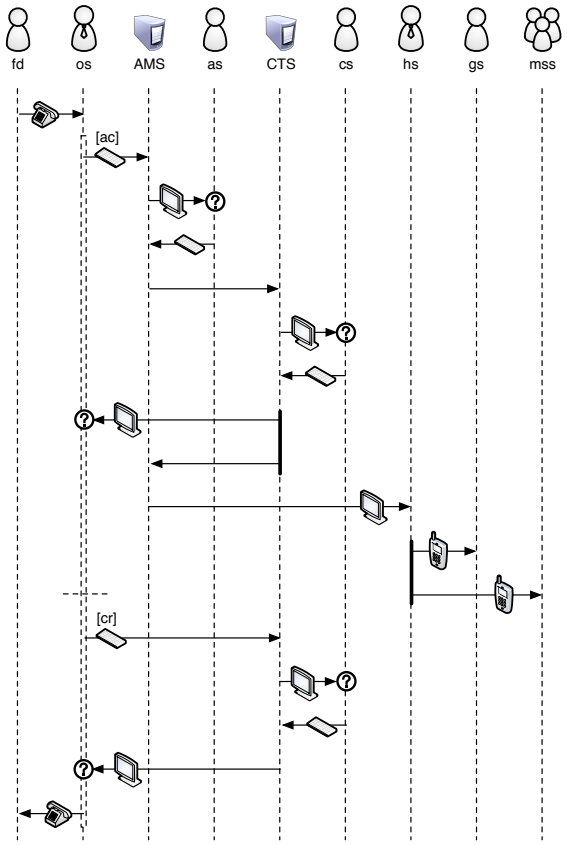

Figure 3.3: Current operational workflow triggered by an aircraft or crew anomaly detected by Flight Dispatcher.

the OCC personnel about that issue so they can try to find a solution. Looking into the diagram illustrated in figure 3.3, he first calls the OCC Supervisor to let him know about that. The OCC Supervisor now has a conditional action to take, if it is an aircraft anomaly he inputs the problem on the "AMS", if it is a crew problem he opts by the "CTS". An example of a crew problem might be the case of a cabin assistant of *flight 355* that would travel to Paris on *flight 423* that gets sick on the meantime.

Back to our workflow, the OCC Supervisor has an aircraft anomaly at hands, so he fills a form on the Aircraft Movement System, to which the Aircraft Specialist is hopefully paying attention. Now the Aircraft Specialist, also called Aircraft Manager, has to reason over the problem, looking for a solution. In subsection 3.1.4, the decision making and reasoning processes found on the Operational Control Centre will be audited, but for now, let's assume that there is an Airbus A330 on Lisbon Airport that arrived at 16:00 and is being prepared to flight to Frankfurt International Airport at 18:45. Let's also assume that *CSTOL* ship, from our example, is an Airbus A320. For a better comprehension, table 3.2 summarizes the considered flight schedule.

Bearing in mind the above scenario is likely that Aircraft Specialist decides to switch

Table 3.2: Conjectural flying schedule illustrating operational workflow (figure 3.3).

FLT_NBR	FROM_AIRP	TO_AIRP	SCHD_DEP_DATE	SCHD_ARR_DATE	AIRC_REG	ACTYP_CD
355	BAA	LIS	15/02/2010 15:00	15/02/2010 17:00	CSTOL	A320
423	LIS	CDG	15/02/2010 18:00	15/02/2010 19:00	CSTOL	A320
534	FRA	LIS	15/02/2010 14:00	15/02/2010 16:00	CSTMU	A330
534	LIS	FRA	15/02/2010 18:45	15/02/2010 20:45	CSTMU	A330

aircrafts. Such change would prevent an estimated 45 minutes delay on *flight 423*. Passengers heading to Charles de Gaulle Airport would board on *CSTMU*, supposed to be ready to takeoff by 18:00, as it arrived at 16:00. Then, the *CSTOL* aircraft, estimated to arrive at 17:45 will have an hour to "rotate" and take *flight 534* passengers to Frankfurt.

Returning to the workflow diagram, figure 3.3, the Aircraft Specialist enters his decision on the "AMS" terminal. This information is instantaneously synchronized over the network with the Crew Tracking System, making the new schedule visible to the Crew Specialist. In a similar fashion, the "cs" has now to analyze the solution proposed by the "as", in order to foresee potential crew shortages or other irregularities. For instance, Airbus A330 is larger than Airbus A320, thus requiring more crew members to carry out on flight duties. Another problem might have its origin on crew member rotation. If the crew members on duty on the late *flight 355* are also assigned to *flight 423* then 15 minutes might be insufficient to move crewman from one aircraft to the other, or some mandatory resting periods might be violated.

Mandating or not some crew assignment changes, the Crew Specialist is required to evaluate, take action and confirm the solution suggested by the Aircraft Specialist through the "CTS" terminal. His input will be readily synchronized, once again, with the Aircraft Movement System, making it available to both OCC Supervisor and HCC Supervisor. As the main character on the Operational Control Centre, "os" is required to ratify the decisions proposed by the Specialists, while the "hs" uses a VHF radio to communicate changes to ground operatives, namely the Ground Supervisor and Maintenance Services. Considering our example, the later collaborators will now be responsible to take action regarding *CSTMU* aircraft provision.

Before going back to workflow description, it is desirable to mention the lack of information about OCC Supervisor criteria to ratify or not the decisions recommended by Specialists. The interviews carried out indicate an "always accept" criteria, with some few, random and circumstantial exceptions. As we will see next, this step on the airline organizational processes will assume a lower relevance when compared to other activities. It was just included on the workflow diagram in order to attest the supervising role of the OCC Supervisor.

The sequence of activities illustrated till now, had origin on an aircraft anomaly foreseen by Flight Dispatcher but figure 3.3, also depicts an alternative operational workflow

translated by the conditional input action carried out by the OCC Supervisor. As the schema shows, if we consider a crew problem, then "os" would directly fill a form on the "CTS" terminal, triggering Crew Specialist action. This possibility hardly requires Aircraft Specialist participation and putative solutions arisen for the problem are transmitted to Flight Dispatcher using phone, so he can notify involved crewman.

To conclude this subsection, it worths emphasizing that the sequence diagrams grouped on section A.1 correspond to abstractions of the most patterned and typical operational workflows observable during regular flight operations. Unanticipated global events, such as natural disasters, acts of terrorism, extreme weather conditions; or even disruptive local happenings, like work stoppages promoted by workers' union or emergency landings, may lead to extraordinary measures that will mirror different operational workflows.

3.1.3 Activity Duration and Geographical Dispersion

The most prevailing goal of the research study documented along this text, is to simulate the organizational structure and operations management practices observable on a real airline company. Thereafter it should propose and introduce changes into the simulation, assessing putative improvements. For other words, this roughly means converting the operational workflows analyzed in the previous subsection into software models, that may receive input data and mimic the collaborative work scenarios.

In order to do that and figuring a way to evaluate workflow performance, we had to know the amount of time consumed by the various subprocesses that compose every operational workflow. These subprocesses overlap, to some extent, with the notion of activity illustrated in figure 3.2. Besides having some perception on those durations, it was also important to understand if the time required to perform an activity on a given workflow was the same time to perform the very same activity on a distinct workflow.

Although section 3.2 will be solely dedicated to present the methodologies behind simulation, given the fact that we had to deliberately send inquiries to airline personnel in order to gather the activities duration and geographical dispersion, we opted to unveiled some simulation intrinsic data at this stage, as it is mainly concerned to empirical observation.

Before presenting some numbers, it is important to note some points. First, the majority of TAP collaborators agreed that it was impossible to assign an objective duration to each activity. It does not matter if it was communicating via radio or reading data from a computer screen, every case has its nuances motivating a random value within a minimum and maximum duration. Second, when confronted with the same activity having different starting and ending points, for instance, if filling forms on "AMS" or "CTS" would take the same amount of time or if a phone call between Station Supervisor or Flight Dispatcher and the OCC Supervisor would present the same duration, they told us that

they were unable to precisely point out differences. Interactions were about anomalies and/or solutions, so if there were differences, they would have to be tracked individually, a tedious, subjective and error prone task.

After gathering TAP collaborators answers, and assuming some commitments in terms of data expressiveness versus simulation complexity, we ended up with the activity durations exposed on table 3.3.

Table 3.3: Time intervals of TAP operational activities.

	Duration (mins)		
Activity	**Min**	**Max**	**Avg**
Communicate by radio	2	5	3.5
Communicate by phone	4	8	6
Data input on system	2	7	4.5
Perceive or read data	1	4	2.5
Solution reasoning	8	15	11.5
Decision approval	0	1	0.5

As stated before, the activities listed on table 3.3 match the activities found on the appendix A diagrams and described on figure 3.2 with one exception. On the durations table, Solution Reasoning and Decision Approval actions were split in order to better distinguish the role of Specialists from Supervisors. As explained in the previous subsection, while the decision making processes carried out by Specialists are systematic, complex and time consuming, on the other hand the decision approval performed by supervisors is fuzzy, based on a "always accept" criteria.

Another interesting fact demanding attention is the reason behind the different times across communications, or why interacting by VHF radio is less time consuming than using the phone. The justification lies on the typology of the anomalies to be reported. While the radio is used between HCC Personnel, comprising Maintenance Services, Passenger Services and the Ground Supervisor, and the HCC Supervisor; the phone links OCC Supervisor with Station Supervisor and Flight Dispatcher. The anomalies detected by HCC Personnel are always related to "on base" (Lisbon Airport) operations, such as aircraft defects, mandatory passenger security, late cargo loading, and so on. On the opposite side, Station Supervisor reports from a foreign station, for instance, a TAP flight had problems on Frankfurt International Airport. Concerning the Flight Dispatcher, as we saw before, he is also concerned with en route problems, harder to transmit to OCC Supervisor.

Anticipating the need to reengineer the airline business processes exposed on the previous section, we were also concerned about understanding the geographical dispersion of operations control since the early stages of our research. On subsection 3.1.1, *Airline Business and Organizational Concepts*, we distinguish different facilities spread trough

several locations.

Aiming at having an idea about the distances between facilities we soon questioned the time taken to move from one point to another. Once again the answer was multifold. While the airline personnel would kindly provide objective metric distances, measuring the time to reach a facility from a given point is not as straightforward. The problem sometimes lies on the notion of facility, or the resources that maybe used to help personnel mobility. For instance, the Airport Parking occupies an area of several square kilometers, so inquiring about how long would it takes for a technician from the Maintenance Services to reach, by feet, the Hub Control Centre, will return an indefinite answer. But if we provide a vehicle to the technician we will get two indefinite answers.

Even facing the above shortcomings on obtaining accurate data, we were able to complete table 3.4.

Table 3.4: Time intervals between TAP operational facilities.

		Duration (mins)					
		feet			vehicle		
Facility 1	**Facility 2**	**Min**	**Max**	**Avg**	**Min**	**Max**	**Avg**
Airport Parking	Hub Control Centre	5	15	10	1	3	2
Airport Parking	Areeiro Crew Terminal	20	40	30	2	5	3.5
Passenger Check-In	Hub Control Centre	2	6	4	–	–	–
Passenger Check-In	Areeiro Crew Terminal	18	30	24	3	5	4
Areeiro Crew Terminal	Hub Control Centre	15	25	20	2	4	3
Lisbon Airport	Operational Control Centre	–	–	–	15	45	30

Data on table 3.4 will be helpful on redesigning current TAP work practices, on the final stages of the research study. Given the impossibility to get precise durations for hypothetical moving activities one must rely on minimum and maximum values to describe the time intervals between facilities.

A final word goes for empty cells, such as the Passenger Check-In to Hub Control Centre and the Lisbon Airport to Operational Control Centre. In the first case, the facilities are fairly close to each other, making it inconvenient to use a car or other vehicle to travel between them. The second case is the opposite scenario, facilities are located at distinct points of Lisbon city, making it impossible to reach each other by feet. On this later case, the huge variation across minimum and maximum values is relate to traffic and rush hours.

3.1.4 Airline Anomalies Description and Classification

While having an in-depth understanding about the current TAP organization structure, concepts involved, observable workflows and subprocesses duration was mandatory to set out our research efforts, the decision making carried out at the Operations Control

Centre was another topic that deserved our attention. The goal was not only simulate how TAP collaborators act on a daily basis, but also how they reason in order to solve operational problems.

As stated before, when it comes to figuring out a way of suppressing a *soon-to-happen* or *already-happening* flight delay, two main characters come into play, the Aircraft Specialist and the Crew Specialist. They are often called *Managers* given their ability to change previously defined rosters (see subsection 2.1.1). Besides gathering some knowledge about how these collaborators solve the airline operational problems, we were also interested in understanding the nature of the anomalies behind putative delays.

Fortunately, airline companies are aware for the need of label and categorize the different delay reasons, and TAP is no exception. On the other hand, at the moment, there is not a standard delay list widely adopted by airline companies thorough the world. The most comprehensive and popular attempt to standardize the reasons behind commercial flight late departures was brought by IATA, the *International Air Transport Association*, an industry trade group involving airline companies from multiple nations.

Table B.1 on appendix B, section B.1, lists all the 72 delay reasons proposed by IATA, spread by 13 categories. It includes the numeric delay code accompanied by a label and a textual description. Airline companies willing to use those codes should add an additional column to their databases in order to record the anomalies. For instance, if a flight suffered a delay because there were no parking stands available, the code *87*, *AIRPORT FACILITIES* should be used.

As we will see later, TAP had already adopted IATA delay codes, although it is used next to the proprietary delay labeling system. Given the lack of extended descriptions or categorization, the later is more dubious and complex comprising more than 200 delay codes. Table B.2 on section B.2 lists a subset of 101 TAP proprietary delay codes and related labels. A brief comparison between IATA and TAP delay classification shows more precision on the later but also some redundant and doubtful entries.

Having a precise understanding of how airline companies identify and record commercial flight delays represented a huge step on the later research stages as it allows to treat anomalies in a targeted way. But, as stated in the beginning of this subsection, it was also desirable to know how different anomalies were handled by Hub and Operational Control Centers personnel. For instance, in the occurrence of a 41 IATA code, *TECHNICAL DEFECTS*, that roughly corresponds to a 831 TAP delay, *AIRCRAFT DEFECTS AT HOME BASE*, what are the measures proposed to minimize or suppress the foreseeable delay?

While we knew, in advance, that scrutinizing the decision making practices would not be as straightforward and precise as receiving a delay code list, TAP are beginning to undertake some business analytic approaches in order to improve its performance. Luckily, they were willing to share some statistics for the purpose of our research study.

Tables 3.5 and 3.6 provide a clear, although probabilistic, view of the decision making processes involved in operations control. The first is concerned with role of Aircraft Specialist, the second respects to the Crew Specialist. Both tables share the same typology, the rows indicate resolution methodologies, while the central columns are related to problem categories.

Table 3.5: Aircraft Specialist action probability according to problem category.

	Flight/Aircraft Problem Category										
Action	**AIRP**	**ATC**	**COMM**	**HAND**	**MAINT**	**METEO**	**CREW**	**ROT**	**SEC**	**OTH**	**Avg**
Change aircraft	0%	0%	0%	0%	80%	0%	0%	80%	0%	60%	22%
Reroute	0%	50%	0%	0%	0%	0%	0%	0%	0%	7%	6%
Join flights	0%	0%	0%	0%	2%	0%	0%	0%	0%	4%	1%
Delay flight	95%	45%	90%	98%	15%	98%	95%	19%	100%	25%	68%
ACMI	0%	0%	0%	0%	1%	0%	0%	0%	0%	1%	0%
Cancel flight	5%	5%	10%	2%	2%	2%	5%	1%	0%	3%	4%

Table 3.6: Crew Specialist action probability according to problem category.

	Crew Problem Category					
Action	**SIGN**	**RULES**	**INDUTY**	**ROT**	**METEO**	**Avg**
Use reserve at airport	10%	5%	0%	20%	0%	7%
Use nearest reserve at home	40%	20%	0%	40%	0%	20%
Exchange with crew another flt	10%	10%	0%	10%	0%	6%
Use crew with free time	10%	10%	0%	5%	0%	5%
Use day off crew	10%	10%	0%	10%	0%	6%
Use crew on vacation	5%	10%	0%	5%	0%	4%
Propose aircraft change	1%	0%	0%	0%	0%	0%
Proceed without crew	2%	5%	90%	5%	0%	20%
Cancel flight	2%	10%	10%	0%	2%	5%
Accept delay	10%	20%	0%	5%	98%	27%

Before presenting further explanations around those tables, it worths explaining the newly arisen concepts. The actions shown on the rows translate the decisions proposed by the Specialists after being notified about an operational anomaly/problem. In its turn, the anomalies/problems listed on appendix B are grouped forming "problem categories", that are represented by means of capital letters. This anomaly clustering is carried out by TAP to allow a better statistical analysis.

One point that deserves a special focus is the categories associated with each Specialist. It is very important to note that the Crew Specialist is solely concerned with crew problems while the Aircraft Specialist cares not only about aircraft problems but also flight problems. This way, despite they are both required to the proper function of the Operational Control Centre, the Aircraft Specialist assumes a more relevant role than the Crew Specialist. If we look back to the workflow diagrams presented on subsection 3.1.2 it is clear that, neglecting the anomalies detected by the "DOV" and "CTS" terminals,

which are directly related crew problems, the Aircraft Specialist is the first to be notified about problems. It then analyses the anomaly, reasons about it, and transmits its conclusions to Crew Specialist.

Back to anomaly categorization, TAP statisticians kindly provided the description of the categories found in the central columns of the probability tables. Flight and Aircraft problem categories are explained in table 3.7 while crew counterparts are shown in table 3.8.

Table 3.7: Flight/Aircraft problems category description.

AIRP	Airport infrastructure causes: SEF, Stands, Airport Capacity, ULDs, RX machines, etc.
ATC	En route and destination ATC restrictions as well as en route and destination meteo conditions.
COMM	Protection of passengers due to cancellation of another flight or passengers missing after check-in.
HAND	Problems boarding passengers and/or loading cargo, problems related with taxing/runway officer.
MAINT	Problems due to some kind of malfunctions and/or maintenance related.
METEO	Adverse meteorological conditions at departure airport impairing landing or handling.
CREW	Missing crew members and other crew related problems (table 3.6 and 3.8).
ROT	Problems related with the rotation of the aircraft. For example, late arrival of the incoming aircraft.
SEC	Baggage identification, search and retrieve of baggage after boarding.
OTH	All others problems not related with any of the previous.

The categories within the tables should not be confused with the previous IATA anomaly grouping previously mentioned and evident on table B.1, *IATA Numeric Delay Codes and Description*. While there are some similarities between them, this categorization is intended to mirror the operational issues faced by TAP, while the later aims at being more prevalent.

Now that all concepts were clarified, it is easy to describe an anomaly resolution process. For instance, a ULD, *Unit Load Device*, inadvertently hits an aircraft during cargo loading. Consulting our delay code tables on appendix B, a 52 IATA delay code or a 980 TAP delay code was raised. The Aircraft Specialist is notified and starts to analyze the problem. This kind of code corresponds to an aircraft problem related with the *MAINT* category, as it roughly translates a new malfunctioning problem or some maintenance will be required. According to table 3.5, there is 80% probability of changing the aircraft

(subject to ship availability), 15% chances of delaying the flight (until repair), 2% likelihood of joining flights (moving the passengers to another TAP flight), 2% probability of cancel the flight and, finally, 1% chances of ACMI (leasing an aircraft from other airline or leasing company).

Table 3.8: Crew problems category description.

SIGN	Crew member not reporting for duty at home base.
RULES	Crew member has exceeded any labour/law rules like duty time.
INDUTY	Crew member unavailable after sign-on. For instance, accident or illness during flight.
ROT	Crew member miss a flight due to the delay of a previous flight or other causes related with crew rotation.
METEO	Adverse meteorological conditions at departure airport impairing landing or handling.
OTH	Other reasons not included on previous.

Now let's see an hypothetical anomaly where the Crew Specialist will intervene. A flight was schedule to depart from the home base at a given time, crew members were timely assigned to the flight but a cabin assistant did not report for duty. This will likely raise a 67 IATA delay code or a 1001 TAP delay code. As this is an inherently crew problem, the Aircraft Specialist would not be notified. If he was, he would decide based on the *CREW* category, 95% chances of delaying the flight or 5% probabilities of canceling it. Then, he would inform the Crew Specialist about his conclusion. One way or another, and since the anomaly belongs to the *SIGN* crew category, the Crew Specialist would propose the all the actions included on table 3.6, but with different probabilities. First he would try to contact a spare reserve crew member that is not on duty (40%), other options would include changing the missing cabin assistant with a crew member assigned to a future flight, use a crew member that is on the airport or having free time (10%). Crew members having the day off or on vacations might be called to ascertain availability (10% and 5% respectively). If there was a decision, coming from the Aircraft Specialist, geared towards delaying the flight and pending for approval, that might be an option (10%). Decisions unlikely to be taken include proceeding without crew, proposing aircraft change and cancel de flight (all <5%).

Although the above examples were intended to make clear how the decision making processes were driven by anomaly classification and probabilistic action, the later also proved that the information provided till now is insufficient to properly model the operations related to an airline. At some point, on the crew example, we did not know if the Aircraft Specialist would participate or not on the reasoning process. This happened because we have already linked anomaly classification with decision making but we are

still missing some connections between operational problems, translated by delay codes, and the concepts that trigger the workflows on appendix A.

Trying to illustrate this lacking of information, and back to the crew example depicted on the previous paragraph, if we know who gets aware of the missing crew member, than we will be able to match an existing workflow and understand if the Aircraft Specialist would take part of the resolution process. After inquiring airline personnel, we were told that 1001 TAP delay codes are reported by the "DOV" system, where the crew members must register for service, and is set up to raise an alarm if a crewman misses the call. Accounting for this extra information, and crossing it with the workflow associated to the Flight Operations Portal, figure A.6, we realize that the Aircraft Specialist doesn't take part in the process, allowing us to simulate such event without glitches.

The lack of this important piece of information had led us to manually classify all the provided delay codes, both IATA and TAP, into two dimensions. The first is concerned with who detects the anomaly and, therefore, who triggering the respective workflow. The second is around the TAP categorization used to probabilistically precipitate a decision making process. Concerning the latter, the last portion of section 3.2 will be dedicated to describe how we tried to improve the manual assigning of problem categories to delay codes by means of decision trees.

As one can imagine, manually classifying 173 delay codes according to textual and thus imprecise descriptions, was an inglorious and tedious task. Having in mind the dimension of the challenge, mainly in terms of time consumption and patience proofing, we opted to personally classify the codes and later deliver it to airline personnel asking for data validation. That was how it happened, and after getting back the documents, around 30 out of 173 possible corrections were made, which somehow attests our familiarity with airline operations.

Appendix C gathers the final human (manual) classification of IATA and TAP delay codes accomplished during the aforementioned phase or our research study. It is split into two sections, each one dedicated to a different set of codes. Table C.1 is concerned with IATA delays, so the first column of the table corresponds to the values shown on table B.1 of the appendix B. On the other hand, the subsequent section contains a table classifying TAP delays, table C.2, and therefore the first column matches the codes listed on table B.2.

Despite representing distinct instances, the second, third and last column of appendix C tables respects to the same entities. As one may observe, they are both labelled with the same attributes: *concept*, *flight/aircraft problem* and *crew problem*. The first attribute conforms with the notion of operational concept, described on section 3.1.1, and its set of values should match the items listed on table 3.1, namely the Human Collaborators or Computerized Systems two or three letters identifiers. The third column, *flight/aircraft problem*, is in close connection to the TAP flight/aircraft problem categorization explained

above and restricting its values to the capital letter labels listed on table 3.7. Similarly, the last column cares about TAP crew problem classification, being related with the identifiers in table 3.8.

It worths remembering that crew problem classification is conditionally dependent on flight classification and thus, the majority of fields associated with the last column, *crew problem*, are empty. If a flight/aircraft problem is classified as *CREW*, then it will require further inspection in order to determine the nature of the crew problem.

On the later stages of this study, the plain text version of the appendix C tables will be used to generate code to be embedded on our simulation but for now, it is useful to conceptually explain how the empirical elements presented fit together. Let's assume that at some point in our operations scenario, a 33 IATA anomaly, *LOADING EQUIPMENT*, occurs. This means that a ULD, provided by the airport, was required to load some cargo into the ship but it was not available. According to our manual classification of IATA codes and later validated by TAP personnel (table C.1), this anomaly was detected by the Ground Supervisor, so it will trigger the corresponding workflow A.1. Following the sequence of activities illustrated on the later diagram, Ground Supervisor will communicate the problem to the HCC Supervisor, which in turn will fill a form on the "AMS" terminal alerting the Aircraft Specialist, a deciding agent. He will then analyze the problem, which virtually corresponds to cross the classification data with the action probabilities on table 3.5. Since a 33 IATA anomaly belongs to the *AIRP* problem category, there are 95% chances of the Aircraft Specialist delaying the flight, and a 5% likelihood of canceling it. The Crew Specialist will be aware of the Aircraft Specialist solution but having in mind that *AIRP* problems do not mandate crew changes, it will not take action. Back to the workflow, the OCC Supervisor will accept the solution proposed by the Aircraft Specialist and the HCC Supervisor will communicate the decision to the Ground Supervisor, who will wait for an ULD to be available (flight delayed) or dismiss ULD necessity (flight cancelled).

To conclude this subsection, it is important to justify the Station Supervisor absence from the classification tables, when there is a workflow diagram dedicated to him. As explained at the beginning of this section, Station Supervisor works on a foreign base, meaning that it exists a Station Supervisor in every airport to where TAP flies. In a real life scenario, when a TAP aircraft reaches a foreign base, let's say Frankfurt Airport, the servicing is carried out by local outsourced entities. When an anomaly is detected, for instance, when there is a problem with the aircraft or adverse weather conditions, Frankfurt Airport personnel reports to the TAP Station Supervisor who is then responsible to call TAP OCC Supervisor as indicated by the respective workflow diagram. When looking to the classification tables, this means that the triggering concept is not only concerned with the delay code but also the departure airport. Disregarding the delay code, if the departure airport is other than Lisbon International Airport, then the operations

triggering concept is always the Station Supervisor.

3.1.5 Operational Activity Logging

Section 3.2 will be solely concerned on expose the approaches followed for operational data analysis and how it allowed decision making improvement. Therefore, this subsection will not go farther than justifying the need for real data usage and presenting an overview about TAP operational activity logging.

At the beginning, when we thought about pioneering a simulation of an airline organizational structure aiming at assessing its performance, two things were clear. First, it was required an in-depth understanding about airline operations control, ideally supported on airline collaborators experience. Second, we needed to have access to real operational data with the purpose of feeding our simulation and after compare reality with simulated outputs.

Fortunately TAP was receptive to the idea of not only share the intrinsic empirical knowledge exposed till now but also open its databases to our research study. Obviously we were not given direct access to its repositories but TAP agreed to provide large text files with requested dumps.

At this point, it is important to clarify the concept of operational activity from a database recording perspective. Wrapping up all the empirical observations made till now, besides the mandatory flight schedule and delay codes logging, the inputs for our simulation, it would be very interesting to have some way of comparing the real-life decision making processes to the artificial reasoning implemented at simulation level.

When looking into tables 3.5 and 3.6 the most prevailing way of solving an airline operational disruption is by delaying the flight or changing the aircraft/crew member. This way, we requested data that would allow us to compare the scheduled to the real operational plan, thus allowing to track decision level changes.

At its origin, the current database implementation at TAP only maintains a single flight record at a time, overwriting any changes to the scheduled record. Table 3.9 intends to illustrate the referred approach.

When looking into table 3.9, the most relevant database field to comprehend the logging mechanism at TAP is the *TABLE_ID*. Before 15/02/2010, three flights were scheduled, so the *ACTL_DEP_DATE* field, actual departure date, was obviously empty. On the 15/02/2010, as the flights successively take place, the *ACTL_DEP_DATE* is updated, causing no harm to the existing data. But an anomaly on *flight 894* motivated an aircraft change, instead of the *CSTPB* ship, it was used the *CSTPG*. This update query was carried out over the existing record, as the table id, *3231344* may attest.

At first, the described logging methodology in use at TAP may serve it purposes but in the context of our research study it is worthless. Assuming that we ask for data on the

Table 3.9: Current operational logging approach at TAP.

TABLE_ID	FLT_NBR	SCHD_DEP_DATE	AIRC_REG	ACTL_DEP_DATE
Database before day of operations				
3231343	452	15/02/2010 14:15	CSTMU	
3231344	894	15/02/2010 14:30	CSTPB	
3231345	972	15/02/2010 15:30	CSTPG	
Database after day of operations				
3231343	452	15/02/2010 14:15	CSTMU	15/02/2010 14:20
3231344	894	15/02/2010 14:30	CSTPG	15/02/2010 14:50
3231345	972	15/02/2010 15:30	CSTPB	15/02/2010 15:30

17/02/2010, we would be given the last portion of table 3.9, after the day of operations, and consequently we would never get to know that the *CSTPB* aircraft was changed.

To overcome this problem, a database service was implemented in order to take snapshots of relevant table rows, before and after operations. This service was not solely concerned with daily data, but also weekly and monthly. It relied on a "PHASEs" approach to distinguish scheduled from real data, according to table 3.10.

Table 3.10: Operational data collection methodology.

	Schedule		Real	
Period	**PHASE**	**Collected**	**PHASE**	**Collected**
Day	10	Daily for the next day	11	Daily for the previous day
Week	5	Every sunday for the next week	6	Every tuesday for the previous week
Month	1	Every 27th day for the next month	2	Every 4th day for the previous month

From the table above, is clear that "PHASE" is a numerical database field introduced to ease the retrieval and comparison of records. In the next section, it will be demonstrated how the systematic analysis between scheduled and real events took place.

To finalize this short subsection and put an end on the broad *Airline Empirical Observation* section, brief mention goes to the scope of files kindly provided by TAP.

As mentioned before, although we pointed out the limitation of current database recording and described a solution based on a database service, when we requested TAP for operational data we received an archive containing large CSV files corresponding to database dumps. Four types of files were included:

- **op_flight**: related to the flight plan, including departures and arrival time, origin and destination airports, flight numbers, and so on.

- **op_aircraft_roster**: concerned about aircraft planning, it involves aircraft model and registry plate, number os seats available and sold, among other ship related information.
- **op_crew_roster**: the aircraft roster counterpart, but dedicated to crew members, it contains data about TAP employees position and numbers, crew pairings, activity planning, flight assignment, etc.
- **op_flight_dep_dly**: includes the time of delay suffered by some flights accompanied by the underlying IATA and TAP delay codes. From the above files, it is the only which has just one related "PHASE", the real operational scenario.

The topics surrounding operational activity logging will be still revisited twice along this document. First, on the next section, where the processes of data understanding, clearance, analysis and classification will be unveiled. Next on the *Experiments* section of chapter 4.1, where it will be described the data used to feed our analysis and simulations.

3.2 Operational Data Analysis and Decision Making Improvement

Considering the empirical knowledge presented in the previous section, and keeping in mind that one of the main goals of the research study herein documented is to assess organizational structure performance, one may question the reason for a section dedicated to operational data analysis. In fact, and inline with other research activities, we often do not have a clear understanding of all the concepts at the beginning, but then, as our comprehension increases, other topics of interest might arise and the will to explore different approaches start to emerge.

This section roughly fits in the above description. Originally, the operational data analysis that partially entitles this section was intended to create a short size infrastructure where one would preprocess the raw operational files coming from TAP and calculate and visualize results after simulation execution. As it is easy to imagine, such infrastructure started to quickly evolve providing tools well beyond the initial aspirations and assuming and increasingly important role next to the organizational structure performance assessment.

Given the amount and complexity of the concepts involved, figure 3.4 tries to illustrate how data analytics finds its way in-between reality and simulation.

A closer and descriptive inspection of figure 3.4 is worthy. At its core, it illustrates three distinct scenarios represented by cloud or elliptical shapes. The first, on the top translates what is happening, right now, on the Lisbon TAP facilities. The real organizational structure contains a set of operational workflows and another of reasoning processes, which were thoroughly described on section 3.1, *Airline Empirical Observation*.

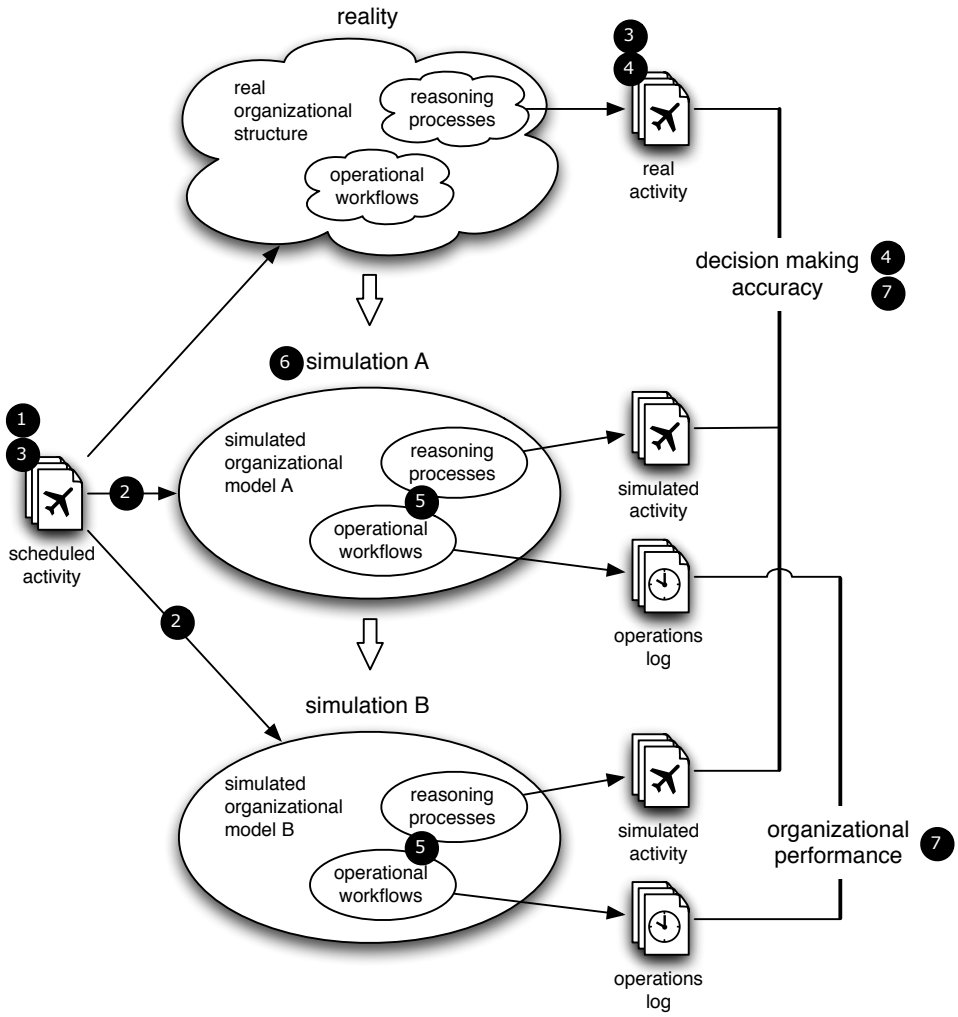

Figure 3.4: The role of analytics in the organizational performance assessment. Extended caption on table 3.11.

The cloud-shaped icon was purposely used, due to the intrinsic complexity of reality, and an always incomplete knowledge about it. At this moment, there is a flighting schedule that must be carried out, there are anomalies that may arise, decisions are taking place to overcome delays, all these activities are being logged into databases. The input and output of all these actions are represented by the arrows starting in the "scheduled activity" file icon, going through the "real organizational structure" and ending up on the "real activity" file icon. Those inputs and outputs match the files referred on section 3.1.5, real airline activity planned in advance and real airline activity that took place afterwards.

Still on figure 3.4, but now moving to "simulation A", the cloud shape of the previous scenario was replaced by an ellipsis. This option reflects the formalism required to model the reality, and thus translating more precise and determined concepts. At this point, the set of fuzzy operational workflows and reasoning processes was replaced by formal algo-

rithms that, even if characterized by some random or probabilistic nature, their behavior will always be deterministic. It is important to recall that "simulation A" does not introduce other constructions or changes beyond a faithful formalization of the knowledge provided by TAP. In other words, "simulation A" attempts at mimic "reality" with the highest fidelity. The same scheduled flight activity that feeds the real airline operations is now inputted into the simulation but unlike "reality", two outputs are produced. Besides the simulated flighting activity, all the operations carried out but virtual TAP collaborators are now logged into files.

"Simulation B" is represented in the same way as "simulation A". The only difference is that a second, "B", organization model is used which involves introducing some changes, at the operational workflows and/or reasoning processes levels.

The scenarios considered allow us to perform comparisons at two distinct levels, decision making accuracy and organizational performance. The former is a side effect of the latter. Actually such comparisons are very different in essence. While the decision making accuracy requires analyzing real and simulated flighting activity measuring the level of similarly, the higher the better; assessing organizational performance requires to define a set of metrics and perform comparisons between them, usually the lower the better.

Back to the title of this section and trying to explain its scope within this research study, the aforementioned analytics infrastructure evolved from input and output file inspection to more advanced data analysis aimed at improving the simulation reasoning processes. Attesting this fact are the seven black numbered dots that represent analytical infrastructure levels and which are spread all over the stages.

From now on this section will be split into smaller subsections, each one dedicated to the most relevant analytical steps or other relevant support features. Heading these document parts, the next subsection wraps up the analysis approach followed along our research study.

3.2.1 Analytical Infrastructure and Support Tools

Since they were mentioned in the introductory portion of this section, this subsection will start by explaining the meaning of the black numbered dots in figure 3.4.

In general terms, figure 3.4 translates the simulation and results analysis stages of the research study documented along this text. The black dots signal the intermediate steps of those stages where the analytical infrastructure takes position. Table 3.11 describes each of the dots.

As one may observe in table 3.11 not all the levels listed are directly related to data analysis and thus this subsection title also mentions the support tools. For instance, levels 2, 5 and 6 are more geared towards file and code generation and handling. In spite of that,

Table 3.11: Analytical infrastructure and support levels (cross-referenced with figure 3.4).

1	Understand database operational attributes as well as track redundant columns in order to simplify the problem at hands and assure the proper selection of data structures.
2	Perform raw data validation, remove unneeded operational attributes and generate lightweight files to feed simulations and ease subsequent information analysis.
3	Compare distinct data PHASEs (section 3.1.5) aiming at gather statistics and produce reports or charts that provide an aggregate view over the airline activity.
4	Based on the comparison described on step 3, apply supervised learning algorithms to observations in order to generate classification decision trees.
5	Generate code from reasoning models, such as classification arrays, decision trees and probability tables, and later merge it into simulation modules.
6	Set up an accessible and intuitive interface that allows for parameter configuration and simulation control, such as the number of executions or simulation speed.
7	Parse output activity files or operational logs in order to collect statistics and present final results by means of tabular reports or pictorial charts.

the analytical infrastructure is of an unquestionable relevance, with an importance on par with the simulation itself.

Back to table 3.11 the levels description numbering was not randomly chosen. It tries to outline the sequence of steps required to assess organizational structure performance, from the raw operational data received from TAP to report and chart generation with aggregate final comparisons. Obviously, and as we will see later, not all the steps have to be touched when researching new simulation scenarios.

The next subsections will be dedicated to specifically explain each level in detail. Nevertheless, in order to provide insight into how levels articulate between themselves, a diagram was produced and it is illustrated in figure 3.5.

In the center of figure 3.5, one may see a sequence of the levels listed on table 3.11, sporting their number and a short description. The connection across units is made through files, that also represent the inputs and output of each step. Arrows were used to represent data flow.

Before presenting an in-depth description of the diagram, it worths clarifying that a *level* should be understood as an approach to address a particular problem. It is usually supported by a specific software model, relying on a set of scripts. As such, and on the context of the analytical infrastructure, another definition for *level* might be a functionality available to humans or other scripts aiming at carry out a specific process.

Returning to figure 3.5, level 1 acts upon the raw operational files provided by TAP

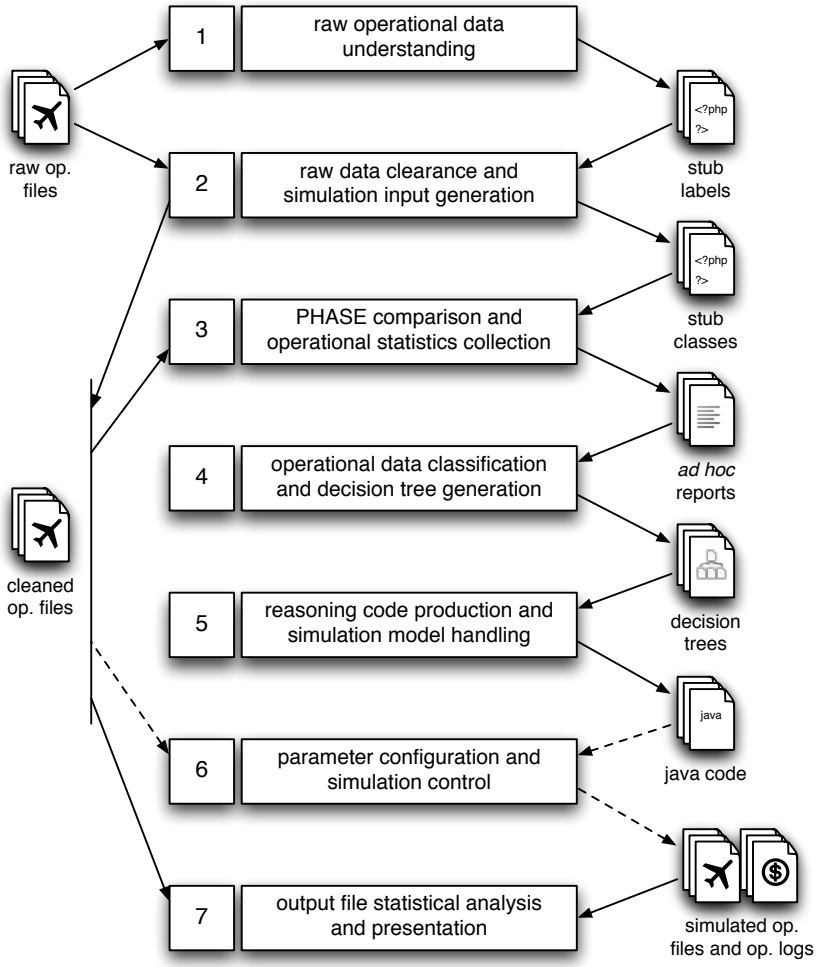

Figure 3.5: The analytical infrastructure articulation process.

and outputs "stub labels" conceptually wrapped in code. Later, the concept of *stub label* will be presented but for now, it matters underlining two aspects. First, the "raw op. files" and the "cleaned op. files", makes no distinction between PHASEs (see section 3.1.5), treating scheduled and real operational activity in the same way. Second, by the end of the process, a set of files are generated containing information used to understand TAP databases attributes as well as placeholders to be filled.

On stage 2, the "stub labels" files and the "raw op. files" are crossed in order to produce "stub classes" and "cleaned op. files". Stub classes will be used on the data analysis of level 3, while the "cleaned op. files" correspond to lightweight versions of the "raw op. files" containing just the relevant columns. The latter are very important files as they will be used in several analytic and simulation processes across the research study.

The next level, uses customized extensions of the "stub classes", to establish comparisons between the scheduled and real airline activity. This analysis is of uttermost importance as it allows to track several parameters such as the number of flights that will

be used during the simulation phase, the number of delays reported, the decision making processes that really took place, and so on. Besides some tables and charts for human visualization, level 3 also generates an "*ad hoc* report" of the reasoning processes attained in reality.

Level 4 is solely feed with the "*ad hoc* reports" previously produced. It then uses different rules to produce formal reports according to the WEKA data mining tool syntax. Later it calls the *J48* Java implementation of the C4.5 classification algorithm, shipped with the WEKA tool, in order to classify the reasoning processes according to delay codes and delay estimation. This process will be thoroughly described on subsection 3.2.4, but at this moment it is enough to understand that level 4 outputs files containing formal representations of decision trees.

As we saw previously, reasoning processes are embedded into simulation models, so level 5 aims at converting the formal representations of decision trees into "java code".

The next level, 6, deserves special attention because it may lead to some confusion. While on the aforementioned levels continuous-line arrows were used, inputs and outputs related to level 6 use dashed-line arrows. This is due to the implicit simulation execution inherent at this stage. In few words, level 6 solely handles some simulation configurations, and provides access to simulation control. It is, by no means, the simulation. The files represented are used by the simulation, and not by the support tool itself, therefore, dashed-lines were used.

Finally, and assuming simulation execution, the analytics infrastructure comes back into play as level 7 uses the "simulated op. files and op. logs" (see figure 3.4). At this step, several cross-comparisons between file sets are performed such as, simulated records with the real counterparts, considering decision making processes; or two or more simulated organizational scenarios, regarding workflow performance assessment.

Now that a big picture over the analytical infrastructure and other support tools was drawn, it matters to detail the intrinsic nature of the most relevant levels by making explicit the underlying implementation or other relevant aspects. We will delve into this subject on the following subsections.

3.2.2 Raw Operational Data Understanding and Clearance

While this subsection is of arguable relevance, it was purposely included to describe some issues faced at the early stages of our research study, when we received large data sets with few or none metadata about them.

After asking TAP for a whole week of scheduled and real operational data, it promptly replied with a link to an archive, containing 7 CSV files, as listed in table 3.12.

At first we were amazed by such amount of data, but if the number of rows, databases records, were more or less expected, the number of columns, database fields, impressed.

Table 3.12: Real operational data set size.

	Number	
Filename	**Rows**	**Columns**
w2010_02_15_ops_plan_aircraft_roster_phase05.csv	7320	72
w2010_02_15_ops_plan_aircraft_roster_phase06.csv	7302	72
w2010_02_15_ops_plan_crew_roster_phase05.csv	22196	78
w2010_02_15_ops_plan_crew_roster_phase06.csv	22424	78
w2010_02_15_ops_plan_flights_dep_dly_phase11.csv	992	32
w2010_02_15_ops_plan_flights_phase05.csv	7318	72
w2010_02_15_ops_plan_flights_phase06.csv	7304	72

Besides its high number, columns had short underscored versions of their real meaning and we did not know which it was. Examples were, *SCHD_DEP_DATE*, *EMPL_PREF_LEV* or *OP_SUFFIX*. While the first label resembled *Scheduled Departure Date*, the next two were impossible to scrutinize. Appendix D lists all the operational database fields accompanied by other information made available during level 1 analysis. Bear in mind, that at the beginning we only had the labels, the second column of each table.

Besides the aforementioned issue, visual inspection of the data suggested redundancy between columns, emptiness across all records, or prevalence of certain instances. Simply put, it "suggested" because it was impossible to visually track thousands of rows.

Following this, we decided to implement a set of scripts that would read all the database records and calculate diverse parameters aimed at help us decide upon which fields would deserve our focus. After executing the script, a set of files with the parameter results would be generated, making it easy to select the fields to maintain or to remove. These files are captioned on figure 3.5 as "stub labels", and table 3.13 depicts its structure and provides and example.

Table 3.13: Stub database label files anatomy.

label	**description**	**removable**	**regex**	**redundant**	**values**
LABEL	<*text*>	<*yes|no*>	<*regex*>	a,b,c	x%: [val1]; y%: [val2]
RANK_CD	Crew rank or flight position	no	/[A-Z]3$/	54,60	53%: [CAB]; 17%: [SCB]; 15%: [CPT]; 15%: [OPT]

As one may see, there are three placeholder fields, *description*, *removable* and *regex*, along the automatically filled attributes. It is noteworthy to remember that besides operational database logging comprehension this analytical level also aims at file clearance and validation, a step where the *removable* and *regex* fields will become determinant.

Looking into the example, after executing the script one would get a file will all the attributes listed, including our "RANK_CD". Besides the label textual meaning, we also know that "RANK_CD" is redundant with the the columns 54 and 60, and the relative

distribution of its values: 53% of the rows are filled with “CAB”, 17% with “SCB”, 15% take the value “CPT” and finally “OPT” holds the remaining records.

After checking the values in columns 54 and 60, likely to provide additional insights, and using the common sense around the “CPT” short form for “Captain” we would conclude that “RANK_CD” points out the inflight position assumed by a crew member. This information could be manually entered on the *description* placeholder, for later reference. Subsequently, and with further inquiry, we would be able to figure out that “CAB” describes regular cabin personnel, “SCP” stands for cabin supervisor and “OPT” means co-pilot.

Tables of appendix D list all the labels per database file. At this stage we discovered that *op_flight* and *op_aircraft_roster* files (see section 3.1.5) contained roughly the same information, so the latter was disregarded in subsequent analysis. The reason for this is because TAP does not have a dedicated database table to track specific aircraft information, storing this information next to the flights data.

Back to table 3.13 example, after knowing what “RANK_CD” stands for, one should decide if the column is relevant for the research study. By filling the *removable* field with “yes” or “no” and providing an additional regular expression on *regex*, the “stub label” files will be complete and ready to proceed to level 2.

Level 2 is regarded as a support stage with straightforward outcomes, therefore it won’t deserve a special subsection. Having the raw operational files and the stub label files as inputs, it simply crosses the latter with the former, skipping all the attributes marked as removable, validating the syntax of the others and creating by copy cleaned operational files (captioned as “cleaned op. files” in figure 3.5). It also produces “stub classes” files, that render an *object-oriented* approach to the forthcoming analysis processes.

3.2.3 PHASE Comparison and Statistical Analysis

Till now, we saw the goals of the research study documented along this text and we have a set of cleaned and targeted files comprising *pre* as *post* airline activity records. In order to provide an answer to the goals set, we must simulate various airline operational scenarios, but if we do not have a clear understanding over the data that will feed the simulation we will not be able to evaluate its efficiency, at least in terms of decision making processes.

Following this, the present subsection aims at explaining what happens at level 3 of the analytical infrastructure. This level carries out a statistical analysis of flights, delays and crew rosters, intersecting the information in-between them and attempting to generate a informal and aggregated representation of the real airline reasoning processes, the “*ad hoc* report”. In the meantime, it presents several reports to the user, pointing out interesting facts about the records to be inputted into the simulation and alerting for data consistency and integrity violations.

Bearing in mind that operational data is split into PHASEs (subsection 3.1.5), the list bellow summarizes the main statistical analysis implemented on level 3:

- **IDs and Foreign Keys**: for each file/table, we had to determine the set of fields that would uniquely identify each row, its identifier. It was also important to link the different files/tables by means of foreign keys.
- **Number of Flights**: by traversing *op_flight* files we were able to determine the number of flights before and after operations, tracking down orphan flights on both sides.
- **Flight Row Completeness**: checking the number of empty fields on each row across PHASEs, proven useful into evaluating the integrity of the data at hands.
- **Flight Column Completeness**: comparing the number of empty fields per column between PHASEs, allowed us to understand which attributes were updated after operations.
- **Flight Field Difference**: this analysis measured the number of different fields across PHASES. For instance, if the AIRC_REG (see table D.1) column would show 3 occurrences then 3 aircrafts did not fly as scheduled, and putatively had to be replaced.
- **Number and Grouping of Delays**: by tracking the *op_flight_dep_dly* file, we were able to determine which flights suffered anomalies. This analysis was behind a crucial discovery by reveling that a flight might had more than one delay associated.
- **Delay Types and Resources**: in order to assist the manual delay code classification (appendix C), this stage aimed at gather unique delay type/resource pairs as well as the number of their occurrence.
- **Crew Roster Matching**: from the *op_crew_roster* files, we examined the number of crew rosters *pre* and *post* operations, detecting orphan rosters on both PHASEs.
- **Delay Calculations**: considering all the files, there were 3 ways of retrieve the flight departure delay, in minutes. Verifying the consistency between different values was accounted for on this analysis.
- **Time Difference Distribution**: the time between real and scheduled departures was calculated for each flight allowing us to understand the distribution of positive or negative (because some flights leave ahead of time) time differences.

The incremental analysis discriminated above yielded not only an in-depth understanding of the fields used in airline operational logging but also some awareness concerning missing or inconsistent data. On chapter 4, *Results and Analysis*, some statistics collected after the data set used in the research study will be unveiled, evidencing the relevance of this stage.

Exposing the implementation behind each item of our PHASE comparison is beyond the scope of this study. Nevertheless, and at this point, it is noteworthy to mention the absence of true linkage between tables, which revealed to be a moderate shortcoming on data analysis.

As explained on subsection 3.1.5, our post-operational data was spread by 3 files, while our pre-operational data was spread by just 2. The difference lies on the *op_flight_dep_dly* file, which contains the anomalies that motivated a late flight departure. For each set of files we had to figure out a group of fields to reference each other rows. In few words, and considering the schedule PHASE, we had to know which crew roster, on the *op_crew_roster* file, belonged to each flight, on the *op_flight* file.

After some testing we determined that to univocally identify a flight one would require four attributes, FLT_NBR, FROM_AIRP_CD, SCHD_DEP_DATE and FLEG_STAT. Appendix D contains the description of the columns but further explanation should be supplied regarding the last one. FLEG_STAT registers the number of failed take offs. At the beginning, every flight is assigned "S", but if there is a problem after departure that forces the aircraft to go back to the origin airport, the database record is duplicated and the FLEG_STAT field is set to "1" on the new row.

Back to flight ID, we noted that in most cases two or more flights departing from the same airport on the same day get different numbers. This logic would made FLT_DATE, FROM_AIRP_CD, SCHD_DEP_DATE and FLEG_STAT enough to identify a flight. Unfortunately, some cases fell out of this rule and we were forced to use SCHD_DEP_DATE instead of FLT_DATE. And this is where the problem began, since SCHD_DEP_DATE cannot be found on delay and crew roster files.

Further file inspection, showed us that this problem could be alleviated by tracking DATE_TIME and/or STRT_DATE of the *op_flight_dep_dly* and *op_crew_roster* files, respectively. The former indicates the date and time of delay registration, while the latter points out the crew member start of activity. Despite this fact, those attributes also proved to be very inaccurate and given the small percentage of affected flights, less than 0.1%, we preferred to use the FLT_DATE as foreign key.

To enlighten the aforementioned problem, table 3.14 shows two flight records suffering from referential deficiency.

Table 3.14: The operational database foreign key problem (attribute description on appendix D).

flights (op_flight)

FLT_DATE	FLT_NBR	FROM_AIRP_CD	FLEG_STAT	SCHD_DEP_DATE
15/02/2010	114	OPO	S	15/02/2010 06:45
15/02/2010	114	OPO	S	15/02/2010 14:15

delays (op_flight_dep_dly)

FLT_DATE	FLT_NBR	FROM_AIRP_CD	FLEG_STAT	~~SCHD_DEP_DATE~~ DATE_TIME	MINS
15/02/2010	114	OPO	S	15/02/2010 15:30	45

crew roster (op_crew_roster)

FLT_DATE	FLT_NBR	FROM_AIRP_CD	FLEG_STAT	~~SCHD_DEP_DATE~~ STRT_DATE	EMPL_NBR
15/02/2010	114	OPO	S	15/02/2010 12:15	65290.3255

According to table 3.14 there are two flights 114 from Porto on the 15th June 2010, so it is impossible to distinguish them by FLT_DATE. Using the SCHD_DEP_DATE instead makes it possible to uniquely identify each other. But, now we want to know which flight suffered the 45 minutes delay on the *op_flight_dep_dly* file. That is something impossible to state for sure, because the DATE_TIME field is manually inserted into the database, making the gap between scheduled departure plus delay and time of registration impossible to estimate. In this case, if we consider the 06:45 flight it took 8 hours for the airline personnel to register the delay; assuming

the second flight, just 30 minutes. The likelihood of the delay corresponding to the 14:15 flight is well greater than the 06:45 flight, but not with complete certainty. Regarding the crew roster, the linkage is less dubious. If the employee 65290.3255 will start his activity at 12:15, he would not be assigned to a flight that had already departed, so he is related with the 14:15 flight.

Now that some referential constraints between files were exposed and workarounds were suggested, it is time to return to level 3 expected outcomes, namely the "*ad hoc* reports". This last level 3 feature intends to draw a big picture over the reasoning processes carried out by Aircraft and Crew Specialists (refer to section 3.1).

For the purpose of accomplish this goal, we needed to compare each PHASE concerning aircraft and crew rosters. An excerpt of such comparison report is displayed in table 3.15 and the some points worth consideration are described next.

Table 3.15: Excerpt of a PHASE comparison *ad hoc* report. *ac* and *cm* abbreviations found on the solution column stand for *aircraft* and *crew member* respectively.

	PHASE 1|5|10		PHASES 2|6|11							
fid	***pre* aircraft**	***pre* crew**	**tap code**	**iata code**	**dly1** (mins)	**dly2** (mins)	**dly3** (mins)	***post* aircraft**	***post* crew**	**solution**
... *(1)*	CSTOL	27423.3|CAB 27626.1|OPT 27941.4|CAB	858 951 1032	93 35 9	41 6 25	72	72	CSTOL	23250.4|CAB 27626.1|OPT 27941.4|CAB	cm replaced
... *(2)*	CSTOO					9	9	CSTOO		
... *(3)*	CSTOJ	18079.4|CCB 19823.4|CPT 27908.3|OPT	1004	68	8	7	7	CSTOJ	12793.6|CCB 19823.4|CPT 27628.7|OPT	cm replaced (2x)
... *(4)*	CSTTS		1021	93	10	8	8	CSTTG		ac replaced
... *(5)*	CSTTR	30236.4|CAB 30244.8|CAB 30309.9|CAB	1018	89	12	15	15	CSTTD	30236.4|CAB 30244.8|CAB 29466.0|CAB	ac replaced cm replaced
... *(6)*	CSTNP	25060.5|CPT 28375.4|CAB 30540.9|OPT				-7	-7	CSTNP	25060.5|CPT 28375.4|CAB	cm missing
... *(7)*	CSTJE	18127.1|CCB 25849.1|CPT 28081.8|CAB 28277.2|OPT	949 1018 858	34 89 93	15 5 5	25	24	CSTJE	18127.1|CCB 25849.1|CPT 28081.8|CAB 28277.2|OPT	
... *(8)*	CSTNM	24532.4|CPT 25099.3|CAB	836	85	5	5	5	CSTNM	24532.4|CPT 29599.8|CAB 30295.0|CAB	cm replaced cm added
... *(9)*	CSTNN	20328.1|CCB 23103.5|CPT 28620.3|OPT	1018	89	5	0	0	CSTNN	20328.1|CCB 23103.5|CPT 28620.3|OPT	

Table 3.15 is an excerpt of an "*ad hoc* report" obtained with real data. It is purportedly long in order to include some inconsistent cases, which in turn show how imprecise and inaccurate the provided records are. Later, on the next subsection about data classification, we will show the compromises we had to assume and the efforts made to allow a reasonable analysis.

Before presenting some notes about the cases illustrated on table 3.15, it matters to point out that "fid" stands for flight ID, a string of characters comprising by FLT_NBR, FROM_AIRP_CD,

SCHD_DEP_DATE and FLEG_STAT, but here replaced by a numbering scheme to keep the table more compact an easier to reference. Next, the table shows nine columns retrieved or calculated from the different PHASE files. The first two, related to PHASEs 1, 5 or 10, depict the scheduled aircraft and crew. The next illustrate the *post*-operational scenario, indicating the anomaly codes suffered by the flight, associated delays, and the aircraft/crew really used. The final column, analyses the data on the row to determine the putative solution(s) taken by Operational Control Centre Specialist.

Another brief word goes to the delays. Including three time intervals on the table might look strange at first but they correspond to the three available ways of calculating the real delay of a given flight. The first one, corresponds to the MINS database field of the *op_flight_dep_dly* file and it is manually filled by a TAP collaborator; the second one is the difference between the actual and scheduled departure date, or the ACTL_OFFBLK_DATE and SCHD_DEP_DATE fields, and the latter is the automatically calculated field DLY_DEP_MIN of the *op_flight* file. Including the three delays aimed at showing the rare but existing discrepancy between them, something that although exquisite, was later used in our favor during reasoning algorithm generation.

Back to table 3.15, on the first row one may see that flight *(1)* suffered three distinct anomalies, 858, 951 and 1032, that motivated a crew member replacement. On the next subsection, we will come back to this case as it is a specimen that imposed heavy constrains to our research.

Flight *(2)* represents data incompleteness, because it suffered a delay of 9 minutes but it has no delay codes associated or even crew assigned. These cases were, unfortunately common, and they were impossible to analyze. On the next chapter, some numbers regarding the relative amount of missing information will also be presented.

The next row, *(3)* alerts for two aspects. First the discrepancy between delays; second the deepness of our analysis. Considerations about the former were already made, the TAP collaborator inputted a 8 minutes delay while TAP database and our calculations determined just 7 minutes. Regarding the latter, our "*ad hoc* report" tracked multiple crew roster changes not only considering the crew member number but also accounting for his rank.

On the 4th flight, *(4)*, an aircraft change was detected after a 1021 anomaly that also caused a moderate and divergent delay of 10/8 minutes. More notorious was the inability of track down any crew changes given their absence, both on *pre* and *post* operational scenarios. Another unfortunate case of missing data.

Flight *(5)* simply illustrates our algorithm capability of determine aircraft and crew member alterations at the same time; while flight *(6)* points out an interesting fact. First and foremost, a flight may depart ahead schedule, as the -7 minutes proves. Second, even though there was not anomalies or delays reported, crew changes were detected, namely a crew member missed the flight (and was not replaced).

The next row, *(7)* illustrates a scenario where there were 3 reported anomalies that produced delays of 15, 5 and 5 minutes respectively but there were not any aircraft or crew changes. At this point we are not concerned about justifying such behavior but if the anomalies were related with adverse weather or restrictions on departure airport, there was nothing airline personnel could do.

On flight *(8)*, an anomaly translated by a TAP 836 delay code, caused not only a crew member replacement but also a crew member addition. These cases assumed some prevalence uncovering putatively incomplete crew roster assignments during the schedule phase.

Finally, row *(9)* translates a flight that suffered no delays but was signaled as having suffered a 1081 anomaly that caused no changes. The TAP collaborator putatively tried to overcome system validation by inputting a delay of five minutes.

The paragraphs above and originating table 3.15 revealed the analysis made possible through the provided TAP operational files and pointed out some shortcomings on that analysis. The next subsection explains how the "*ad hoc* reports" were used to improve the decision making algorithms of our simulations.

3.2.4 Data Classification and Reasoning Code Generation

Recalling section 3.1, we saw that on a real airline company the decision making agents, the Aircraft and Crew Specialists, reason solely according to IATA or TAP delay codes. At least, that was the only information made available by TAP. Since we will be simulating operational activities, we may consider the reported delay codes as anomalies happened before decision making and try to have Specialists behave as in reality, using the provided action probability tables.

Although the approach described in the last paragraph is perfectly valid from a simulation perspective, where agents decide based on reasoning schemes provided by real-life counterparts, in our study we wanted to go farther than that and research new decision making practices based on additional parameters or existing reasoning algorithms.

Other motivation to apply supervised learning techniques was the impossibility of tracking certain real resolution actions through the provided real data. For instance, looking into probability tables 3.5 and 3.6 there are 6 actions associated to Aircraft Specialist and 10 actions that might be chosen by the Crew Specialist. Accounting for the data on the real files, and the analysis materialized by the "*ad hoc* reports" we are only capable of validate some sort of the actions, namely "Change aircraft", "Cancel Flight", crew member changes and as a resort, "Delay Flight".

Figure 3.6 hopes to clarify the trichotomy between Operational Control Centre actions, our simulation reasoning algorithms and the possible real-life data validation.

First and foremost, and as the caption points out, the squares containing numbers and lower-case letters in the diagram illustrated on figure 3.6 are closely related to the action taken by airline Specialists when facing an anomaly during operations. An exception worthing notice is the absence of the "Propose aircraft change", "Cancel flight" and "Accept delay" Crew Specialist actions (lower-case letters), as they somehow overlap with the Aircraft Specialist ones.

The diagram is divided into three columns. The first tries to symbolize the reasoning processes that lead to actions on reality and "simulation A". Subsection 3.1.4 describes those decision making practices that are based on action probabilities. On real-life, an anomaly is communicated to Specialists and later, on the TAP operational databases is registered the delay code that describes it. In an opposite direction, our "simulation A" will use those delay codes to become aware of anomalies and reason about them.

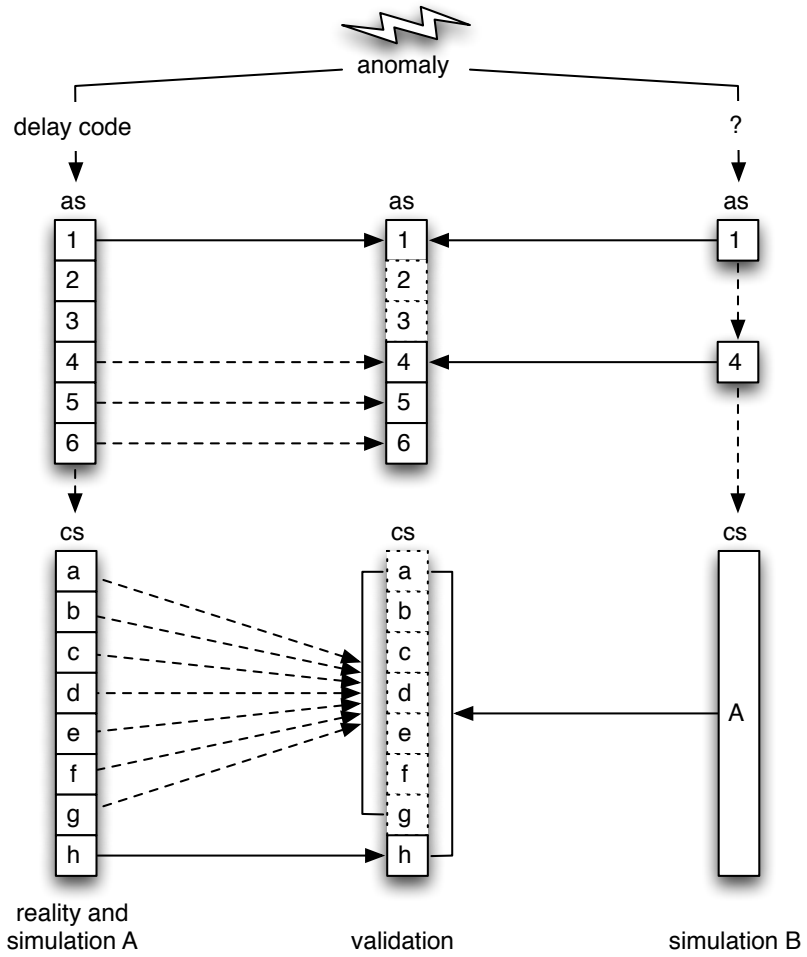

Figure 3.6: The real-life, simulated and validated reasoning trichotomy (numbers from *1* to *6* match the actions on table 3.5; letters from *a* to *h* correspond to actions on table 3.6; *as* and *cs* stands for Aircraft Specialist and Crew Specialist).

The next section will be solely dedicated to detail our simulation internals, but after running "simulation A", we will get operational results translating the decisions carried out by our virtual Specialists. That is where the column in the middle comes into play as it translates the solutions calculated in the PHASE comparison "*ad hoc* report" illustrated in table 3.15.

The arrows in-between the first and second columns attempt to indicate which actions might be validated after real operational data analysis. Trying to concretize, the next paragraph presents a simplified example.

Let's imagine that an aircraft would take much time to get ready for departure, an already defined aircraft rotation problem. In reality, someone would notify the Aircraft Specialist, on our "simulation A" a 1021 TAP delay code or a 93 IATA delay code would be raised (more details later). According to our manual delay code classification (see appendix C) both codes fall into the "ROT" category (see table 3.7). In reality the Aircraft Specialist would change the aircraft on 80% of the cases (see table 3.5) and since our simulation agent implements the same reasoning

strategy, it would do the same. Consider they both acted that way, replacing the aircraft. Now, let's assume this case corresponded to the row *(4)* of our "*ad hoc* report" (table 3.15) where the aircraft was replaced in reality. The decision of changing the aircraft at the simulation level was validated by real-life data analysis. To a short extent, as we considered just one flight, this proves the correctness of the reasoning algorithms implemented at the simulation. Performing this kind of analysis for many flights may confirm or deny such observation.

Back to figure 3.6, this comparison between reasoning output and real data analysis is represented using the middle arrows and as it is possible to verify not all the actions validation is as straightforward as the example above. Actually, the example on the prior paragraph fits on the "Change aircraft" action, represented by "square 1", and linked with a continuous line arrow to the validation column. Unfortunately, the remaining validations are not as seamless, except the "square h", representing the "Proceed without crew" action, which, as we saw on the "*ad hoc* report" is easily tracked.

"Squares 2" and "3" respects to "Reroute" and "Join Flights". These cases are impossible to discover from the analysis of the files provided by TAP, and thus are not even linked to validation column. On chapter 4, where the simulation results will be exposed, these cases will be measured as uncertainty.

"Delay flight", or the case depicted by "square 4" is very interesting to analyze. We opted to signal it with a dashed arrow because, while it is one of the most prominent decisions to surpass an anomaly, it is very hard to track it when examining the real operational data supplied by TAP. Delaying a flight is usually regarded as a fallback action to which Specialists resort when nothing else is possible. Leasing an aircraft and canceling the flight are the other two and will be explained next. The problem with validating a delay decision is because it assumes the role of cause and consequence. The *op_flight_dep_dly* file from where we obtain the delay codes to identify the disrupted flights, only lists the flights that actually suffered a delay. This causes difficulties in simulating the operational decision of delaying a flight because that action is *always* implicit, even if the Specialists never wanted it to be. For instance, a flight might had a mechanical problem, so the Aircraft Manager decided to replace it. Fortunately airline personnel was able to prepare the new aircraft on time, so it did not suffer any delay. Although there was an anomaly this flight would not be registered on the *op_flight_dep_dly* file. Shortly, we will see how this implicit nature of the delay, obliged us to assume some compromises when classifying the data.

The last dashed validation related to Aircraft Specialist (5 and 6) are concerned with "ACMI" and "Cancel flight". When looking into table 3.5 probabilities, one may notice that the later two actions are used in less than 5% of anomaly occurrences. That is because leasing an aircraft to other airlines/renting companies (ACMI) or canceling a flight is very expensive when compared to other solution strategies. Along their low probability, it is not precise to determine "ACMI" or flight cancelations from the TAP operational data. Having an orphan scheduled flight that is not listed on the *post*-operational flights does not necessarily mean that the flight was cancelled. It is the same logic of the absence of crew members on certain flights (see table 3.15). As we saw, and just because a flight does not have an assigned crew, it obviously does not mean that the flight

took place without pilots or cabin personnel. Crew members flew with the aircraft, the missing ones are the records. Regarding the "ACMI", is it easier to determine than cancelations, although it requires an additional list of all the aircrafts operated by TAP without extra cost.

Now moving to the Crew Specialist actions, and as already discussed, they are much less prevalent than the flight or aircraft actions previously described (refer to subsection 3.1.4). "Square a" to "square g" translate the actions from "Use reserve at airport" to "Use crew on vacation" (see table 3.6). Although there are 6 variants, in essence they all translate the same action, the absent crew member is replaced by another one. Following this, we chose to represent the validation arrow using a dashed line since it is possible, with some uncertainty, to validate the decision making process behind such action. The "square h" was mentioned at the beginning, as being a case, on par with "square a" that we can track down with absolute confidence.

Before explaining "simulation B", it is clear that our reasoning validation strategy is far from ideal when it comes to assess the empirical decision making processes exposed. Aiming at simulating hundreds or even thousands of flights, we are not able to inquiry TAP personnel for action validation, or try to manually analyze all the output produced. We must rely on what we have, the database records and if we want our simulation to perform better, we should try to figure out new reasoning algorithms. That is where classification is handy, as it provides some Artificial Intelligence tools that may be used to *teach* our simulation to reason over the inputs.

"Simulation B" uses a different approach when it comes to decide upon an anomaly. The general idea is using the supervised machine learning techniques explained on the *State of the Art* (section 2.2) to generate decision trees that will replace the empirical probabilistic action tables used on "simulation A". Doing so, and as figure 3.6 illustrates, we will be able to classify the delay codes, and other additional information according to our validation strategy. The question mark on the top of the "simulation B" denotes such additional data. In the next chapter, *Results and Analysis*, we will demonstrate how the use of the recorded delay as an hypothetical estimation of the forthcoming delay, contributes to increase the reasoning accuracy. Actually, this is not new, as the airline personnel accounts for delay estimation on their decision practices, they simply were not able to determine or provide insights to the extent of such accounting.

With the purpose of classify the data at hands, we had to take our "*ad hoc* report", and follow the supervised machine learning process illustrated in figure 2.3. The first step of the process, "Data collection", was already concluded and its output was our informal "*ad hoc* report". The next step was "Data pre-processing", so we had to define a set of formalisms to filter and prepare the information analyzed on level 3 of our analytical infrastructure (see figure 3.5).

In order to clarify the set of formalisms, listing 3.1 depicts the contents of table 3.15 after the classification preprocessing phase.

Listing 3.1: *Ad hoc* report classification preprocessing

```
(3),1004,68,8,7,7,cr
(5),1018,89,12,15,15,ac|cr
(8),836,85,5,5,5,ac
(9),1018,89,5,0,0,dl
```

A first look into table 3.15, figure 3.5 and listing 3.1 allows us to clearly understand what we wanted to achieve with such data conversion. With data classification, we intended to remove the dubious dashed validation arrows and have straightforward and precise ways of verifying the performance of simulation output. This way, we had no other options than target our approach towards the analysis of the real operational data. Let's describe the contents of listing 3.1.

Each row from the listing is related to a row from table 3.15. The numeric "fid" at the beginning is for reference only. The formal reported produced after the "*ad hoc* report" follows the *comma-separated values* syntax.

Although flight *(1)* had a crew member replaced, it did not show up in the preprocessing text. It worths underline that the reason for this fact has to do with multiple anomalies associated with just one flight. As we saw on subsection 3.1.5 about how TAP record its *pre* and *post* operational activity, the minimum granularity between the same flights was one day. This means that we can track the flight the day before and the day after operations. Having multiple anomalies, each with its own associated delay, might cause different actions that lead to roster changes. For instance, we know that flight *(1)* had three reported anomalies, namely 858, 951 and 1032 if considering the TAP delay codes, and we also know that a crew member was replaced. What we do not know is which anomaly motivated such crew change. This way, in order to avoid harming the correct and precise classification of single anomalies, these rows were subtracted to the classification and simulation process. Later, on chapter 4 the impact of these subtraction on our research study will be determined.

Moving to the 2nd flight of table 3.15, it easy to understand its absence from listing 3.1. Despite its nine minutes delay, no associated delay code was filled so it is unsuitable for classification.

Flight *(3)* allows us to start to present our formalism regarding solution encoding, as it appears on listing 3.1. As one may observe, the delay codes 1004 and 68 and the delays, 8, 7, 7, were directly migrated from the "*ad hoc* report" to our preprocessing report. The difference lies in the last value `cr`. In order to prepare data for algorithmic classification, we had to encode the different solutions into symbols. Looking into figure 3.6 it is clear our compromise in grouping all the crew changes into one class, designated by "A". This simplification also translate the lower relevance of the Crew Specialist on the empirical actions observation. At this point, it worths momentarily suspend our listing 3.1 exposition to introduce table 3.16 containing our proposed mapping between solutions and their symbolic representation.

Table 3.16: *Ad hoc* report solution symbolic representation (*ac* and *cm* abbreviations found on the solution column stand for *aircraft* and *crew member* respectively).

solution	**symbol**
cm replaced	`cr`
cm missing	
ac replaced	`ac`
ac replaced cm replaced or missing	`ac|cr`
fallback	`dl`

Table 3.16 will be revisited as we explain listing 3.1. But at this moment, it worths emphasizing that the number of changes was disregarded during the conversion between solutions and symbolic character strings. Flight *(3)* illustrates this case. There were two crew members replaced but the formal representation only includes the label `cr`.

Even though flight *(4)* has an associated anomaly it was ignored during the preprocessing phase. The reason is again data incompleteness. As it is possible to observe, no crews were assigned to the flight and therefore we are not able to determine if the delay code usually motivates crew changes. Following this, we preferred to ignore rows with missing fields in order to prevent later corruption on the learning process.

The 5th flight meets all the conditions to be accepted for classification. All the database fields are present, it has a reported anomaly that caused changes on both aircraft and crew. This way, and according to table 3.16 it would be represented with the `ac|cr` symbol, what is reflected in listing 3.1.

Flights *(6)* and *(7)* do not appear on the formal report due to reasons already pointed out. The former lacks an anomaly, while on the other hand, the latter has multiple anomalies.

The 8th row of table 3.16 presents a special case worthing attention. While at first it would deserve the `ac|cr` it was solely labelled with `ac`. The reason for this already deserved a brief mention, when we figured out that crew member addition was not listed as an action taken by the Crew Specialist. This way, when a crew member was added to the *post*-operational roster, it regarded as incomplete scheduling and thus, not considered at this conversion stage.

Finally, the last row, depicts the *fallback* solution on table 3.16. Some paragraphs ago, we underlined the "Delay flight" action as being a last resort to where the Specialists recur when nothing else was possible. We followed the same logic here. Flight *(9)* had an anomaly reported, some precise delays set, but no changes on rosters. On this cases, we considered that the Specialist simply decided to delay the flight, and accordingly we mark it with the `dl` symbol.

At this moment is important to recall that the approach used to analyze and encode the "*ad hoc* report" produced in level 3 of our analytical infrastructure is taking place at level 4, which in turn is dedicated to operational data classification and decision tree generation. As we stated at the beginning, all the methodology explained on the previous paragraphs correspond to the "Data pre-processing" stage of supervised learning process, figure 3.4.

The next step on the classification process is to select a set of features to classify our solutions. This simply implies going to our listing 3.1 and specify which features (*tap code*, *iata code*, *delay1*, *delay2* and/or *delay2*) will be used to categorize the solutions. Actually, as unveiled in the *State of the Art*, our research study will make use of the WEKA data mining tool, making it seamless to classify our data according to just a subset of the features.

In the next chapter, *Results and Analysis*, we will perform some tests using different features, so for now it remains explaining how the WEKA libraries were used and showing a putative outcome of our classification.

Simply put, after having a structure of attributes and classes identical of our formal report, listing 3.1, it is straightforward to use the decision tree generation algorithms ship with the WEKA

tool. The only thing we need is to add an header to the text presented on listing 3.1 precisely indicating the name and type of the attributes and their symbols list. Since this step is more concerned with WEKA internal, it will not be extensively discussed here.

WEKA tool provides a graphical interface to select the attributes to classify and choose from different decision tree generation algorithms. After classifying the inputted data it returns a textual representation of a decision tree among several statistics. The WEKA tool also performs decision tree validation according to user defined rules. Again, we will defer the exhibition of different algorithm usage and validation statistics to the next chapter.

In order to demonstrate how the obtained decision tree fits in our decision-making improvement and proceed to the next analytical infrastructure level, dedicated to reasoning code generation, listing 3.2 shows an excerpt of a decision tree obtained with C4.5 classification algorithm.

It is convenient to underline that the tree bellow was not produced solely after the data on listing 3.1 as it was insufficient to produce significative results. It corresponds to an excerpt of the classification of more than three hundred flights, that used the *tapcode* and *delay1* features. To be more comprehensive, only the *tap codes* branches that match listing 3.1 code were included.

Listing 3.2: Decision tree obtained with C4.5 algoritm

```
tapcode = 836
|   delay1 <= 7
|   |   delay1 <= 5: dl
|   |   delay1 > 5: cr
|   delay1 > 7
|   |   delay1 <= 9: ac
|   |   delay1 > 9: cr
tapcode = 1004
|   delay1 <= 8: cr
|   delay1 > 8: ac|cr
tapcode = 1018
|   delay1 <= 6: dl
|   delay1 > 6
|   |   delay1 <= 11: cr
|   |   delay1 > 11: ac|cr
```

Aiming at providing an example of decision tree improvement, let's cross the data on listing 3.1 with the decision tree above. At first we have a 1004 anomaly with an estimated delay of 8 minutes. According to our decision tree this will cause our Specialists to change the crew. The next row shows a 1018 anomaly with 12 minutes delay, which is classified as `ac|cr`, that is changes of both aircraft and crew. The flight *(3)* suffered a 836 anomaly and a 5 minutes delay, which translates into a "Delay flight" action. Last but not least, the 4th flight has a 1018 reported anomaly next to a 5 minutes delay, which corresponds to `dl`. Summing up the above classifications we have `cr`, `ac|cr`, `dl` and `dl`. Matching with the real solutions representation, `cr`, `ac|cr`, `ac` and `dl`, we have a 75% success, putatively higher than the probability-based reasoning supplied by TAP.

To conclude this subsection it is important to mention that level 5 of the analytical infrastructure, figure 3.5, provides support tools that convert WEKA generated decision trees, such as the one depicted on listing 3.2, to Java code. This code is later embedded into the reasoning modules of our simulation, allowing for interchangeable decision making strategies.

The next section will be solely dedicated to expose the internals of the simulation itself. Instead of extensibly describe the code implementation behind the organizational structure modeling, we preferred to present the most scientific features of *Brahms*, the simulation engine used, and how they articulate with the TAP operational workflows. A brief reference goes to the limitations of *Brahms* in terms of execution and visualization, and how they were overcome through software engineering solutions.

3.3 Organizational Structure Performance Assessment

The analytical infrastructure detailed on the previous section is useful not only to understand the limitations of the empirical airline reasoning processes but also to develop new decision making methodologies to be embedded on our simulation. Although the reasoning accuracy was object of scrutiny, the text did not mention how the other side of our research study, organizational structure performance, will be assessed.

This section intends to suppress the lack of information regarding the organizational structure simulation by presenting the main features of *Brahms*, the modeling and simulating tool that introduces a new human-centered computing paradigm. In order to accomplish this, we will first draw a big picture of the simulation system as a whole justifying the use of certain technologies and pointing out additional contributions to the communities behind those technologies. Next, we will explain how *Brahms* greatly improved the experience of modeling the workflows on TAP (appendix A). Finally, a subsection will also be dedicated to expose some aspects of a visualization module developed to allow a better understanding of the concepts being simulated.

3.3.1 Background and Overall Simulation Architecture

As referred during introduction and incrementally uncovered along the text, the main goal of this research study was to simulate the operational control on a real airline company. Obviously, simply creating a model of such reality and mimic its intrinsic features would be of arguable interest so we aimed at, thereafter, propose changes that would lead to more efficient activities and workflows.

The empirical observations listed on section 3.1, made us aware of the reality in TAP, our case study airline company. We soon noticed that we would be treating a case that falls into the popular business process reengineering paradigm.

Following this, we had to adopt a simulation tool that would simplify the modeling of the concepts related to our airline company while at the same time featured some business process reengineering capabilities. Meeting this requirements was *Brahms* the Business Redesign Agent-Based Holistic Modeling System.

Although some theoretical information was already presented about *Brahms* on the *State of the Art*, it worths point out some technical information about this system for the purpose of clarify certain options or side activities carried out along this study.

First and foremost, *Brahms* is a Multi-Agent System featuring the BDI, *Beliefs, Desires, Intentions*, architecture [2]. While these characteristics are not enough to distinguish it from many other simulation engines, *Brahms* is currently being developed by the *Brahms Team* at NASA Ames Research Center in collaboration with the Carnegie Mellon University and it have been successfully used in NASA's Mission Control, to automate human tasks for the International Space Station. Its source code is proprietary but NASA freely distributes it for research purposes only.

At this point, we simply thought that if *Brahms* was enough for NASA it would certainly suit our needs. After further inspecting the features provided by *Brahms*, we noticed that it was a much more advanced tool than other Multi-Agent Systems that we knew about. It sports its own agent-oriented programming language, adds up some human-centered computing concepts and has its own production rules system.

The characteristics above ought to require an additional effort in implementing our airline company but we decided to take the challenge. Another feature lacking in *Brahms* is the ability to visualize the concepts being simulated. While this functionality was not required to get a quantitative comparison of different organizational structures, it was regarded as an educational and clarifying way of understanding the operations carried out in an airline company.

Being mostly characterized as an academia or scientific tool, *Brahms* lacks a wide user community, where one may get models, code examples or help. Despite of that, it is thoroughly documented and their creators lead a discussion group to assist early-adopters.

As *Brahms* runs in a closed virtual machine, the first contact with the simulator community intended to inquiry about the possibilities of developing a visualization of a running simulation. While the primary approach would be to interpret a set of output files *post*-simulation, *Brahms* features a Java API (JAPI) allowing for environment expansion and control.

Interacting with JAPI would be roughly the same as interacting with a java application therefore we decided that our visualization would be built-in in the analytical infrastructure, described on section 3.2.1, and use the latest advancements in browser technology.

Figure 3.7 depicts the components and architecture beneath the simulation portion of our study. The first noticeable aspect is the mention to the analytical infrastructure. Actually, and recalling figure 3.5, the *BROWSER* component on the diagram above matches the 6th step in the analytical infrastructure articulation process, mainly the *simulation control*. The 6th step is responsible to trigger simulation execution, which in turn reads the cleaned operational files and produces a set of simulated files, flighting activity and operation logs, used in later analysis.

While this section is not meant to be too technical, other aspects of figure 3.7 require further inspection. Starting from the beginning, the human user has the ability to interact with the analytical infrastructure through a *browser*. Given the set of technologies used, it is important to emphasize that at the time of writing, the only browser that supports our infrastructure was *Google*

[2] Software model that implements the main aspects of Michael Bratman's theory of human practical reasoning.

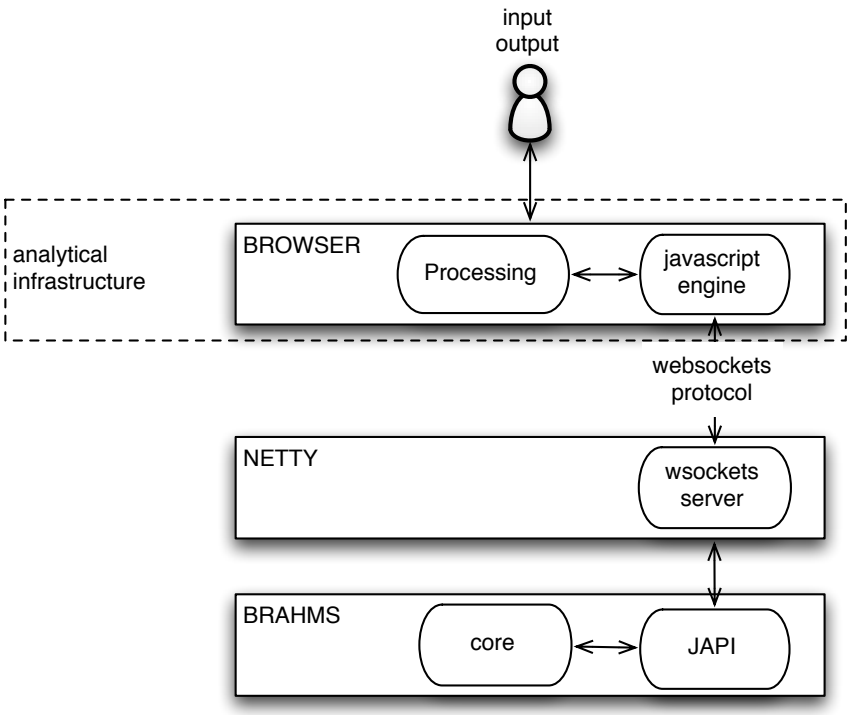

Figure 3.7: Overall simulation architecture and components.

Chrome. Nevertheless, with the fast technological evolution, it is likely in the near future other major browsers start to implement the technical innovations employed by our simulations.

Along with other analytical infrastructure features, besides visualizing the "ad hoc reports" and other analysis, the user may configure certain simulation parameters, as the time window, the number of executions, select the different reasoning modules available, among other. Concerning the outputs, simulations produce reports that are not directly accessible to the user. Instead, level 7 of the analytical infrastructure (see figure 3.5) provides features to read the reports, statistically aggregate the data and display easy manageable charts. As a side note, those charts use the *Google Visualization API*, making them dynamic and recalculated every time a simulation runs or new configurations are requested.

Moving down in figure 3.7, the *browser* portion of the simulation architecture contains the visualization module. It is mainly composed of two interwoven parts: the "javascript engine" and "processing". The former handles all the communications with the "websockets server", discussed soon, decoding the incoming messages and controlling "processing" animations.

Processing is widely used in the scientific and academic field given its ability to create powerful representations of large sets of data. While the original *Processing* is based and to be used with the Java programming language, considering our *browser* requirements we had to use a *javascript* port of the language.

The *BROWSER* component also allows for simulation control, that is, starting, pausing and stop the simulation. The visualization module will be described on subsection 3.3.3, so at this

point it just matters to understand we are in presence of a distributed infrastructure where messages come in, go out and an animation of the simulated theatre is displayed in-between.

The "websockets server" uses *NETTY*, a Java non-blocking I/O socket framework, to implement the recently introduced *websockets protocol* as part of the *HTML5* specification. The use of such technology is solely implemented on *Google Chrome*, thus the reason of our simulations only work with this browser.

The *NETTY* component is too much technical to deserve further inspection. As any server, it establishes TCP connections with remote clients and then exchanges messages with the Java API of *Brahms*. Besides a message gateway, it shares some similarities with the "javascript engine" as it decodes and encodes the messages exchanged with the simulation core.

The next subsection, 3.3.2, will be solely concerned the next component, *BRAHMS*. As we referred, the selected simulation engine to implement our organizational structures features an agent-environment, "core" and a Java API, "JAPI". The former is more concerned with agent, geography, activities and other real entities modeling; the latter, is more technical, being used for handling java objects or other services.

To conclude, it worths pointing out an important contribution to the *Brahms* community made along the research study herein documented. Given the need to explore *Brahms* to its full extent, and attending to scarcity of examples over the internet, we decided to publish a set of tutorials in the *Brahms* discussion group. This initiative was warmly welcomed by *Brahms* creators and a transcript of such articles may be found on appendix E.

3.3.2 The Simulation Module

This section is focused on the *BRAHMS* component of the simulation architecture (figure 3.7). We will start by describing the "core", that is how we used *Brahms* formalisms and programming language to implement the empirical observation exposed on section 3.1. As stated, the "JAPI" side is more technical, therefore we will not delve deep into it, solely pointing its use as mean to solve some *Brahms* shortcomings.

Recalling section 2.3.2 of the *State of the Art*, *Brahms* supplies a number of human-centered structural formalisms to help modeling real entities. Thus, one of the first steps in modeling a scenario with *Brahms* is to make a correspondence between real and artificial concepts. Table 3.17, intends to clarify our approach concerning such mapping.

Besides presenting a number of associations, table 3.17 also hopes to illustrate the expressiveness offered by the *Brahms* modeling language. Although comprehensive it just contains a subset of *Brahms* concepts.

The first column respects to the *reality*, the next two contain the name of virtual entities implemented on our simulations. Starting by "Facilities and Locations", *Brahms* is very complete in what concerns geography modeling. There are *areas* and *areadefs* and we may not have the former without the latter, that is, first the area must be defined, we must specify what it is then we may name it. Looking at the "ACT" example, we first had to create a generic "ACT" extending the *Building* definition shipped with *Brahms* and then we were able to define "LisbonAirportACT" as

Table 3.17: Concept mapping between reality and *Brahms* formalisms (caption in table 3.1).

Reality	*Brahms*	
Facilities and Locations	*area*	*areadef*
ACT	LisbonAirportACT	ACT (Builing)
AP	LisbonAirportAP	AP (BaseAreaDef)
CI	LisbonAirportCI	CI (Building)
HCC	LisbonAirportHCC	HCC (Building)
LIS	LisbonAirport	Airport (BaseAreaDef)
OCC	TapOCC	OCC (Building)
Computerized Systems	*object*	*location*
AMS	HCC_AMS	LisbonAirportHCC
	OCC_AMS	TapOCC
CTS	OCC_CTS	TapOCC
DOV	ACT_DOV	LisbonAirportACT
(AS)	Lisbon_AS	(World)
Human Collaborators	*agents*	*groups*
as	AircraftSpecialist	OCCSpecialists
cms	CrewMember	CrewMembers
cs	CrewSpecialist	OCCSpecialists
fd	FlightDispatcher	TriggeringAgents
gs	GroundSupervisor	GroundPersonnel, TriggeringAgents
hs	HccSupervisor	ApprovingAgents
mss	MaintenanceMan	GroundPersonnel, TriggeringAgents
os	OccSupervisor	ApprovingAgents
pss	PassengerMan	TriggeringAgents
ss	StationSupervisor	TriggeringAgents

an instance of it. In the case of "AP", Aircraft Parking, as it is not a *Building*, we had to choose the *BaseAreaDef* as parent.

Our naming conventions reveal another *Brahms* feature not foreseeable in the table, the *part of* construct. Listing 3.3 shows an excerpt of the *part of* and *path* formalisms.

Listing 3.3: Excerpt of *Brahms* area and path definitions

```
area LisbonAirportAP instanceof AP partof LisbonAirport {  }
path LisAP_to_from_LisHCC {
   area1: LisbonAirportAP;
   area2: LisbonAirportHCC;
   distance: 600;
}
```

As it is clear, besides specifying what the area is, we may also specify the relation between areas (*part of*). Figure 3.1, shows the Aircraft Parking inside the Lisbon Airport and *Brahms* allows such modeling. Another very convenient feature is the *path*. It defines a relationship between two areas not in terms of composition but geographical dispersion. Again, something very handy to set the distances (in seconds) between buildings or areas. According to table 3.4, the distance, on average, between the Aircraft Parking and the Hub Control Centre is, by feet, 10

minutes, so we must specify it as 600 seconds. As we will see later, if our agents use a vehicle, and thus only spend 2 minutes, the 600 seconds time may be overwritten in the *move* activity.

Moving on to the next portion of table 3.17, it is about computerized systems. To model inanimate things, *Brahms* offers the concept of *object*. Actually, some real world objects might be modeled as agents because while they are physically inanimate, they may be used to reason over facts and therefore help humans take decision, e.g., a computer. The notion of *agent* in *Brahms* is a little narrower than other multi-agent systems because *objects* might also react and reason as agents. Apart from the naming convention used for systems, which is irrelevant, another key property is its location. In *Brahms* every *object* and *agent* might be given a location. Again, according to figure 3.1 the airline systems were distributed across different locations.

Concerning the human collaborators, and as stated above they were modeled as *agents*. As a multi-agent system, this is no surprise. The innovative factor in *Brahms* is the existence of groups. When implementing an agent, one may use the *memberof* keyword to set its group membership. For instance, In table 3.17, the "AircraftSpecialist" and the "CrewSpecialist" are members of the same group, the "OCCSpecialists". This is a very powerful feature in Brahms because when need to implement activities to be performed by the agents, we just need to implement them at the group level, then the activities are automatically inherited.

Before introducing *Brahms* activities, we may not skip the "Lisbon_AS" object. The "AS" systems stands for *Airport Screen* and during our interviews with airline personnel nobody noticed its existence, thus the reason for appearing between parentheses. It is here to illustrate a simple case where a modeler needed to use a workaround to simplify or make it computationally feasible to mimic reality.

Recalling our empirical scenario, when some agents detected anomalies they would trigger workflows. In table 3.17 they appear as members of the "TriggeringAgents" group. In reality, they perceive anomalies in the course of their activities: verifying an aircraft, checking-in passengers, loading cargo, and so on. But in our simulation we only had files with those anomalies. The closer approach would be to put those agents all reading the file and checking if they were responsible to trigger the next anomaly. While it would be correct to do that way, it would not be wiser because it would put too much strain on IO operations to read the same file (or checking the same list), over and over again.

Following this, we created the *Airport Screen* system that roughly mimics those screens found at the airports with the next departures or flight delays. It reads the file and "tells" the agents about upcoming anomalies. Now that hopefully the notion of auxiliary object was explained another topics worths discussion: how does the Airport Screen tell the other agents?

Answering this question definitely proves that *Brahms* is a fairly different multi-agent system founded on a totally new paradigm. As any other programming language, the *Brahms* agent-oriented programming language also supports the primitive types, such as integers, characters, etc. and the map collection. Unfortunately, it does not support lists, a major flaw that had to be overcome through the use of JAPI, a workaround explained soon. While *Brahms* supports those data types, agents and objects are unable to directly handle them. At this point is very important

to underline that *Brahms* is human-centered and human beings do not act or reason upon integers, they do that according *facts* or *beliefs*. This is where the BDI software model enters and somehow distinguishes *Brahms* from the majority of multi-agent systems.

Now that *facts* and *beliefs* were introduced, when our *Airport Screen* detects an anomaly it creates a *fact* or *belief*. As it is located in the "World", a *Brahms* abstraction to everywhere, all the agents or objects perceive such fact/belief. It is up to the modeler to implement the activities they must perform, if any, when they detect the fact.

To better illustrate the human-centered paradigm of *Brahms*, listing 3.4 contains a purportedly oversimplified code excerpt. It shows a routine that every minute checks the flight list using a Java object (more about this later) and concludes a fact, *triggerConcept*, represented by the *conceptid* string. In the *TriggeringAgents* when the *triggerConcept* fact matches the *conceptCode* the agent does something.

Till now we presented an overview about how we modeled facilities, systems, and collaborators. The next step is to summarily expose the *Brahms* formalisms concerned with activities.

Listing 3.4: The *Brahms* human-centered paradigm

```
// in Lisbon_AS...
repeat: true;
when(knownval(current.currMinute > current.cfMinute))
do {
   string conceptid = as.checkFlights();
   conclude((current.triggerConcept = conceptid), bc:0, fc:100);
}

// in TriggeringAgents group...
when(knownval(Lisbon_AS.triggerConcept = current.conceptCode))
do {
 ...
}

// in GroundSupervisor agent...
initial_facts:
   (current.conceptCode = "gs");
```

Revisiting figure 3.2 we identified six main activities: communicate (by radio and phone), data write, data read, reasoning and approve. To these six, let's add a new one that will be used on our future organizational structure proposals: move between locations.

Brahms supports multiple activities, one of them is the Java activity. The Java activity will not be discussed into detail but it worths mention because it might be regarded as executing a Java method, with inputs and multiple return values. As such, it virtually allows *Brahms* to achieve anything possible with Java. For instance, the activity of checking flights on listing 3.4, which required to read from a file, could have been implemented using a Java activity.

Other types of activities, more relevant to our study were the communicate and the move activities. Concerning the former, what we knew was that certain airline operators, in presence of

an anomaly, would pick the radio or phone and communicate such fact to a supervisor. We also knew that such activity would consume an indefinite amount of time (refer back to table 3.3).

As in other systems, there are always a number of ways to implement the same scenario and our simulation was no exception. There would be several ways of communicating a disrupted flight but in our case we opted by the flight number. Ideally, it would have been better to pass a Java object because, as we will see soon, our flights were implemented as such. The problem is that agents in *Brahms*, as human counterparts, are solely capable of transmitting *facts* or *beliefs*, usually represented through primitive data types.

Listing 3.5 intends to show how easily *Brahms* makes the transmission of *facts* and *beliefs* across agents. The excerpt presented is, again, part of the *TriggeringAgents* group, therefore it will be inherited by multiple agents, each one with its own recipient. To surpass this issue the communicate activity showed resembles a function where the "with" field is variable.

Still on listing 3.5 the "about" field indicates the fact or belief to be sent, in this case the *disruptedFlightNumber*. Once in possession of the fact or belief, the *recipientAgent* may act or reason upon it. Last but not least, the activity duration. By asserting the "random" property as true, we want *Brahms* to pick a value between the "min_duration" and the "max_duration". Those values were taken from table 3.3 of the real activities duration and simply converted to seconds.

Listing 3.5: The *Brahms* communicate activity

```
communicate reportDisruptedFlightByPhone(BaseGroup recipientAgent) {
   with: recipientAgent;
   about: send(current.disruptedFlightNumber);
   random: true;
   min_duration: 240;
   max_duration: 480;
}
```

The way *Brahms* handles activity timing was of uttermost importance for our study. The other activity types benefit from the same random approach and therefore the previously seen "distance" in geography *paths* (see listing 3.3), may be overwritten using a *move* activity.

The *move* activity is not much different from the *communicate* activity, instead of a "with" and "about" properties, it has a "location" property telling the agent where to go next. The motion takes a certain amount of time that might be random, as in listing 3.5 or static, asserting "random" as false and providing a "max_duration".

Before moving to the JAPI component of the simulation architecture (refer back to figure 3.7), a brief word goes to *Brahms* classes. Along with *areas*, *objects*, *groups* and *agents*, *Brahms* also supports *classes*. The problem is, these classes are not as powerful as the Java counterparts. Actually, they use the same *Brahms* agent-oriented programming language syntax, and the same human-centered paradigm. Therefore and simply put, *classes* are to *objects* as *groups* are to *agents*. We did not list the *classes* in table 3.17 as there is solely one, the *TriggeringObjects* that works in a similar fashion than *TriggeringAgents*.

Till now described our approach in what concerns the modeling of the most visible concepts and activities using the *Brahms* proprietary agent-oriented language. Although we recognized how expressive, distinct, innovative and somehow powerful it is, we must also underline its shallow learning curve and, as we will see next, the lack of some widely used data types and support functions.

As we stated previously, *Brahms* supports several primitive data types and *maps*. Unfortunately, *lists* are not available and they are one of the key data structures to store our flights. Even the *flight* object, which composed of several attributes, such as scheduled departure date or flight number, would be much better abstracted by means of plain Java objects.

To address such issues *Brahms* provides two options. The latest *alpha* version allows for direct Java objects manipulation. Older versions support already mentioned Java activities. In one case or the other, there are some conventions one should respect but in the end is roughly like calling a static java methods.

Without getting into much detail, in the implementation of our simulations, the *Brahms* JAPI was used in several scenarios. First and foremost to store within *ArrayLists* our *flights* and *delays* objects. Second to perform file input and output, operations not supported at the *Brahms* level. Third to implement the Specialists reasoning activities. Last and fourth to stream the ongoing events to the "websockets server", see figure 3.7.

To conclude, it worths emphasizing that this section did not aim at thoroughly describe the implementation of our simulation using *Brahms*. That would require a technical manual as long as this report. The intention here was to present the human-centered nature of *Brahms* and how that paradigms fit the reality being modeled.

3.3.3 The Visualization Module

As a side goal, the visualization module was not required to produce the answers to the main goals of this research study. It simply receives some messages from the *Brahms* component, such as which activity is being carried out by which agent, and displays an animation of the simulated theatre. Therefore, the main purpose of the visualization was to provide an educational tool to allow people to learn how the airline operations management work and to better understand the proposed organizational changes.

As it was explained in subsection 3.3.1, the visualization module uses *Processing.js* to render images and animations on the recently introduced *HTML5 canvas*. It is tightly connected to the *javascript* that decodes the messages coming from the simulation.

The visualization is composed by two distinct areas, the operational area, very similar to figure 3.1 and an airport screen. The former its where the main action takes place, through arrows we may observe the current workflow state. The latter provides visual hints about flight departures and state. The airport screen lists all the flights within a future time frame, if a flight suffers an anomaly it is depicted in a different color and a workflow is triggered in the operational area. Assuming it was the Ground Supervisor to detect the anomaly, and its next activity is to notify the

HCC Supervisor, then an arrow is displayed between him and the HCC Supervisor with a visual indicator of a radio communication.

Without being too much technical, the underlying architecture of the visualization had to closely implement the concepts introduced by *Brahms*. This means we had to implement classes to represent the agents, the objects, the area definitions and so on. While requiring an additional effort, such approach also allows for a flexible display.

Given the need to represent several distinct organizational structure, the visualization had to be dependent on the simulation. At the beginning, a list of the *Brahms* concepts is passed to the visualization so they can be displayed. Other features are also present such as *onMouseOver* actions that return further information about the concepts and so on.

To conclude, a final words goes to the amount of *Processing* code required to implement solely one action or even *Brahms* concept. We must keep in mind that behind a *Brahms* concept abstraction there is a complex and large code base, therefore the need to execute the *Brahms* environment in a virtual machine. The problem is that we lack such constructions in *Processing* and we are required to implement them by hand. Following this, to implement every activity or concept is a lengthly process, the reason why the visualization will always be less expressive than the simulation itself.

Chapter 4

Results and Analysis

The practical concepts presented in the previous chapter aimed at deepen the knowledge over the approaches taken along this research study. First, we started by understanding the organizational structure at TAP, the more relevant business processes in airline operations and how TAP records its operational information. After these empirical observations, it was also exposed the role of the real operational data on decision making improvements at simulation level. This step counted on supervised learning techniques from the Artificial Intelligence discipline to replace probabilistic reasoning by the decision tree counterpart. Finally, the adoption of *Brahms* as a Multi-Agent modeling and simulation system, was promoted as a way of boosting business process reengineering.

At this point, one should have an in-depth understanding about not only the goals of this research study but also the methodologies used to accomplish them. With that in mind, this chapter gathers the answers to the most prevalent questions raised in the beginning by presenting and discussing the final results obtained. To do so, it is split into three sections. The first analyses the experiments carried out, such as which data was used or which scenarios were tested. Next, the conclusions about reasoning accuracy will be depicted by comparing the different reasoning approaches outputs with real-life counterparts. Finally, the organizational structure performance of TAP will be evaluated based on its time efficiency and workload balance.

4.1 Experiments

Throughout this document, the importance of using operational data from a real airline company has been promoted not only as a source of increased confidence on the future results but also as the origin of some troubles. Following this, the first subsection bellow will be dedicated to present not the format of the input data, a subject covered on section 3.1.5, but the substance of simulation inputs. That is, which flights were used to produced the results to be unveiled on the next sections.

Next, the subsection 4.1.2, will briefly discuss some experiments with classification algorithms or alternative features selection during the decision making improvement phase. It is mostly regarded with WEKA data mining tool settings and reports analysis.

Finally, and converging to the most relevant goal of the research study documented along this text, some current operational workflows on TAP will be subject to minimal transformations that will hopefully make the overall organizational structure score higher after a set of metrics that will also be explained.

4.1.1 Simulation Input Data

As referred since the beginning, we knew in advance that our research study would require the imitation of what takes place in an airline company. Following this, a simulation would be the most scientific approach to our problem, so we took the challenge and adopted a targeted tool to do so, as described in section 3.3.

After implementing a simulation, another point requiring debate was which data would be used to feed the simulation. In other words, what would be the inputs after which our simulation would generate the outputs. This topic was of uttermost importance because in an organizational structure not all the business processes assume the same prevalence, there are workflows that take place a higher number of times than others. Using random data would never be the same, and the trust on the final results would be compromised.

Getting access to real airline operational information beyond the advertised by travel agencies or online tickets shops would be impossible due to the data confidentiality featured by the majority of companies, not only in the airline field. Fortunately, the collaboration with TAP, the major portuguese commercial airline, allowed us to use real *pre* and *post* operational flights.

In what concerns data logging, format and analysis, sections 3.1.5, 3.2.2 and 3.2.3 already provide in-depth information. What remains is characterizing the data used to generate the results, that will be displayed shortly, of our research study.

As we saw, the logging mechanism purportedly implemented by TAP in order to fulfill our requests, performed on a daily, weekly and monthly basis. This means that we would not be able to request flights from day x to day y and data would always come in *chunks* of one day, week or month. Merging *chunks* together would not be reasonable, as it would cause the repetition of some flights, something far from desirable.

When considering the amount of data to be analyzed, we had to keep in mind other shortcomings such as the time of execution and memory consumption. For start, the *Brahms* Agent-Environment that interprets the simulation code, works as *Virtual Machine* inside the *Java Virtual Machine*, making computations slower by several order of magnitude. Another factor adding up to the increasing execution time is the need of run the simulation several times to mitigate the effect of the probabilistic nature of reasoning algorithms and *Brahms* activities time intervals.

Accounting for all the aspects above, and after some experimental evaluation of different data set sizes, we decided to use the flights spanning across one week. We observed that one week of regular airline operations, without personnel strikes or extraordinary adverse conditions, was

enough to determine a proportional pattern in results with a satisfactory running time. Using more records would increase the computational time, without noticeable variations in results, relatively speaking.

To be more precise, our simulation was fed with the flights operated by TAP from 15th February 2010 to 21st February 2010, a whole seven days week of activity. As mentioned before, some records had to be excluded according to different criteria. Figure 4.1 points out some interesting intersections between files to illustrate data set reduction.

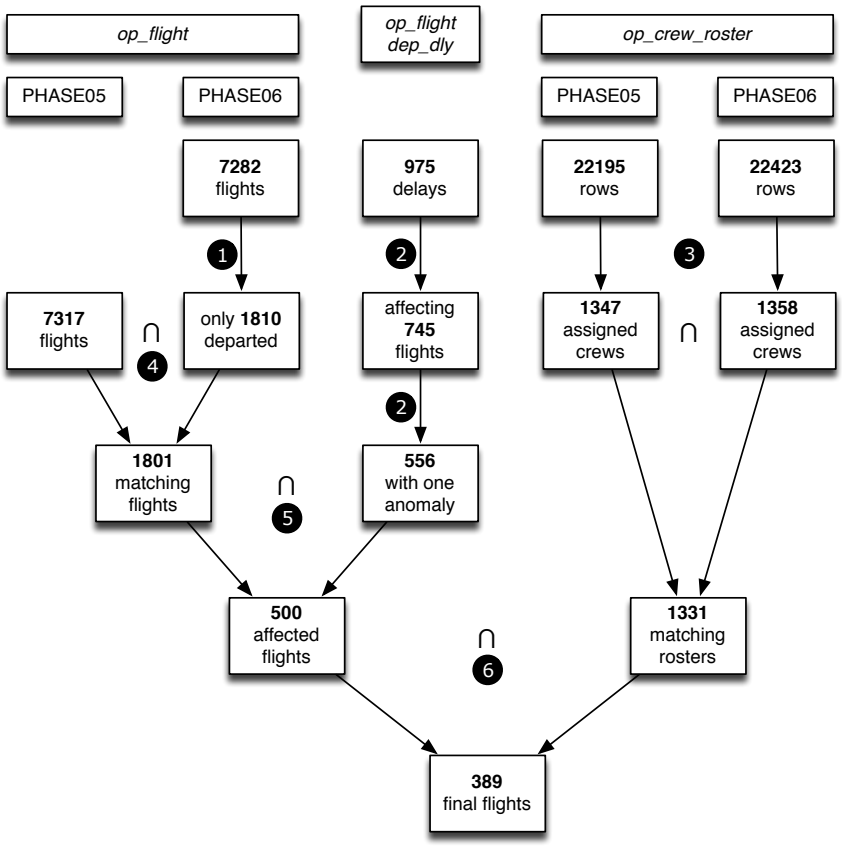

Figure 4.1: Operational files intersection and data set reduction. Extended caption on table 4.1.

As it is immediate to observe and without going into much detail, our initial data set started with a significative number of flights that was reduced by means of data clearance and intersection. The process of file joining and flights reduction is signaled with black numbered dots that are described on table 4.1, but by the end of the process we ended with 389 flights that triggered workflows or decisions during our operational simulation.

In general terms, figure 4.1 starts with the three kinds of files provided by TAP divided by the weekly phase identifiers (see section 3.1.5). "PHASE05" corresponds to *pre* operations, "PHASE06" translates *post* operations. At the beginning we had a number of flights, delays and crew member scheduling that started to be excluded after some criteria and the sets intersected

with each other. Orphan entries were not accounted for as they could introduce errors or discrepancies. In few words, we may say that we only cared about flights that really occurred, with precise aircraft and crew rosters signaled, that suffered one and just one anomaly before departure. Table 4.1 thoroughly explains the points of intersection and exclusion.

Table 4.1: Operational file intersection and exclusion points (cross-referenced with figure 4.1).

1	For some indefinite reason, from the 7282 flights listed on the *post*-operational file, only 1810 had the ACTL_OFFBLK_DATE field (actual departure date) filled with meaningful dates, so we had to exclude 5472 putative flights.
2	There were 975 delays reported on the *op_flight_dep_dly* file but, as referred on section 3.2.3, some flights had more than one delay reported. Those with just one anomaly totaled 556.
3	The crew roster files contained much unneeded rows containing, for instance, crew member days off or vacations. Every crew roster involves between 5 and 10 crew members so in the end we had 1347 and 1356 flights with crews assigned.
4	Intersecting the scheduled flights with those who really took place, gave us 1801 flights. This means that there were 9 flights that we could not analyze as we did not their scheduled aircraft or crew.
5	Crossing the flights with complete information with the disrupted flights gave 500 flights. Unintentionally we discovered 56 delays in non-scheduled flights.
6	Finally, and once again, intending to only process complete and consistent records, we intersected the flights that suffered one anomaly with the available crew rosters.

To conclude, two points worth mention. The first, less relevant, is concerned with simulation feeding. Given the visualization module developed, and in order to provide and educational perception about the operational control on airline companies, the simulation is really inputted with the 1801 scheduled flights. Therefore, the flights that did not suffered anomalies, are only displayed to the user and do not trigger any reasoning or workflow processes, not accounting for statistics. Next, despite only 389 flights contributed to measure reasoning accuracy and workflow performance, this number is enough to produce trustful results in a timely manner. As stated, experiments were made with a larger number of records and, in relative terms, results were similar.

4.1.2 Reasoning Algorithms and Classification Features

Recalling subsection 3.2.4 about data classification and reasoning code generation, we presented an infrastructure geared towards producing input files for the WEKA data mining tool and convert its outputs into Java code. Later, this Java code would be migrated to our simulation in order to take decisions after the 389 disrupted flights mentioned in the previous section.

This side goal of this research study aimed at better mimic the decision processes currently seen on TAP. As explained on section 3.1.4, our case study airline company provided probabilistic tables with the actions take after a specific anomaly category.

At this point, the first step of our research was to obtain the decision based on those probabilistic tables. Then, we compared the solutions outputted by our simulation with the real life actions, gathered during the operational data analysis stage (see section 3.2.3).

Following this, the reasoning accuracy results that will be presented shortly will be twofold. First, the solutions obtained after the probabilistic tables provided. Second, the solutions based on supervised learning techniques.

Concerning the first, there are two scenarios to consider based on TAP or IATA delay codes. As mentioned early, we may use TAP or IATA delay codes as putative anomalies suffered by the flights, and although some similarity, these sets of codes are different in specificity and number. Another peculiar aspect related to the empirical probabilistic action is its randomness, therefore we run the simulation 100 times trying to mitigate the effect of casual actions. It is noteworthy to mention that considering the symbolic nature of the actions, the aggregate function used to extract an action from the 100 was the maximum number of occurrences, instead of the average.

Now moving to the supervised learning techniques, as we saw on subsection 3.2.4, there were five features to be used in the classification process: *tapcode*, *iatacode*, *delay1*, *delay2* and *delay3*. Eventually other features could have been calculated but we intended to keep the logic behind the airline operations control. If on the empirical probabilistic action we solely used the *tapcode* or the *iatacode*, during classification we made some experiments mixing one or two delays as if they were operational delay estimations. For instance, assuming that a technician diagnoses a failure on the air conditioning system of an aircraft, the deciding Specialists may ask him how long will it take to fix the problem. This kind of delay estimation on advance is perfectly valid from an operational perspective and adds value to the classification process.

Concerning the classification algorithms and despite the broad offer of the WEKA data mining tool, we focused on the *C4.5* and *Random Forrest* algorithms. The reason was the popularity of the first and the good results obtained with the second. Here is important to point out that other algorithms were briefly tested using the flexibility of WEKA, although their inferior or identical performance prevented them to be included in our results. One must bear in mind that the goal was to improve reasoning accuracy through the use of supervised learning techniques and not to perform a comparative study across different classifiers.

Still on the supervised learning, it is desirable to mention that the results presented respects to decision tree implementation and input at the simulation level. This means that we got the WEKA formal description of the different decision trees, it was converted to Java code, merged into the simulation and then results were obtained through the simulation. Albeit this approach, some cross-validation statistics will be briefly discussed.

To conclude, a final word goes for the evaluation of the reasoning accuracy. For both scenarios, empirical probabilities or decision trees, the process is the same. There is a list, similar to listing 3.1, with the actual actions taken by the Operational Control Centre Specialists (last column). During each simulation run, an output file is produced following the same conventions (see table 3.16). Thereafter, the analytical infrastructure described in the previous chapter has tools to traverse both listings and return the number of *matches* between the real and simulated lists.

4.1.3 Operational Workflow Transformations

Assessing the impact of the organizational structure on airline operations is the major goal of this research study. As such, and recalling section 3.1, *Airline Empirical Observation*, we collected a number of elements that allowed us to simulate a real airline company.

At the center of our simulation lies the operational workflows gathered on appendix A, they represent the activities undertaken by airline personnel to carry out certain tasks. Other relevant participants are the time required to complete those activities (see table 3.3) and the concepts, human collaborators or automated systems, that trigger the aforementioned workflows.

Following this, the metric used to assess organizational structure performance was time. Actually, and as we will see shortly, time is enough to estimate other direct metrics such as collaborator *stress* and *well-being* or indirect ones like personnel cost or phone bills. While the former were analyzed, the latter were not. Again, we preferred to stay away from random values and therefore calculating costs was impossible due to the lack of accurate data.

As we saw on the previous subsection, the reasoning accuracy study was mostly a comparison between empirical probabilistic actions and machine learning techniques counterparts. Here, we follow a similar criteria. We will start by assessing the current performance of TAP's operations control structure and thereafter we will evaluate three incremental transformations of the real organizational structure.

Also in a similar fashion to reason accuracy, the empirical workflow scenario will be assessed using TAP and IATA delay codes. In order to detach observations from simulation results, it worths presenting at this point two diagrams comparing the distribution of workflows analysis per simulation execution. As clarified on section 3.1.2 the workflows of appendix A are triggered by airline employees or automated systems.

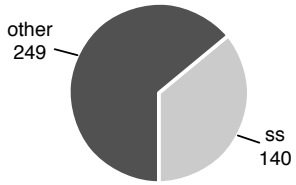

Figure 4.2: Station Supervisor versus other concepts workflow distribution.

As we saw on chapter 3, the Station Supervisor differ itself from other concepts because it triggered a specific operational workflow independently of the nature of the anomaly reported. Figure A.4 depicts the workflow triggered by the Station Supervisor from a foreign base and figure 4.2 above illustrates the number of such workflows in our simulation. As it is possible to observe, the number of delays reported from airports other than Lisbon International Airport, the home base, correspond to 56% of the anomalies reported. Although it is not a high number given the amount of airports to where TAP flies, we must keep it in mind for later performance assessment discussion.

Excluding the anomalies triggered from foreign bases, solely concerned with Station Supervisor, it now matters to draw a picture about the concepts responsible for reporting anomalies at Lisbon International Airport. Figure 4.3 illustrates such distribution.

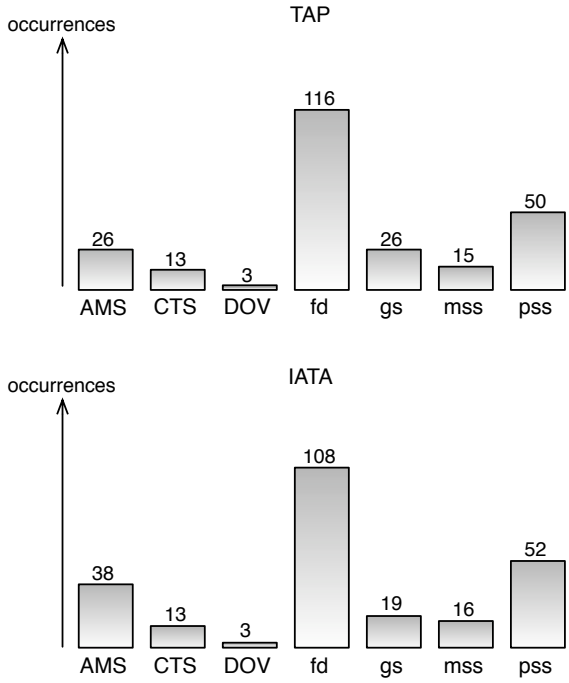

Figure 4.3: Home base concept workflow distribution (TAP/IATA). Caption on table 3.1.

It is very important to emphasize that from now on, and in order to preserve the familiar short notation, the diagrams comparing different concepts use the labeling on table 3.1. Therefore, the higher number of workflows, 116 and 108, are triggered by the Flight Dispatcher, labeled with "fd".

Proving a consistent manual classification, the TAP and IATA distributions on figure 4.3 do not differ substantially. Summing up the number of occurrences will give 249 anomalies which in turn, summed with the 140 triggered by the Station Supervisor, totals for the final 389 anomalies on figure 4.1.

Now that an overview about the anomaly distribution across concepts was provided, it remains to explain the experiments carried out during our research. As stated, we propose three distinct incremental and slightly different organizational structures. Along with the real operational workflows, the proposed ones are gathered at section A.2 of appendix A. The new sequence diagrams are captioned with *proposed (I)*, *proposed (II)* or *proposed (III)* depending on their nature.

Before explaining our proposals, a brief mention goes for new activities and symbols introduced. Adding up to the actors and activities illustrated on figure 3.2 there is now a symbol to represent a "move" activity and another for an Aircraft Movement System with mobile computing support. Figure 4.4 presents the icons associated with the new entities.

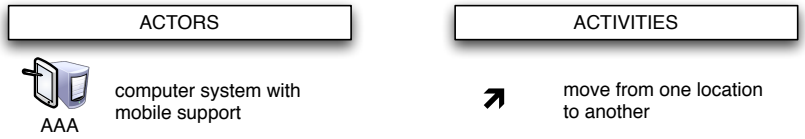

Figure 4.4: New operational workflow entities at TAP.

When proposing new organizational structure scenarios, we kept in mind their feasibility. We did not want to force great performance improvements at the cost of affecting the successful control of operations. This way, we proceeded to small changes never related with operational inputs or outputs, that is never altering the triggering or deciding entities. Taking this approach, and considering the set of TAP collaborators or computerized systems, our initial choice was to remove the Hub Control Centre Supervisor from the operational theatre.

Looking for the current sequence diagrams on appendix A, it is possible to observe that the HCC Supervisor usually plays as information distributor, only assuming a supervising position when facing anomalies related to Passenger Services. Removing the HCC Supervisor required three major changes in the sequence diagrams. First, the Ground Supervisor and Maintenance Service that previously communicated with HCC Supervisor via radio were now required to move to the Hub Control Centre in order to fill a form on the Aircraft Movement System. Second, the decisions taken by Aircraft and Crew Specialist, previously returned by radio through the HCC Supervisor, were now communicated by the OCC Supervisor via phone. Third, the decision carried out by the HCC Supervisor after Passenger Services anomaly reporting was migrated to the OCC Supervisor. All the above changes are clearly illustrated on figures A.9, A.10, A.11, A.12 and A.13 of appendix A.

At this point, it is desirable to mention that the new sequence diagrams on appendix A, only translate the new business processes, therefore, the old TAP operational workflows where the HCC Supervisor did not take part, remained unchanged and were not duplicated.

As we will see, our proposal *(I)* had a negative impact on TAP operational performance, so we analyzed the results and proposed a second scenario with minor changes. Our proposal *(II)* departs not from the current TAP organization structure but from our first proposal. We now suggest to add *intranet* mobility support to the existing Aircraft Movement System, making it easily manageable through a wireless smartphone or laptop computer inside the airport.

Figures A.14, A.15, A.16 and A.17 evidence or proposal *(II)*. It is important to emphasize that proposal *(II)* only introduces *intranet wireless* interactivity, thus only the Ground Supervisor and the Maintenance Services benefit from in. These restrictions opens the door o proposal *(III)*.

On our third proposal, the Aircraft Movement System is fully converted from a terminal with *intranet wireless* to a *internet-based* system accessible from everywhere. The airline operators that would benefit from the new AMS, were instructed to properly handle the system. These include the already seen Ground Supervisor and Maintenance Services and now the Flight Dispatcher and Station Supervisor. Figures A.18 and A.19 represent the new changes in operational workflows.

To conclude, a brief word goes for the random nature the activity duration and consequently the execution of the simulation 10 times to mitigate the appearance of extreme and casual results. This number was enough because the inherent amount of anomalies, and thus workflows (see figure 4.3) already balance the results obtained.

4.2 Simulated Decision Making Discussion

On subsection 4.1.2 we described the experiments carried out to obtain the results that will be listed shortly. As referred, and in a similar fashion to what we will find in organizational structure assessment, there are two broad comparisons that could be made. The first respects to the empirical and probabilistic decision making processes, based on historical statistics and unveiled by TAP (refer to subsection 3.1.4). The second is related to the application of supervised learning techniques to the operational data sets supplied in order to extract reasoning algorithms (see subsection 3.2.4).

Each of these sides has its variants. In the empirical, one may use the TAP or IATA delay codes manually classified according to a set of TAP categories (refer to table 3.7); in the machine learning process the variants may be obtained by changing the feature selection or using a different classifier.

On section 3.2.3 we thoroughly presented an approach to compare the *pre* and *post* operational data provided by TAP and extract the solutions proposed in real life. At this point, we emphasized several inconsistencies in the data and alerted for the impossibility of inferring certain resolution methodologies from the data set at hands (figure 3.6).

In order to better understand such problems and establish a control to group to compare against our reasoning algorithm results, figure 4.5 illustrates the distribution of the real solutions.

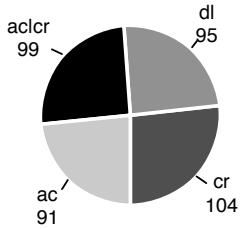

Figure 4.5: Distribution of the real reasoning processes solutions. Caption on table 3.16.

As we conveyed on subsection 3.2.4, the data provided solely allowed us to determine crew changes (“cr”), aircraft changes (“ac”), crew and aircraft changes (“aclcr”) and flight delay (“dl”). Coincidently, the real solutions found for the 389 disrupted flights that will be inputted into the reasoning algorithms, are almost evenly distributed with a maximum of 13 flights difference between crew changes and aircraft changes.

This list of flights and solutions behind the pie chart on figure 4.5, will be used to verify our artificial solutions. Starting with the empirical reasoning algorithms, figure 4.6 displays the solution distribution considering TAP or IATA delay codes.

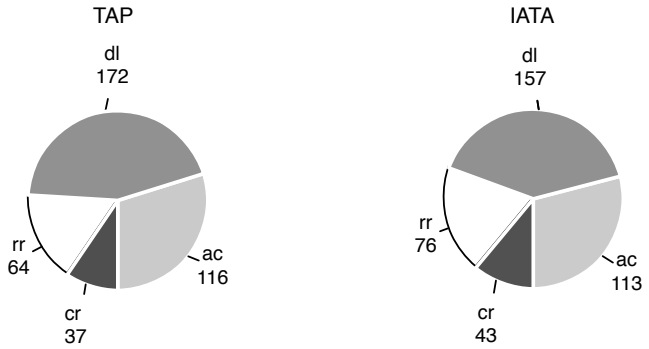

Figure 4.6: Distribution of the empirical reasoning processes solutions. Caption according to table 3.16 and "rr" stands for re-routing.

The two most noticeable differences when comparing the empirical outputs with our control distribution is the absence of the "ac|cr" solution and an unrecognized "rr" solution. The former is related with the incomplete information provided by TAP. Recalling subsection 3.1.4, no details were provided about the articulation between the decision making processes of Aircraft and Crew Specialists and therefore we were unable to implement such relation in our algorithms. The later, "rr", which stands for re-routing seems to be a popular (according to TAP statistics) solution employed by the Specialists but unfortunately it is untraceable from the provided operational data, thus it is absent from figure 4.5.

Given the impossibility of tracking the re-routing operational action, we may associate an uncertainty level to the outputs of the empirical reasoning algorithms. According to figure 4.6, when using TAP delay codes there is a **16.5%** uncertainty in the solution matching, and regarding IATA delay codes this value raises for **19.5%**. In few words, these numbers translate the amount of anomalies that will be putatively solve through re-routing and therefore will not be able to be validated.

Regarding the empirical decision making algorithms, the similarity with the real solutions were around **19%** if using TAP delay codes and **22%** in the case of IATA delay codes. Despite these low matching rates, we must keep in mind the associated uncertainty.

Starting from these considerably lower correspondence between reality and the empirical probabilistic methods provided by TAP, we tried to use decision tree classifiers to improve the solution matching. At this point, it is important to briefly justify our option for decisions trees. Unlike Neural Networks or Naive Bayes, decision trees are straightforward for humans to understand. Since we were trying to improve the reasoning algorithms related to be simulated by human agents, it would make sense to implement simple constructions. Other reason is related to the use of the occurred delay as an estimated delay. This simple approach may also be explored to improve the

real decision processes seen at a real operational control centre, being easily ported to the actual human reasoning practices.

Considering the algorithms used, we started by solely focus our attention on the widely know *C4.5*. Thereafter, and given the interface provided by WEKA to test other decision tree based algorithms, we run other tests. A distinguishable performance was obtained by *Random Forrest* so we decided to also include the results for this last algorithm.

Figure 4.7 shows the distribution of solutions using the *C4.5* algorithm as classifier.

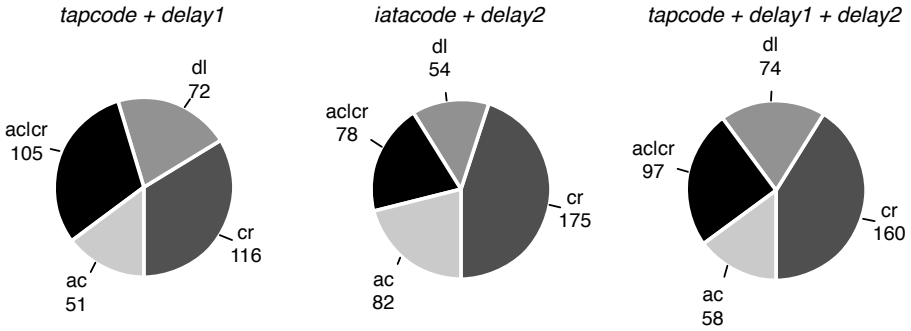

Figure 4.7: Distribution of the *C4.5* artificial reasoning processes solutions. Caption according to table 3.16.

As expected, the *re-routing*, "rr", action does not show up which translates an inexistent uncertainty level. All the anomalies are solved using a decision possible to verify after the *pre* and *post* operational data analysis. Ideally, such absence of uncertainty would be converted into a solution matching increase between 16.5% and 19.5%, the uncertainties respecting to the *re-routing* action calculated for the empirical probabilistic algorithms.

From the charts above it is also possible to infer the feature selection carried out when using the *C4.5* algorithm. In the first experiment, *tapcode* and *delay1* were used, the second used *iatacode* and *delay2*, and the last included *delay2* to the first experiment. In terms of distribution, the first experiment is the one to translating a closer proximity with the real distribution (see figure 4.5). Other feature selections were explored, although we opted for simply present the most representative ones.

Nevertheless the distribution, different algorithm and features selection output different matchings percentages. Table 4.2 depicts such reality.

Table 4.2: Comparative solutions matching between simulation and reality.

Empirical		*C4.5*			*Random Forrest*	
tapcode	*iatacode*	*tapcode delay1*	*iatacode delay2*	*tapcode delay1 delay2*	*tapcode delay1*	*tapcode delay1 delay2*
19% (16.5%)	22% (19.5%)	54%	46%	57%	73%	86%

Table 4.2 is split into three parts. The first contains the percentage of matching solutions when using the manually classified TAP and IATA delay codes on the empirical probabilistic actions unveiled on subsection 3.1.4. These percentages were referred above and the small numbers in parentheses respects to the uncertainty levels. The central part of table 4.2 is concerned with the *C4.5* classification using different features. As one may observe, although IATA codes have outperformed TAP codes in the empirical scenario, this time it is the opposite. This fact will be explained shortly so for now it matters to understand that with *C4.5* we we able to increase the matching solutions up to 57% using TAP codes and two delay estimates. It also matters to point out that we carried out test with three delay estimates but no significant results were obtained and therefore were suppressed. The third part of table 4.2 respects to the *Random Forrest* algorithm. Generating widely branched trees, this algorithms was able to raise our solution matching till 86%.

One aspects are important to stress at this point, the numbers shown above correspond simulation results and were obtained by taking the decision trees produced by the classifiers, generating Java code and embedding such code in the Specialist agents. Therefore, the results above might be regarded as validating the decision trees against the training set. While this seems a blurred approach to assess the classification process, our aim was to improve the decision making practices within the simulation, that is, trying to reproduce the decision carried out in the real airline as close as possible.

Adding to the above, if we were trying to build a Decision Support System to assist OCC personnel reason over the arisen anomalies, then the decision tree validation process had to be different. For instance, we had to use *cross-validation* as explained in subsection 2.2.1. Table 4.3 contains the decision tree *cross-validation* results.

Table 4.3: *Cross-validation* results for the decision trees used.

C4.5			*Random Forrest*	
tapcode delay1	*iatacode delay2*	*tapcode delay1 delay2*	*tapcode delay1*	*tapcode delay1 delay2*
25%	24%	27%	30%	32%

The *cross-validation* results are more even than our simulated results. *Random Forrest* maintains its leadership but for a lesser margin. If we were building a DSS, we were not able to assure more than 32% of correct classification.

To conclude, the reason for the TAP codes outperform IATA counterparts is their higher number. As stated, the anomaly classification problem presents some shortcomings such as the lower number of features and their similarity. Therefore the higher number of symbolic attributes (features) the better as they increase the probability of matching different a class.

4.3 Simulated Organizational Performance Discussion

After presenting the experiments carried out to determine organizational performance on subsection 4.1.3, it is time to display the aggregate results obtained after simulation execution.

As stated, to beginning our assessment we started to measure the overall time required for the current TAP organizational structure to handle the 389 flights that suffered one anomaly within the week considered in subsection 4.1.1. This preliminary control assessment was twofold. First we used manually classified TAP delay codes to trigger the different workflows (see table C.2). Then we also tested the IATA delay codes counterparts (table C.1).

It is very important to not confuse triggering concept classification with solution classification. As one may see in appendix C the former relates to who detected the anomaly, the latter is concerned with the resolution method. While manual classification of problems was improved by using decision trees (refer back to the previous section), we would not be able to apply the same approach to the concepts classification since TAP does not record such information on any database.

Back to our performance assessment results, and before displaying some illustrative charts, table 4.4 gathers the most relevant labels found on the future bar charts making it easier to interpret the results.

Table 4.4: Column identifiers for operational performance results.

Human Collaborators	
as	Aircraft Specialist
cms	Crew Members
cs	Crew Specialist
fd	Flight Dispatcher
gs	Ground Supervisor
hs	HCC Supervisor
mss	Maintenance Services
os	OCC Supervisor
pss	Passenger Services
ss	Station Supervisor
Activities	
radio	Communicate by radio
phone	Communicate by phone
input	Input data on system
read	Read data from system
rson.	Decide upon anomaly
appr.	Approve solution
move	Move between locations

Considering the current TAP organizational structure and the TAP delay code categorization, the 389 anomalies required **283.29** hours to be addressed. Using the IATA delay codes, this time slightly dropped to **282.47**.

Before proceeding it worths elucidating how the values in bold were calculated. When an anomaly is detected, that is, when our simulation reads a flight associated with a delay code, it

gets the concept that in real-life detected the anomaly and starts a workflow. This workflow is composed by a set of activities each one with a minimum and a maximum time to complete. As mentioned in section 3.3, *Brahms* randomly assigns a duration to an activity and those durations are store in a file. Later the analytical infrastructure reads the file and sums up all the activity durations.

It matters to point out that, unlike collaborator *stress*, a metric that we will see next, the summing of durations is exclusive between agents. For instance, let's consider a communication by phone, which always involves two agents. Considering that the phone call lasts for 3 minutes, the overall organizational structure timing is incremented in 3 minutes. On the other hand, each agents' *stress* metric is also incremented in 3 minutes, contributing 6 minutes for the total *stress*.

On this context, *stress* might be understood as collaborator *workload*. It is a metric directly obtained from the activities duration, but attending to the example presented on the previous paragraph, with a distinct impact on the organizational structure.

Charts 4.8 and 4.9 show the collaborator *stress* and the summing of durations per activity associated to the current TAP organizational structure. The only difference between them lies on the manual delay code classification used to produce both statistics. The first used the proprietary TAP delay codes, the second used the open and more generic IATA codes.

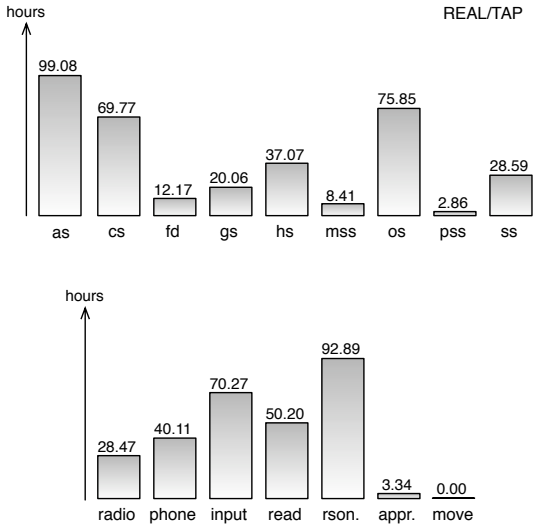

Figure 4.8: Collaborator *stress* and activity durations (Real/TAP). Caption on table 4.4.

As one may observe, if there was not many differences in the overall organizational performance, **283.29** of TAP against **282.47** of IATA, around 0.3% difference, the values between collaborator *stress* and activity duration are also roughly the same. But we should not see TAP and IATA codes as performance competitors, the competition will start soon, with our proposals of workflow changes.

Actually, and we underline this point, the charts above were solely included for two reasons. First, to understand that even though there are huge difference between delay code sets, the manual

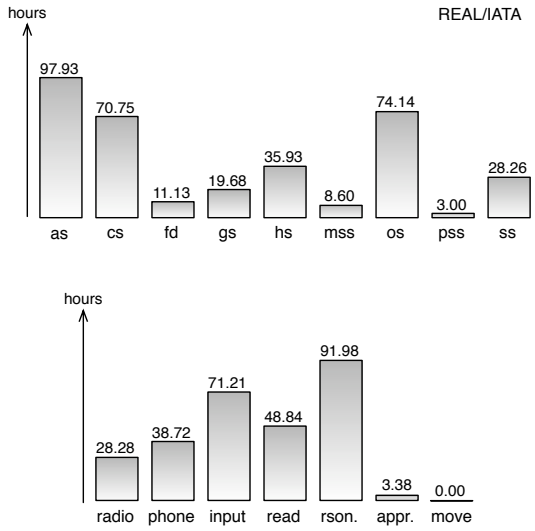

Figure 4.9: Collaborator *stress* and activity durations (Real/IATA). Caption on table 4.4.

concept classification undertaken by us and later validated by airline company personnel is faithful and trustful. While this seems pointless, it allows us to exclusively focus on a set of delay codes, be it TAP or IATA ones. We are now aware that, neglecting slight variations caused by the random nature of activity duration, drawing a chart for a set of codes is automatically valid for the other. The other reason is to present a control experiment that will allow us compare the future workflow transformations.

Since we are in presence of some data about each collaborator and aggregate activities, it is interesting to analyze the different distributions. For instance, the longer *stress* columns correspond to the Aircraft Specialist, OCC Supervisor and Crew Specialist. This really matches what happens in reality since these agents were characterized as the deciding personnel, and they are all located at the Operational Control Centre of TAP, a Decision Centre (see subsection 3.1.1). Other interesting observation is the difference between the *stress* of Aircraft and Crew Specialists. Throughout this text, we depicted the former as being more prevalent than the latter and looking into figure 4.5 we saw that Aircraft Specialist takes part in around 40% more decision processes that the Crew Specialist. Looking into the above charts, the *stress* levels between those actors differ around 30%.

Concerning cumulative activity duration times, it worths mention the absence of moving activities, since the agents communicate through radio or phone and the current collaborator placement suppresses the need to reach distant automated systems for form filling or information inputing. Another interesting point is the low value of "Solution Approval" time reflecting the low relevance of such activity, emphasized on subsection 3.1.3.

We are now ready to move into our first organizational structure transformation proposal. As referred on subsection 4.1.3, we suggested the removal of the Hub Control Centre Supervisor and after simulating the new workflows (see appendix A, section A.2) we obtained the chart 4.10.

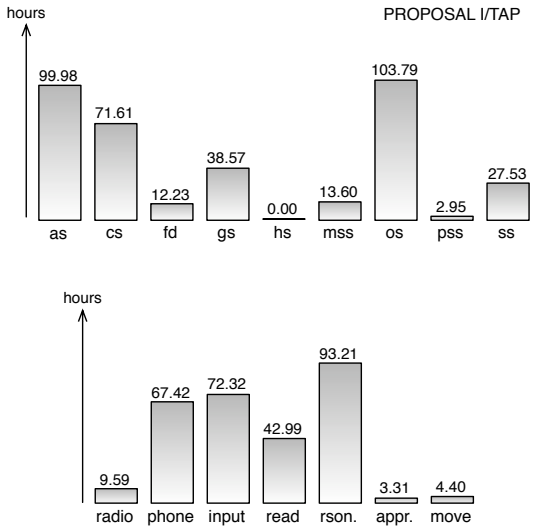

Figure 4.10: Collaborator *stress* and activity durations (Proposal I/TAP). Caption on table 4.4.

The two most noticeable difference between the real operational scenario and our proposal is the absence of *stress* in the HCC Supervisor ("hs") column and 4.40 hours of accumulated moving activities. The reasons for these are simple. First, the HCC Supervisor was removed and therefore does not produce work. Second, the Ground Supervisor and the Maintenance services had to start moving into the Hub Control Centre in order to report the anomalies, an activity later carried out by the "hs". A brief note goes to the time consumed by moving activities. We based our estimation on the table 3.4 with a vehicle usage of 50%.

Inline with the previous paragraph, the *stress* levels of the Ground Supervisor and Maintenance Services increase 90% and 61% respectively. OCC Supervisor also suffered a 36% increase on his *workload* because he now assumed other responsibilities, like communicating decisions to Ground Personnel and approving passenger anomalies solutions.

In what concerns cumulative activity durations, the most noticeable change was the huge decrease of communications by radio, 66% and the even higher increase on phone transmissions, 68%. Obviously, these changes would have a negative impact on cost but, as previously explained, our study did not covered such metric. There was also a slight increase in system inputting and moderate decrease on data perception (read), 14%.

Finally, the overall performance of our proposal scored **293.24** hours, far worst than the current TAP organizational structure. It is discussable that in terms of financial impact this solution would have scored lower, given the absence of the salaries of HCC Supervisors. While this might be truth, since we had no accurate data of such financial values we were not able to scrutinize those hypothesis.

By analyzing the increments in terms of moving and communication activities we realized that having a static computer terminal where data is inputted, besides technologically outdated was preventing us from optimizing the time efficiency of TAP organizational structure. This way,

in our proposal II we suggested to modify the Aircraft Movement System so it can be accessed through mobile devices, such as smartphone or laptops, within the airport range.

After implementing such changes at the simulation level, chart 4.11 was obtained.

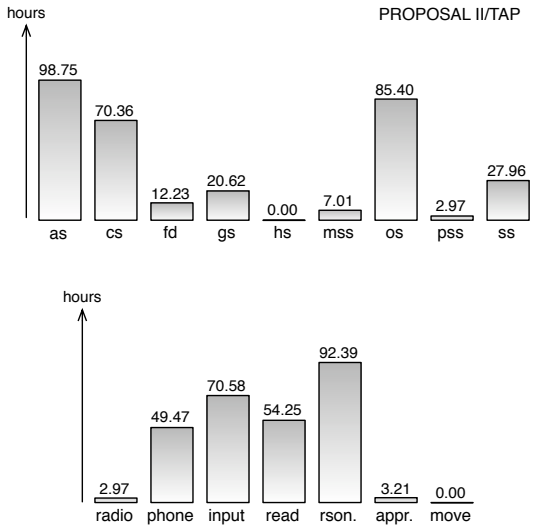

Figure 4.11: Collaborator *stress* and activity durations (Proposal II/TAP). Caption on table 4.4.

At first it is desirable to recall that our proposal II is a transformation after our proposal I, we did not go back to the current TAP organizational structure. Following this, the HCC Supervisor is absent from chart 4.11. Another important consideration should be made regarding the new "AMS" system. Since the system now features *intranet wireless* capabilities, it was considered that only the Ground Personnel, "gs" and "mss", have direct access to it.

Considerations made, when we compare proposal I with proposal II, the first noticeable difference is the reduction in the *stress* levels of Ground Supervisor, Maintenance Services and OCC Supervisor, that is, those who were highly affected when we removed the HCC Supervisor on our first proposal. The decrements in *workloads* represent, 46%, 48% and 18% respectively. It is also very interesting to notice that proposal II *stress* levels are very close to those observed in the current TAP organizational structure (see chart 4.8), the only exception is the OCC Supervisor, that suffered a raise from 75.85 hours to 85.40 hours, a 13% increase.

Comparing the accumulated activity timing across proposals I and II, it is easy to perceive a decrement in communication activities of around 47% on average. One may also notice the now absent time consumed on moving activities but an increase on 25% on data reading from systems, translating the now required attention to mobile devices. When comparing proposal II cumulative values with the current TAP performance, we denote a massive decrease in radio communications of around 90% and a lighter raise of 18% on the time spent on phone. Justifying the former is the current heavy usage of radio communications inside the airport, that we suggest to be replaced by direct access to the Aircraft Movement System.

Simulating our second proposal according to TAP delay code categorization, the 389 anomalies required **272.87** hours to be addressed, the best mark till now.

Given the good results obtained after a simple technological transformation and considering that other agents could benefit from the same accessibility, mainly the Station Supervisors spread across foreign bases that represent 56% of the total number of anomalies reported, in proposal III we converted the *intranet wireless* support to an *internet-based* system remotely accessible from everywhere.

Chart 4.12 was obtained after implementing an *internet-based* Aircraft Movement System at the simulation level.

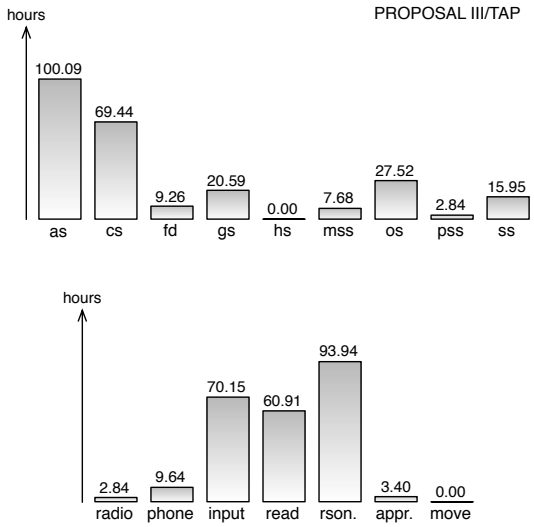

Figure 4.12: Collaborator *stress* and activity durations (Proposal III/TAP). Caption on table 4.4.

The *stress* chart in figure 4.12 is noticeably different from the previous. Comparing with Proposal II, the best overall performer till now, the OCC Supervisor stress dropped 67%, Flight Dispatcher and Station Supervisors also suffered reductions of 24% and 42% respectively.

When analyzing the cumulative duration per activity, the reason behind such decreases is clear. Communications by phone dropped an amazing 80% because now Flight Dispatcher and Station Supervisor were able to input disruption directly in the AMS, surpassing the need to transmit them by phone. On the other way around, the OCC Supervisor was no longer required to call to "fd" and "ss" with the purpose of passing resolution instructions.

On the last chart, the read activity slightly increased 11% representing the necessity of agents to pay attention to a screen in order become aware of decisions taken at the OCC. Anyway, according to table 3.3, the time consumed in reading activities is inferior to a phone call, so it is a beneficial tradeoff.

As one might imagine, our proposal III performed better than others, handling the 389 disruptions in just **240.88** hours.

To wrap all the above results, table 4.5 groups the overall performance scores as well as statistical aggregations regarding collaborator *stress*. Cumulative activity duration was left outside as it would hardly be taken into account for putative organizational structure selection. If, for instance, we had costs associated with each activity, such as phone or internet tariffs or fuel pricing, that would promote another metric, another deciding factor.

Table 4.5: Organizational structure performance assessment final results.

		Real		Proposal I	Proposal II	Proposal III
		TAP	IATA	TAP	TAP	TAP
Overall Score		**283.29**	**282.47**	**293.24**	**272.87**	**240.88**
Collaborator *Stress*	Avg	39	38	46	40	31
	Min	2.86	3.0	2.95	2.97	2.84
	Max	99.08	97.93	103.79	98.75	100.09

To conclude, one must keep in mind that it would be possible to assess a great number of operational scenarios and introduce a number of metrics well beyond those presented on this section. When looking into table 4.5, the airline company manager would be induced to introduce the changes on our proposal III, but we are the first to recognize that we are not in presence of the full scenario.

As we will see in the *Conclusion*, although our research study pioneered the application of simulation techniques to the airline business, other variables had to be taken into account when applying the measures proposed. And as any other simulation, our model may be gradually improved in the future, always providing more accurate results and accounting for a higher number of variable, but in the end it will always be a model of the reality.

Chapter 5

Conclusion

This chapter aims to wrap up all the research work carried out. In a section named *Goal Completion Assessment*, it starts by crossing the initial objectives with the answers provided. Within this section a comprehensive and systematic summary of the approaches followed as well as side activity will also be presented. The next section, *Future Work*, departs from the work already performed and suggests additions for forthcoming research journeys.

5.1 Goal Completion Assessment

The research project documented along this text was mainly motivated by the curiosity to understand the extent at which operations management is influenced by the underlying organizational structure.

Nowadays, with the increasing complexity of goods and services, organizational structures are present almost everywhere. All the human achievements that required more than one person to fulfill them, certainly required an organizational structure, even if nobody realized that.

The prevalence of progressively larger and complex organizational structures in a globalized world, where organizations face ferocious competition, tight deadlines and hostile economic environments, reveals the uttermost pertinence and applicability of the study at hands.

Trying to theoretically improve the putative efficiency of an operations management scenario based on a hypothetical organizational structure would have been daunting. Therefore, we brought the airline operational control centre case study. Despite not being the most familiar operational theatre, it is allegedly in the core of the success of an airline company within the competitive commercial aviation business.

The definition of a case study, while encouraging, was simply the beginning of our research endeavor. The next step was to specify which would be our approach to the problem. Instead of the traditional business process reengineering approach, focused solely in inputs and outputs, we wanted to go further, we aimed at an holistic representation of the problem, that would take into

account not only the assessment of the organizational structure outputs but also the evaluation of its organic parts.

Following this, we envisioned our first goal as a simulation of a real airline organizational structure. This simulation would comprise the most determinant entities in the success of airline operations management, not only human beings but also computerized systems and other relevant concepts. This simulation would mimic reality as closer as possible, would provide accurate data and would be the foundation for future organizational structures transformations.

Several scientific and technical repositories were searched, looking for previous research works around the same topic, at no avail. To the best of our knowledge we would be the first to simulate the airline operational control centre organizational structure in order to study its impact in airline operations management.

By the end of our research, we must acknowledge that this initial simulation effort was not only succeeded but also surpassed. Surpassed in the sense that along our research, other goals emerged and they were addressed as well.

The next paragraphs uncover our research work progress, some shortcomings faced and make the bridge with the achieved results at every stage.

Up to this moment, still on the early stages of our research, we only had a case study company and the holistic approach aspiration. The next steps were clear, select a tool to materialize our simulation and understand the reality to be modeled.

Our experience told us that multi-agent systems were the best suited to implement a distributed environment with multiple actions being carried out by multiple entities. From the wide range of multi-agent systems, we chose *Brahms* as modeling and simulation tool. Now that we already have a virtual airline organizational structure running, we must admit that *Brahms* was the riskiest choice of all.

At the beginning, in favor of *Brahms* we only had the NASA assurance that it would suit our holistic modeling desires. *Brahms* is being used in several NASA mission operations and the core developers of the tool are NASA researchers themselves. On the other hand, we were not aware of the challenges that had to be faced in the adoption of such tool. *Brahms* introduces an emergent human-centered paradigm based on the BDI software model and features its own agent-oriented programming language. If these two factors were not enough to require a lot more effort during the airline modeling, *Brahms* community is almost inexistent, which justifies the absence of articles, tutorials or code examples, besides the official documentation.

Neglecting the negative side of *Brahms* we accepted the challenge after the enthusiastic support offered by its creators. Attempting to promote their tool, they compromised themselves to help us through the recently introduced discussion group. At this point, we also intended to give back the knowledge they were willing to share. Therefore we decided to publish a set of tutorials in the discussion group, detailing step by step some undocumented procedures required to handle the tool.

While this was not a primary goal, it should be understood as an informal contribution to a scientific community and should be regarded as the fulfillment of a side objective. On appendix E is listed the result of such goal.

Back to the early days of our research, we had to get familiar with the operational reality within an airline company. Although we do not consider this stage as a goal, we must emphasize its value. For start, and as an highly restricted business, there is not much published information about this topic. While there are many software houses per country in the world, the same logic does not apply to airline companies. Therefore, the software development methodologies are widely know and scrutinized, the airline operations management are not. Also valuing the airline empirical observation stage is its hidden complexity. The empirical knowledge was conveyed trough interviews or direct inquires, which used informal textual or verbal language. An additional effort was required to shape and formalize such amount of information, as in its essence there were many flaws and imprecisions.

Still on the underlying observation stage of our simulation, we always advocated the use of real operational data as a mean to achieve a more faithful and realistic simulation. Given the willingness of the airline company to provide such data, on our side that would not be an obstacle but another challenge. If at the beginning we were unfamiliar with airline operations management, we were completely outsiders regarding airline operations database logging. With the help of self-made data mining scripts, we started to figure out the information beyond hundreds of unrecognized database column names.

The real *pre* and *post* operational data, the former containing the schedule airline activity and the latter capturing what actually took place, led us to delve into the airline operational control centre decision making processes. This must be underlined as the second major goal of our research work, with an importance on par with the impact of the organizational structure primary goal. Whilst we did not have a control group to test our organizational structures against with, the *post* operational data would allow to track the reasoning accuracy of our simulation, the closer to the reality the better. Starting from here, a new goal was established, improve the decision making practices within our simulation.

From our empirical observation, including some historical probabilistic reasoning tables provided by the airline company, we built the first simulation of an airline organizational structure. Then we fed the simulation with real 1801 scheduled flights that took place between 15th and 21st February 2010. We also inputted 389 anomalies suffered by those flights, such as aircraft defects, crew shortages, late cargo loading, and so on. The goal was threefold: measure the time taken by the overall organizational structure to handle such flight disruptions, evaluate the *stress* suffered by each airline operator and compare the number of simulated disruption solution against real life counterparts. The first two assessments would be grouped as *organizational performance*, the latter was categorized as *reasoning accuracy*.

The first results were obtained. The current airline organizational structure took 289.29 hours handle the 389 disrupted flights, and the 9 airline operators suffered, on average, 39 hours of *stress* each one. Concerning *reasoning accuracy*, implementing the empirical probabilistic actions

kindly provided by the airline company did not allow our simulation to match more than 21% of the real solutions, with 18% of associated uncertainty.

Based on such scenario, our next goals were straightforward, improve those values. Recalling our initial holistic simulation aspirations, in order to increase the current *organizational performance* we made three incremental transformations at the simulation level, later named *proposals*. In the first proposal we removed one airline operation foreseen as redundant; our second proposal gave *intranet wireless* capabilities to an existing static computerized terminal; the third proposal provided *internet* features to the last system.

Proposal *I* took 293.24 hours to handle the 389 disrupted flights and each operator suffered, on average, 46 hours of *stress* time. We were proved wrong concerning the redundancy of the subtracted operator since the overall *organizational performance* has degraded. Despite this fact and because, empirically speaking, the role of such operator was questionable, the *intranet wireless* capabilities given to the existing system aimed at remove certain moving activities introduced after the operator subtraction.

Proposal II scored 272.87 hours to handle all the disruptions with an average *stress* of 40 hours per operator. This represents a 4% overall optimization of the organizational structure but with a penalty of 3% in operator *stress*. Although the marginally better performance, airline collaborator would not definitely enjoy such structural transformations. Accounting for human well-being, our last proposal keep its bet on technological expansion. The once *intranet wireless* capability of the computerized system was replaced by a new *internet-based* system accessible from everywhere and all the operators that would benefit from it were instructed about how to use the system.

Our third proposed took 240.88 hours to manage the 389 disrupted flights and each operator suffered 31 hours of *stress*, on average. This means a up to 15% improvement in the overall disruption handling time and a up to 21% decrease in operator *stress*. Clearly the best organizational structure from the three proposed.

It worths emphasize that the quantitative organizational structure comparison exposed on the last paragraphs is the direct and unequivocal answer to the major goal of our research study, assessing the impact of different organizational structure on operations management. Nevertheless, in the middle of our research, the improvement of the *reasoning accuracy* within our organizational structure was also set as goal, therefore some approaches and results remain unexplained.

To increase the 21% reasoning accuracy obtained with the empirical methods, and possessing *pre* and *post* operational data, we relied on Artificial Intelligent supervised learning techniques to classify the solutions carried out on an airline operational scenario. Our classes match the solutions, such as changing aircraft or crew member, or delay the flight; while our features were the anomalies codes and some delays motivated by the anomaly.

From the existing supervised learning classifiers we solely used decision tree based classifiers. Other methodologies could have been used such as Naive Bayes or Neural Networks, in spite of that we opted for decision tree given their simplicity and understandability. Actually, using the delay as a feature that will participate in the reasoning process, might resemble the a real airline scenario where the estimated delay is accounted for when taking action to face a disruption.

At this stage we used the WEKA data mining tool to speed up our tests and mainly explored distinct feature sets across the *C4.5* and *Random Forrest* algorithms. After obtaining the decision trees from the input data, a script converted such trees into Java code that was later embedded into our simulation.

Using the *C4.5* algorithm we were able to match up to 57% of the real solutions, but with *Random Forrest* the solutions correspondence raised to 86%. Besides answering our interim goal, this results might sustain further research on the airline *decision making* research field.

To conclude, and after proving the successful completion of the established goals, one last objective started to make sense. Given the unfamiliarity of airline operations managements to the majority of people, attested on the absence of literature, and since we have a running simulation of the real and proposed airline organizational structures it would be interesting to develop and educational visualization of the simulated theatre. With this ludic challenge in mind, we explored the *Brahms JAPI* and the most recent *browser* technologies to implement a cartoonish but self-explanatory airline operational animation.

5.2 Future Work

This section aims at suggest some improvements or expansions to the current research work.

As any other modeling work, the simulation of a live reality is never complete. There is always something to be added to a simulation in order to better mimic its underlying reality. Bearing this in mind, our suggestions are all geared towards the improvement of the developed simulations, mainly in terms of completeness.

The ultimate goal should be to incrementally achieve higher levels of realism, therefore our first step would definitely be to validate and tune up the empirical knowledge conveyed through interviews. Unfortunately, accomplish such proposal would be very hard as it requires full collaboration of the airline company and its operators, and some procedures would be regarded as obtrusive.

For instance, the duration of our activity are based on estimates provided by the airline personnel. Whilst this is way better than random values, the ideal would be to record on video the daily activity of the operators and later calculate the time, not only by activity but also by operator, anomaly, weather conditions, time of the day, among so many other factors.

A more doable suggestion, still aiming at validating the current model, would be to take our simulation and visualization next to the involved airline entities, ask their opinion and how could we improve the existing model.

Other important expansion, mainly associated with the *reasoning accuracy* simulation and improvement but still requiring airline collaboration would be to introduce some changes on how the operational data is logged into databases. Throughout our *pre* and *post* operational data analysis, we were forced to remove orphan entries, neglect incomplete flights or ignore multiple anomalies due to a deficient and inaccurate logging system. Actually, the businesses in general should

be more conscious to this regard as massive data analytics is becoming increasingly important nowadays.

In what concerns the exploration new organizational structure scenarios, the possibilities are endless. Starting with the existing structure, one may add or remove collaborators and see how their *stress* levels react, other systems could be added, new activities that complement the existing activities. The perturbation of the actual system might be an option, such as taking into account operators distraction during work, or put some strain, e.g., high number of flight disruptions, in the organizational structure, among many others.

Also interesting would be to introduce other assessment metrics. In our research we were solely able to decently track time. The lack of information regarding salaries or even work plans prevented us to include the important economic factor. But this would have been critical in a real business process reengineering study. Accounting for the economic aspect would require not only the cost per hour, day, week or month for each operator but also the collaborator shifts and work plans. As one may recall, the Operational Control Centre of an airline company works 24 hours a day, 7 day a week, and therefore it must exist a working plan split across several employees.

Still on the metrics side, and having some payrolls, we would be able to track the amortization time of certain organizational structure transformations. For instance, in our proposal *II* and *III* we gave extra functionalities to an existing system. This had some costs involved that were not accounted for. Using the transformations costs and the increased productivity costs, we would be able to tell how long an investment would take pay off.

Last but not least, regarding the *reasoning accuracy* improvement, one could use other supervised learning techniques to try to increase the already high 86% match obtained through the *Random Forrest* classifier. An even more interesting approach would be to leave the easily comprehensible decision trees to the human agents and introduce decision support systems at the simulation based on more complex reasoning schemes, such as the ones obtained through Neural Networks or Naive Bayes.

References

[Alp04] Ethem Alpaydin. *Introduction to Machine Learning and Adaptive Computation*. MIT Press, 2004.

[And06] T. Andersson. Solving the flight perturbation problem with meta heuristics. *Journal of Heuristics*, 12:37–53, 2006.

[Arg97] M. Arguello. *Framework for Exact Solutions and Heuristics for Approximate Solutions to Airlines' Irregular Operations Control Aircraft Routing Problem*. PhD thesis, The University of Texas at Austin, United States, 1997.

[BA97] A. Breslow and W. D. Aha. Simplifying decision trees: a survey. *Knowledge Engineering Review*, 12:1–40, 1997.

[BBNO07] M. Ball, C. Barnhart, G. Nemhauser, and A. Odoni. Air transportation: Irregular operations and control. In C. Barnhart and G. Laporte, editors, *Handbook in OR & MS*. Elsevier, Amsterdam, 2007.

[BD04] A. Barto and T. Dietterich. Reinforcement learning and its relationship to supervised learning. In J. Si, A. Barto, and W. Powell, editors, *Handbook of Learning and Approximate Dynamic Programming*. Wiley Interscience/IEEE Press, 2004.

[BFOS84] L. Breiman, H. Friedman, A. Olshen, and J. Stone. *Classification and Regression Trees*. Wadsforth International Group, 1984.

[BM03] G. Batista and C. Monard. An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17:519–533, 2003.

[Bra07] M. Bramer. *Principles of Data Mining*. Springer-Verlag, London, 2007.

[BSW93] J. Blomberg and P. Sweton-Wall. Ethnographic field methods and their relation to design. In D. Schuler and A. Namioka, editors, *Participatory Design: Principles and Practices*. Lawrence Erlbaum Associates, Mahwah, N. J., 1993.

[Cab01] N. A. Cabrol. Science results of the nomad rover field experiment in the atacama desert, chile: Implications for planetary exploration. *Journal of Geophysical Research*, 106(E4):7664–7675, 2001.

[Cas08] A. Castro. Centros de controlo operacional: Organização e ferramentas. Monograph for Post-graduation in Air Transport Operations, 2008. ISEC - Instituto Superior de Educação e Ciências.

[Cla98] M. Clarke. Irregular airline operations: A review of the state-of-the-practice in airline operations control centre. *Journal of Air Transport Management*, 4:67–76, 1998.

[Cla02] W. J. Clancey. Simulating activities: Relating motives, deliberation, and attentive coordination. *Cognitive Systems Review*, 2002.

[CLLR10] J. Clausen, A. Larsen, J. Larsen, and N. Rezanova. Disruption management in the airline industry: Concepts, models and methods. *Computers & OR*, 37(5):809–821, 2010.

[CO10] Antonio J.M. Castro and Eugenio Oliveira. Disruption management in airline operations control – an intelligent agent-based approach. In Zeeshan ul-hassan Usmani PhD, editor, *Web Intelligence and Intelligent Agents*. INTECH, 2010. (avalilable from http://sciyo.com/articles/show/title/disruption-management-in-airline-operations-control-an-intelligent-agent-based-approach).

[CRR91] J. M. Corbett, L. B. Rasmussen, and F. Rauner. *Crossing the Border: The Social & Engineering Design in Computer Integrated Manufacturing Systems*. Springer-Verlag, London, 1991.

[CSS98] W. Clancey, P. Sachs, and M. Sierhuis. Brahms: Simulating practice for work systems design. *International Journal of Human-Computer Studies*, 49:831–865, 1998.

[DC96] D. Dutton and G. Conroy. A review of machine learning. *Knowledge Engineering Review*, 12:341–367, 1996.

[Ehn88] P. Ehn. *Work-Oriented Design of Computer Artifacts*. Arbetslivcentrum, Stockholm, 1988.

[ER99] T. Elomaa and J. Rousu. General and efficient multi-splitting of numerical attributes. *Machine Learning*, 36:201–244, 1999.

[GK91] J. Greenbaum and M. Kyng. *Design at Work: Cooperative Design of Computer Systems*. Lawrence Erlbaum Associates, Mahwah, N. J., 1991.

[Gro09] T. Grosche. *Computational Intelligence in Integrated Airline Scheduling*. Springer-Verlag, Berlin, Heidelberg, 2009.

[Guo05] Y. Guo. A decision support framework for the airline crew schedule disruption management with strategy mapping. In *Operations Research Proceedings*. Springer-Verlag, Berlin, Heidelberg, 2005.

[HA04] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22:85–126, 2004.

[HB98] K. Holtzblatt and H. Beyer. *Contextual Design:A Customer-Centered Approach to Systems Design*. Morgan Kaufmann, San Francisco, 1998.

[Hin62] J. Hintikka. *Knowledge and Belief*. Ithaca, Cornell University Press, 1962.

[HMS66] E. Hunt, J. Martin, and P. Stone. *Experiments in Induction*. Academic Press, New York, 1966.

[JLN+94] V. Johnson, L. Lettovsky, G.L. Nemhauser, R. Pandit, and S. Querido. Final report to northwest airlines on the crew recovery problem. Technical report, The Logistic Institute, Georgia Institute of Technology, Atlanta, USA, 1994.

[JMF99] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31:264–323, 1999.

[JMPW93] H. Johansson, P. McHugh, J. Pendlebury, and W. Wheeler. *Business Process Reengineering: Breakpoint Strategies for Market Dominance*. John Wiley and Sons, 1993.

[JS02] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study intelligent data analysis. Technical report, University of Ottawa, 2002.

[KK04] N. Kohl and S. Karisch. Airline crew rostering: Problem types, modeling, and optimization. *Annals of Operations Research*, 127:223–257, 2004.

[KLL+07] N. Kohl, A. Larsen, J. Larsen, A. Ross, and S. Tiourine. Airline disruption management — perspectives, experiences and outlook. *Journal of Air Transport Management*, 13:149–62, 2007.

[Kon82] K. Konolige. A first-order formalization of knowledge and action for a multi-agent planning system. In J. E. Hayes, D. Mitchie, and Y. Pao, editors, *Machine Intelligence*, page 41–72. Ellis Horwood, Chichester, UK, 1982.

[Kon94] I. Kononenko. Estimating attributes: analysis and extensions of relief. In L. De Raedt and F. Bergadano, editors, *Machine Learning: ECML-94*. Springer-Verlag, London, 1994.

[Kot07] S. Kotsiantis. Supervised machine learning: A review of classification techniques. *Informatica*, 31:249–268, 2007.

[LJC08] T. K. Liu, C. R. Jeng, and Y. H. Chang. Disruption management of an inequality-based multi-fleet airline schedule by a multi-objective genetic algorithm. *Transportation Planning and Technology*, 31:613–39, 2008.

[LJLT06] T. K. Liu, C. R. Jeng, Y. T. Liu, and J. Y. Tzeng. Applications of multi-objective evolutionary algorithm to airline disruption management. In *IEEE International Conference on Systems, Man and Cybernetics*. IEEE, New York, 2006.

[LM01] H. Liu and H. Motoda. *Instance Selection and Constructive Data Mining*. Kluwer, 2001.

[MA98] De Mantaras and E. Armengol. Machine learning from examples: Inductive and lazy methods. *Data & Knowledge Engineering*, 25:99–123, 1998.

[Mat96] D. Mathaisel. Decision support for airline system operations control and irregular operations. *Computers & Operations Research*, 23:635–726, 1996.

[MBCP98] R. Mayer, P. Benjamin, B. Caraway, and M. Painter. Framework and a suite of methods for business process reengineering. In V. Grover and W.J. Kettinger, editors, *Business Process Change: Reengineering Concepts, Methods and Technologies*. Idea Group Publishing, 1998.

[Mit06] Tom M. Mitchell. The discipline of machine learning. Technical report, Carnegie Mellon University, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, 2006.

[MR02] S. Markovitch and D. Rosenstein. Feature generation using general construction functions. *Machine Learning*, 49:59–98, 2002.

[Mur98] Murthy. Automatic construction of decision trees from data: A multi-disciplinary survey. *Data Mining and Knowledge Discovery*, 2:345–389, 1998.

[NH06] R. Nissen and K. Haase. Duty-period-based network model for crew rescheduling in european airlines. *Journal of Scheduling*, 9:255–78, 2006.

[Pav84] Calvin H. P. Pava. *Managing New Office Technology: An Organizational Strategy*. Free Press, New York, 1984.

[Qui79] J. R. Quinlan. Discovering rules by induction from large collections of examples. In D. Michie, editor, *Expert Systems in the Microelectronic age*. Edinburgh University Press, 1979.

[Qui93] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, 1993.

[Qui96] J. R. Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[Rei02] T. Reinartz. A unifying view on instance selection. *Data Mining and Knowledge Discovery*, 6:191–210, 2002.

[SC02] M. Sierhuis and W. Clancey. Modeling and simulating work practice: A method for work systems design. *IEEE Intelligent Systems*, 2002.

[SCS03] M. Sierhuis, W. Clancey, and C. Seah. Modeling and simulation for mission operations work system design. *Journal of Management Information Systems*, 19(4):85–128, 2003.

[Sho93] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60:51–92, 1993.

[Sie01] M. Sierhuis. *Modeling and Simulating Work Practice. Brahms: A Multiagent Modeling and Simulation Language for Work System Analysis and Design*. PhD thesis, Dept. of Social Science Informatics, Univ. of Amsterdam, Amsterdam, 2001.

[TG84] D. Teodorovic and S. Guberinic. Optimal dispatching strategy on an airline network after a schedule perturbation. *European Journal of Operational Research*, 1984.

[TS90] D. Teodorovic and G. Stojkovic. Model for operational daily airline scheduling. *Transportation Planning and Technology*, 14:273–85, 1990.

[TS95] D. Teodorovic and G. Stojkovic. Model to reduce airline schedule disturbances. *Journal of Transportation Engineering*, 121:324–31, 1995.

[TSW00] L. Tjen-Sien and L. WeiYin. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40:203–228, 2000.

[vHS00] R. van Hoof and M. Sierhuis. *Brahms Language Reference*. NASA/Ames Research Center, Moffett Field, CA, 2000. (avalilable at www.agentisolutions.com/documentation/language/ls_title.htm).

REFERENCES

[Vic99] K. J. Vicente. *Cognitive Work Analysis: Towards Safe, Productive, and Healthy Computer-Based Work*. Lawrence Erlbaum Associates, Mahwah, N. J., 1999.

[WF05] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2005.

[WYS97] G. Wei, G. Yu, and M. Song. Optimization model and algorithm for crew management during airline irregular operations. *Journal of Combinatorial Optimization*, 1:305–21, 1997.

[YAS+03] G. Yu, M. Arguello, G. Song, S. M. McCowan, and A. White. A new era for crew recovery at continental airlines. *Interfaces*, 33:5–22, 2003.

[YL04] L. Yu and H. Liu. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research*, 5:1205–1224, 2004.

[You89] E. Yourdon. *Modern Structured Analysis*. Prentice-Hall, Upper Saddle River, N. J., 1989.

[YQ04] G. Yu and X. Qi. *Disruption Management: Framework, Models and Applications*. World Scientific Publishing Company, 2004.

[YY96] S. Yan and D. H. Yang. A decision support framework for handling schedule perturbations. *Transportation Research Part B*, 30:405–19, 1996.

[ZZY02] S. Zhang, C. Zhang, and Q. Yang. Data preparation for data mining. *Applied Artificial Intelligence*, 17:375–381, 2002.

Appendix A

Operational Workflows

This appendix aims at gathering all the diagrams illustrating operational workflows in one place, allowing for better comparison and understanding.

According to Johansson et al. [JMPW93], the best way to describe and depict business workflows is by means of activity and sequence diagrams or a combination of both. After this, the present appendix is divided into two sections. The first includes the work interactions observable at TAP today. The second, groups the proposed workflows, that will be modeled and simulated in order to evaluate putative improvements.

A.1 Current TAP Operational Sequence Diagrams

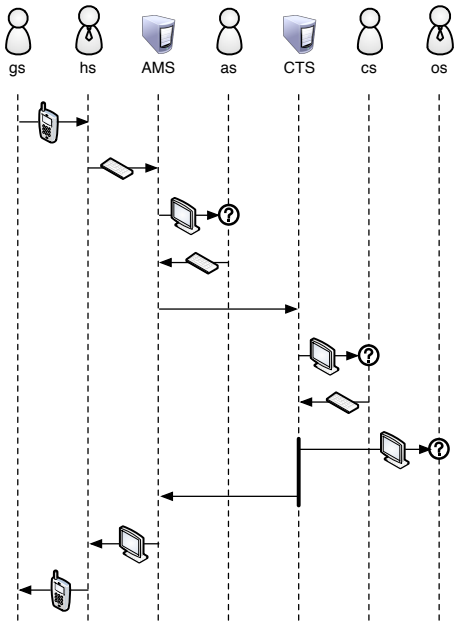

Figure A.1: Current operational workflow triggered by an aircraft anomaly detected by Ground Supervisor.

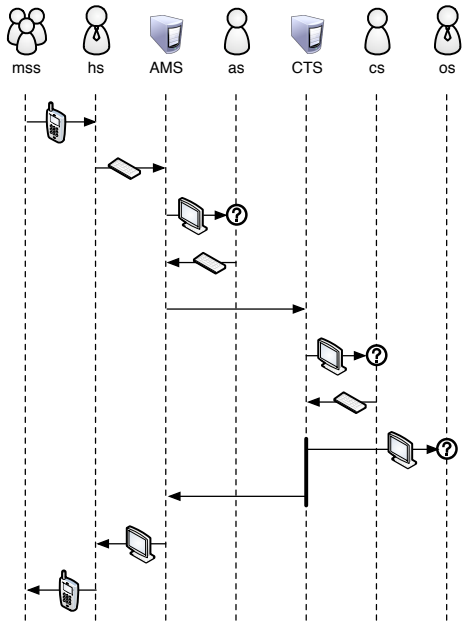

Figure A.2: Current operational workflow triggered by an aircraft anomaly detected by Maintenance Services.

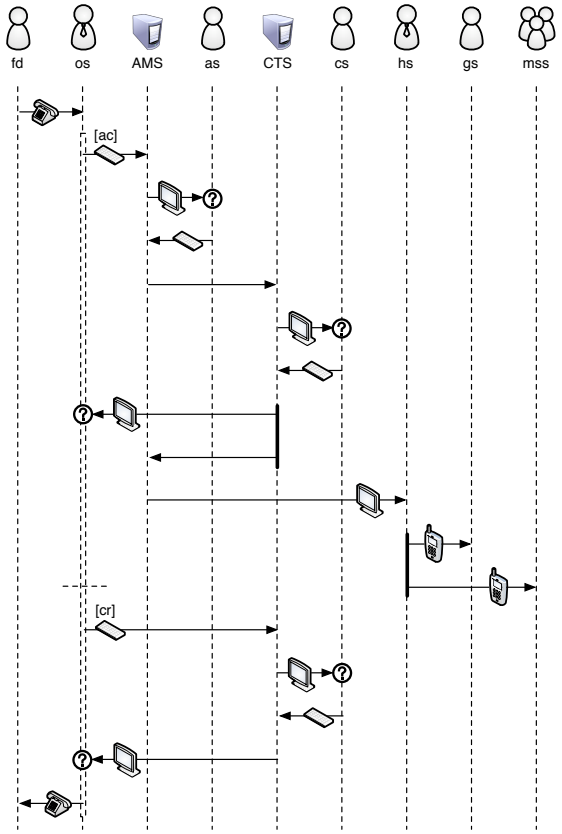

Figure A.3: Current operational workflow triggered by an aircraft or crew anomaly detected by Flight Dispatcher.

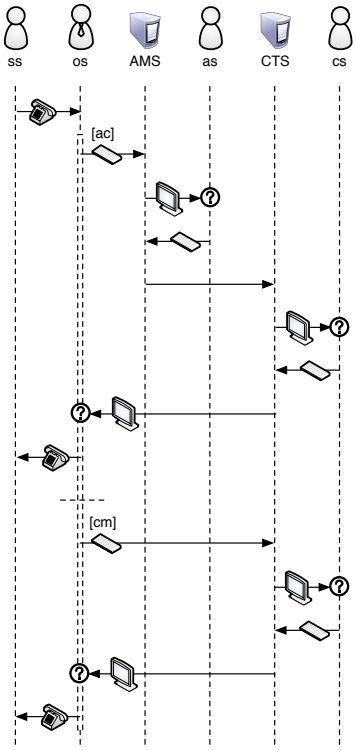

Figure A.4: Current operational workflow triggered by an aircraft or crew anomaly detected by Station Supervisor.

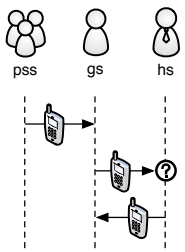

Figure A.5: Current operational workflow triggered by a passenger anomaly detected by Passenger Services.

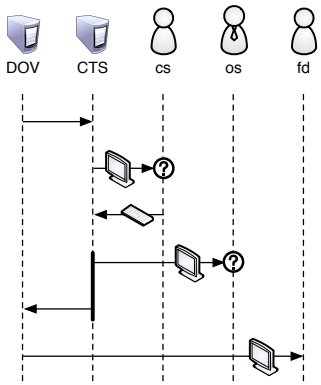

Figure A.6: Current operational workflow triggered by a crew anomaly detected by the Flight Operations Portal.

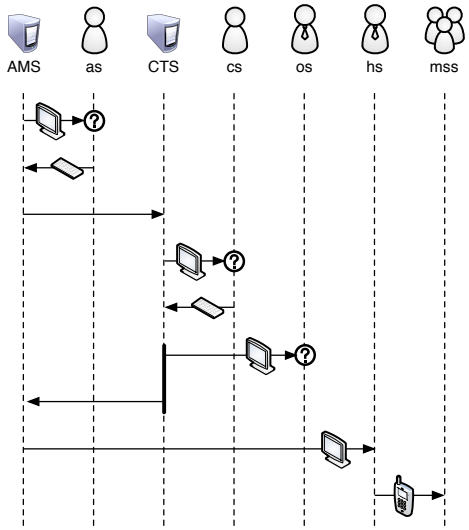

Figure A.7: Current operational workflow triggered by an aircraft anomaly detected by Aircraft Movement System.

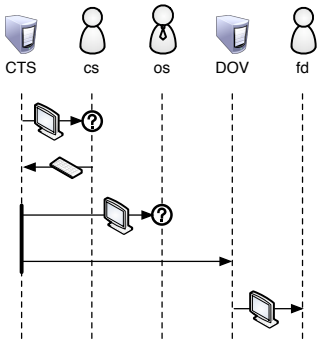

Figure A.8: Current operational workflow triggered by a crew anomaly detected by the Crew Tracking System.

A.2 Proposed Operational Sequence Diagrams

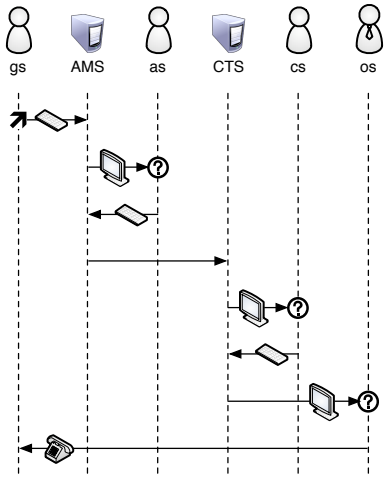

Figure A.9: Proposed *(I)* operational workflow triggered by an aircraft anomaly detected by Ground Supervisor.

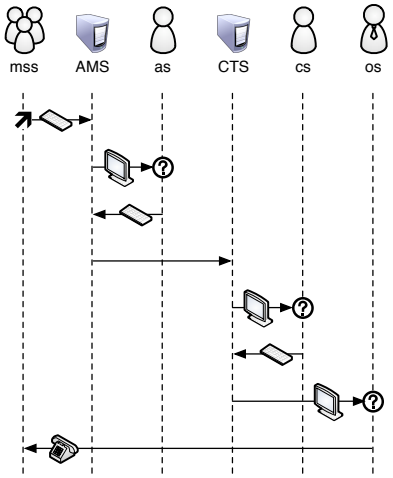

Figure A.10: Proposed *(I)* operational workflow triggered by an aircraft anomaly detected by Maintenance Services.

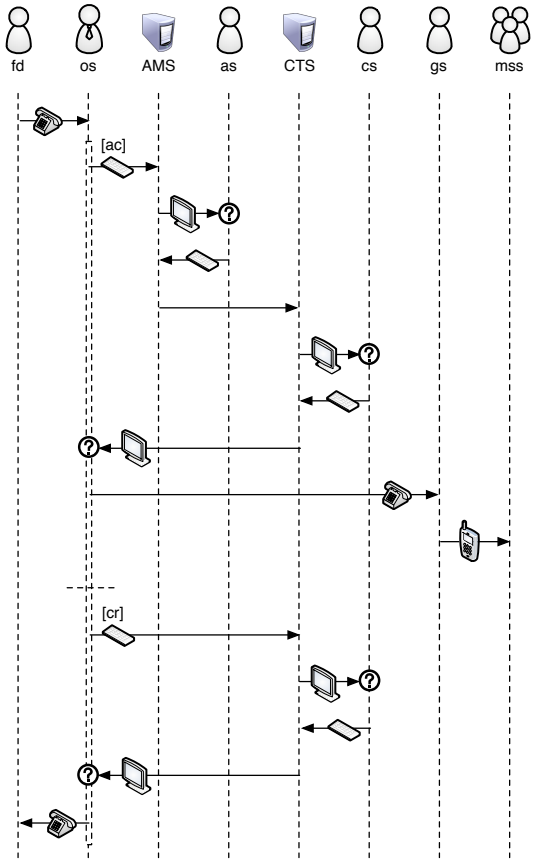

Figure A.11: Proposed *(I)* operational workflow triggered by an aircraft or crew anomaly detected by Flight Dispatcher.

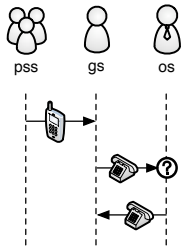

Figure A.12: Proposed *(I)* operational workflow triggered by a passenger anomaly detected by Passenger Services.

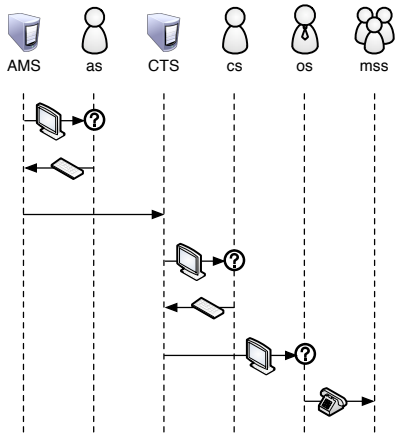

Figure A.13: Proposed *(I)* operational workflow triggered by an aircraft anomaly detected by Aircraft Movement System.

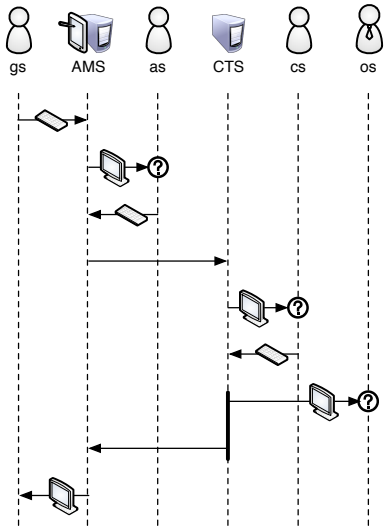

Figure A.14: Proposed *(II)* operational workflow triggered by an aircraft anomaly detected by Ground Supervisor.

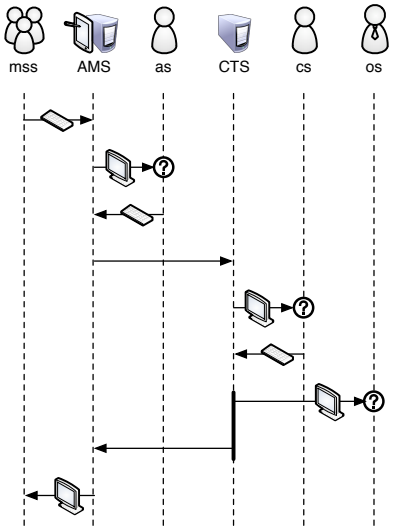

Figure A.15: Proposed *(II)* operational workflow triggered by an aircraft anomaly detected by Maintenance Services.

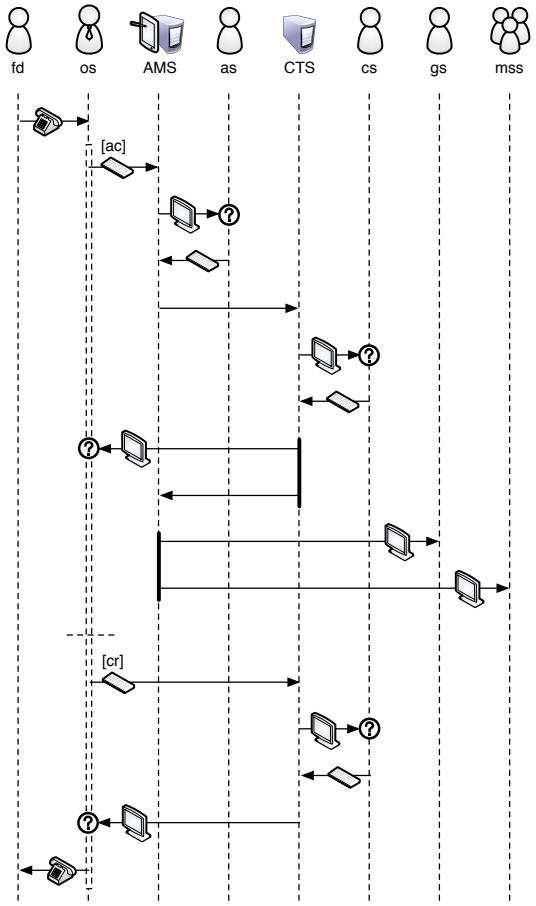

Figure A.16: Proposed *(II)* operational workflow triggered by an aircraft or crew anomaly detected by Flight Dispatcher.

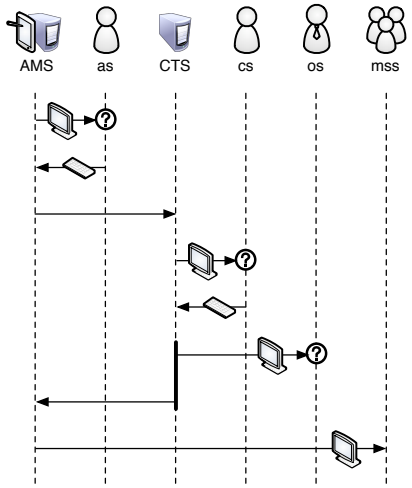

Figure A.17: Proposed *(II)* operational workflow triggered by an aircraft anomaly detected by Aircraft Movement System.

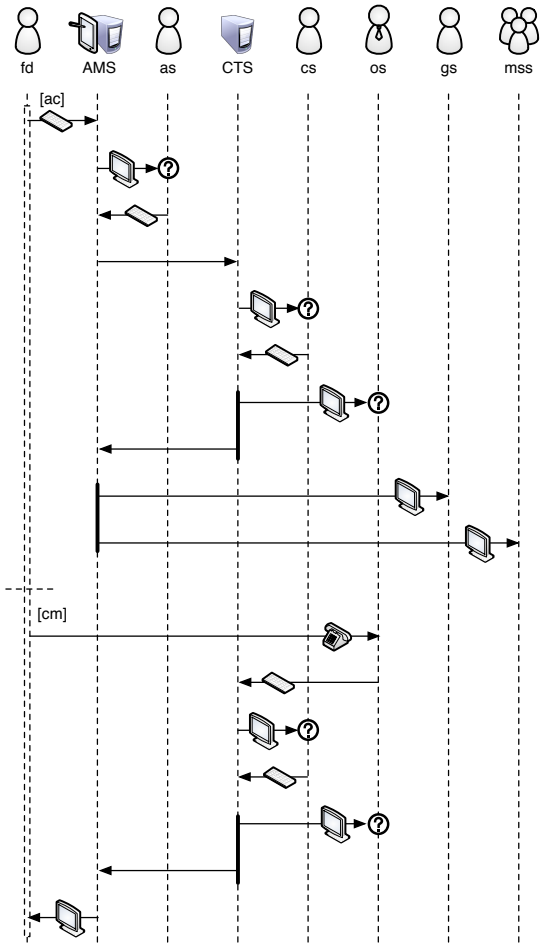

Figure A.18: Proposed *(III)* operational workflow triggered by an aircraft or crew anomaly detected by Flight Dispatcher.

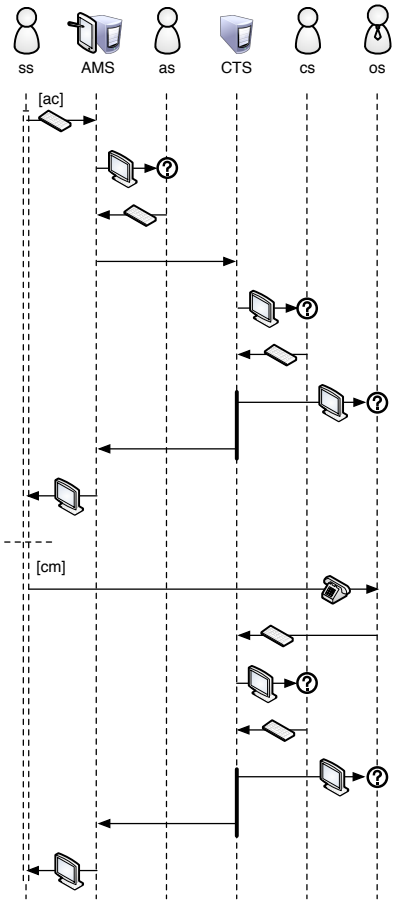

Figure A.19: Proposed *(III)* operational workflow triggered by an aircraft or crew anomaly detected by Station Supervisor.

Appendix B

Delay Codes

Whenever a flight suffers an anomaly on the ground, airline companies use a set of proprietary delay codes to signal what happened wrong. As an attempt to help airlines standardize the reasons behind commercial flight late departures, IATA, the *International Air Transport Association*, created a set of delay codes.

This appendix is intended to gather all the delay codes, both IATA and TAP, for easier reference. Given their general purpose, IATA codes are grouped into related operational fields and offer a more detailed textual description.

B.1 IATA Numeric Delay Codes and Description

Table B.1: IATA numeric delay codes and corresponding descriptions.

code	label	description
Others		
6	NO GATE/STAND AVAILABLE	Due to own airline activity
9	SCHEDULED GROUND TIME	Planned turnaround time less than declared minimum
Passenger and Baggage		
11	LATE CHECK-IN	Check-in reopened for late passengers
12	LATE CHECK-IN	Check-in not completed by flight closure time
13	CHECK-IN ERROR	Error with passenger or baggage details
14	OVER-SALES	Booking errors not resolved at check-in
15	BOARDING	Discrepancies and paging, missing checked in passengers
16	COMMERCIAL PUBLICITY OR PASSENGER CONVENIENCE	Local decision to delay for VIP or press, delay due to offload of passengers following family bereavement
17	CATERING ORDER	Late or incorrect order given to supplier
18	BAGGAGE PROCESSING	Late or incorrectly sorted baggage
Cargo and Mail		
21	DOCUMENTATION	Late or incorrect documentation for booked cargo
22	LATE POSITIONING	Late delivery of booked cargo to airport/aircraft
23	LATE ACCEPTANCE	Acceptance of cargo after deadline
24	INADEQUATE PACKING	Repackaging and/or re-labelling of booked cargo
25	OVER-SALES	Booked load in excess of saleable load capacity (weight or volume), resulting in reloading or off-load

27	DOCUMENTATION/PACKING	(Mail Only) Incomplete and/or inaccurate documentation
28	LATE POSITIONING	(Mail Only) Late delivery of mail to airport/aircraft
29	LATE ACCEPTANCE	(Mail Only) Acceptance of mail after deadline

Aircraft and Ramp Handling		
31	LATE/INACCURATE AIRCRAFT DOCUMENTATION	Late or inaccurate mass and balance documentation, general declaration, passenger manifest
32	LOADING/UNLOADING	Bulky items, special load, lack loading staff
33	LOADING EQUIPMENT	Lack of and/or breakdown, lack of operating staff
34	SERVICING EQUIPMENT	Lack of and/or breakdown, lack of operating staff
35	AIRCRAFT CLEANING	Late completion of aircraft cleaning
36	FUELLING/DEFUELLING	Late delivery of fuel, excludes late request
37	CATERING	Late and/or incomplete delivery, late loading
38	ULD	Lack of and/or unserviceable ULDs or pallets
39	TECHNICAL EQUIPMENT	Lack and/or breakdown, lack of operating staff, includes GPU, air start, push-back tug, de-icing

Technical and Aircraft Equipment		
41	TECHNICAL DEFECTS	Aircraft defects including items covered by MEL
42	SCHEDULED MAINTENANCE	Late release from maintenance
43	NON-SCHEDULED MAINTENANCE	Special checks and/or additional works beyond normal maintenance schedule
44	SPARES AND MAINTENANCE	Lack of spares, lack of and/or breakdown of specialist equipment required for defect rectification
45	AOG SPARES	Awaiting AOG spare(s) to be carried to another station
46	AIRCRAFT CHANGE	For technical reasons, e.g. a prolonged technical delay
47	STANDBY AIRCRAFT	Standby aircraft unavailable for technical reasons

Damage to Aircraft		
51	DAMAGE DURING FLIGHT OPERATIONS	Bird or lightning strike, turbulence, heavy or overweight landing, collisions during taxiing
52	DAMAGE DURING GROUND OPERATIONS	Collisions (other than taxiing), loading/offloading damage, towing, contamination, extreme weather conditions

EDP/Automated Equipment Failure		
55	DEPARTURE CONTROL	Failure of automated systems, including check-in, load control systems producing mass and balance
56	CARGO PREPARATION DOCUMENTATION	Failure of documentation and/or load control systems covering cargo
57	FLIGHT PLANS	Failure of automated flight plan systems

Flight Operations and Crewing		
61	FLIGHT PLAN	Late completion of or change to flight plan
62	OPERATIONAL REQUIREMENT	Late alteration to fuel or payload
63	LATE CREW BOARDING OR DEPARTURE PROCEDURES	Late flight deck, or entire crew, other than standby; late completion of flight deck crew checks
64	FLIGHT DECK CREW SHORTAGE	Sickness, awaiting standby, flight time limitations, valid visa, health documents, etc.
65	FLIGHT DECK CREW SPECIAL REQUEST	Requests not within operational requirements
66	LATE CABIN CREW BOARDING OR DEPARTURE PROCEDURES	Late cabin crew other than standby, late completion of cabin crew checks
67	CABIN CREW SHORTAGE	Sickness, awaiting standby, flight time limitations, valid visa, health documents
68	CABIN CREW ERROR OR SPECIAL REQUEST	Requests not within operational requirements

69	CAPTAIN REQUEST FOR SECURITY CHECK	Extraordinary requests outside mandatory requirements

Weather		
71	DEPARTURE STATION	Below operating limits
72	DESTINATION STATION	Below operating limits
73	EN-ROUTE OR ALTERNATE	Below operating limits
75	DE-ICING OF AIRCRAFT	Removal of ice and/or snow; excludes equipment
76	REMOVAL OF SNOW, ICE OR SAND FROM AIRPORT	Runway, taxiway conditions
77	GROUND HANDLING IMPAIRED BY ADVERSE CONDITIONS	High winds, heavy rain, blizzards, monsoons etc.

Air Traffic Flow Management Restrictions		
81	ATFM DUE TO ATC EN-ROUTE DEMAND/CAPACITY	Standard demand/capacity problems
82	ATFM DUE TO ATC STAFF OR EQUIPMENT EN ROUTE	Reduced capacity caused by industrial action or staff shortage, equipment failure, military exercise or extraordinary demand due to capacity reduction in neighboring area
83	ATFM DUE TO RESTRICTION AT DESTINATION AIRPORT	Airport and/or runway closed due to obstruction, industrial action, staff shortage, political unrest, noise abatement, night curfew, special flights
84	ATFM DUE TO WEATHER AT DESTINATION	Airport and/or runway closed due to weather conditions

Airport and Government Authorities		
85	MANDATORY SECURITY	Passengers, baggage, crew, etc.
86	IMMIGRATION, CUSTOMS, HEALTH	Passengers, crew
87	AIRPORT FACILITIES	Parking stands, ramp congestion, lighting, buildings, gate limitations etc.
88	RESTRICTIONS AT DESTINATION AIRPORT	Airport and/or runway closed due to obstruction industrial action, staff shortage, political unrest, noise abatement, night curfew, special flights
89	RESTRICTIONS AT AIRPORT OF DEPARTURE	Including air traffic services, start-up and push-back, airport and/or runway closed due to obstruction or weather (restriction due to weather in case of ATFM only) industrial action, staff shortage, political unrest, noise abatement, night curfew, special flights

Reactionary		
91	LOAD CONNECTION	Awaiting load from another flight
92	THROUGH CHECK-IN ERROR	Passenger or baggage check-in error at originating station
93	AIRCRAFT ROTATION	Late arrival of aircraft from another flight or previous sector
94	CABIN CREW ROTATION	Awaiting cabin crew from another flight
95	CREW ROTATION	Awaiting flight deck, or entire crew, from another flight
96	OPERATIONS CONTROL	Re-routing, diversion, consolidation, aircraft change for reasons other than technical

Miscellaneous		
97	INDUSTRIAL ACTION WITHIN OWN AIRLINE	Industrial action (includes Air Traffic Control Services)
98	INDUSTRIAL ACTION OUTSIDE OWN AIRLINE	Industrial action (except Air Traffic Control Services)
99	MISCELLANEOUS	No suitable code; explain reason(s) in plain text

B.2 TAP Delay Codes and Labels

Table B.2: TAP delay codes and related labels.

code	label
829	LACK OF OR LATE FUEL TRUCK
831	AIRCRAFT DEFECTS AT HOME BASE
832	AIRCRAFT DEFECTS AT OUTSTATIONS
833	LACK OF STAFF
835	X-RAY BAGGAGE SCANNING
836	SUSPEND MISSING PAX BAG SEARCH
837	OPERATIONAL SECURITY INSPECTION
838	ACC OCC-PAX IRREG
841	SLOW BOARD ON PREVIOUS FLT
843	LOADING/UNLOADING DELAYED ON PREV FLT
848	AIRCFT DEF ON PREV FLT
849	DCS DELAYED ON PREV FLT
850	LACK OF FLT CREW ON PREV FLT
851	LATE CREW BOARD ON PREV FLT
852	WEATHER COND ON PREV FLT
853	AIR TRAFFIC SERVICES ON PREV FLT
854	SECURITY DELAY ON PREV FLT
855	AIRPORT FACILITIES ON PREV FLT
856	LOAD CONNECTION PREV FLT
857	FLT/BLOCK TIME OF PREV FLT
858	ROTATION OTHERS
860	PASSENGER
861	WEATHER AT STATION OF DEPARTURE
865	LATE BAGGAGE ACCEPTANCE
868	SEARCHING/OFF LOADING MISSING PAX BAG
870	COMMERCIAL REASONS, PUBLICITY, PAX'S CON
871	AIRPORT SLOT
877	BAGGAGE
884	ULD LACK OF/OR SERVICE ABILITY
885	PGA OPS CONTROL
887	HCC
888	LATE CHECK-IN/ACEPTANCE AFTER DEADLINE
893	LATE CHECK-IN/CONGESTION CHECK AREA
897	CHECK-IN ERROR/PASSENGER AND BAGGAGE
904	SLOW BOARDING,DISCREPANCIES AND PAGING
905	SLOW BOARDING/GATE ERROR, LACK OF STAFF
908	EXCESSIVE HAND LUGGAGE
909	ILLNESS/DEATH OF PAX
911	LATE OR WRONG DELIVERY FROM DEPARTURE HA
912	LATE OR WRONG DELIVERY FROM TRANSFER HAL
914	BAGGAGE PROCESSING, SORTING, ETC.
919	DOCUMENTATION, ERRORS, ETC.
923	LATE ACCEPTANCE
936	AIRCRAFT DOCUMENTATTION LATE/INACCURATE
940	CABIN CLEANING
941	LOADING/UNLOADING
942	LACK OF LOADING STAFF, ERROR
946	LOADING EQUIPMENT
947	SERVICING EQUIPMENT
948	LACK OF OR LATE PAX. STAIRS
949	LACK OF OR LATE PAX. BUS

951	AIRCRAFT CLEANNING
952	FUELLING/DEFUELLING
953	CATERING, LATE DELIVERY OR LOADING DISCR
954	LATE CATERING DELIVERY BY CATERING COMPA
955	DISCREPANCIES
956	AIRCRAFT CHANGE, MEAL PLAN
959	TECHNICAL EQUIPMENT
963	AIRCRAFT DEFECTS
964	SCHEDULED MAINTENANCE/LATE RELEASE
968	NON SCHEDULED MAINTENANCE
975	AIRCRAFT CHANGE FOR TECHNICAL REASONS
976	AIRCRAFT CHANGE DUE AIRCRAFT DEFECT
980	DAMAGE DURING GROUND OPERATION
984	DEPARTURE CONTROL SYSTEM
985	DCS ERROR
987	FLIGHT PLANS
988	OTHER SYSTEMS
990	DATA LINE INTERRUPTED
991	OPERATIONAL REQUIREMENTS
992	LATE CREW BOARD/DEPARTURE PROCEDURES
993	ENTIRE CREW LATE BOARDING
998	FLIGHT DECK CREW SHORTAGE
999	FLIGHT DECK CREW SPECIAL REQUEST
1000	CABIN CREW BOARDING
1001	CABIN CREW SHORTAGE
1002	CABIN CREW ERROR OR SPECIAL REQUEST
1003	WRONG HEAD CHECK
1004	RE-ORDERS
1006	WEATHER AT STATION OF DESTINATION
1008	ATFM DUE TO ATC EN-ROUTE DEMAND/CAPACITY
1009	DE-ICING OF AIRCRAFT
1011	GROUND HANDLING IMPAIRED
1013	ATFM DUE TO RESTRICTION AT DEST APT
1014	ATFM DUE TO WEATHER AT DESTINATION
1015	MANDATORY SEGURITY
1016	IMMIGRATION, CUSTOMS
1017	AIRPORT FACILITIES
1018	RESTRICTIONS AT AIRPORT OF DEPARTURE
1019	LOAD CONNECTION (PAX/CARGO/MAIL)
1021	AIRCRAFT ROTATION
1022	CABIN CREW ROTATION
1023	CREW ROTATION
1024	ENTIRE CREW TOO LATE DUE TO ROTATION
1025	FLIGHT CREW TOO LATE DUE TO ROTATION
1026	OPERATIONS CONTROL
1028	INDUSTRIAL ACTION OUTSIDE OWN AIRLINE
1029	NOT ELSEWHERE SPECIFIED
1031	DPG (Planning and Management)
1032	SCHEDULED GROUND TIME LESS THAN DECLARED
1035	LATE AIRCRAFT DOCUMENT BRIEFCASE

Appendix C

Manual Delay Code Classification

This appendix contains the human classification of IATA and TAP delay codes according to workflow triggering concepts and TAP problem clustering. Background information about the classification process may be found on subsection 3.1.4, *Airline Anomalies Description and Classification*.

C.1 IATA Delay Code Classification

Table C.1: Manual IATA delay code classification.

code	concept	flight/aircraft problem	crew problem
6	gs	AIRP	–
9	AMS	OTH	–
11	pss	COMM	–
12	pss	COMM	–
13	pss	COMM	–
14	pss	HAND	–
15	pss	COMM	–
16	pss	HAND	–
17	gs	HAND	–
18	pss	SEC	–
21	gs	HAND	–
22	gs	HAND	–
23	gs	HAND	–
24	gs	HAND	–
25	gs	HAND	–
27	gs	HAND	–
28	gs	HAND	–
29	gs	HAND	–
31	gs	HAND	–
32	gs	AIRP	–
33	gs	AIRP	–
34	gs	AIRP	–
35	gs	AIRP	–
36	gs	AIRP	–
37	gs	AIRP	–

code	concept	flight/aircraft problem	crew problem
38	gs	AIRP	–
39	gs	AIRP	–
41	mss	MAINT	–
42	mss	MAINT	–
43	mss	MAINT	–
44	mss	MAINT	–
45	mss	MAINT	–
46	mss	OTH	–
47	mss	MAINT	–
51	mss	MAINT	–
52	mss	MAINT	–
55	fd	ATC	–
56	gs	HAND	–
57	fd	ATC	–
61	AMS	ROT	–
62	fd	ATC	–
63	CTS	CREW	INDUTY
64	DOV	CREW	SIGN
65	fd	CREW	OTH
66	CTS	CREW	INDUTY
67	DOV	CREW	SIGN
68	fd	CREW	OTH
69	fd	CREW	OTH
71	fd	ATC	–
72	fd	ATC	–

code	concept	flight/aircraft problem	crew problem
73	fd	ATC	–
75	gs	AIRP	–
76	gs	AIRP	–
77	gs	METEO	–
81	fd	ATC	–
82	fd	ATC	–
83	fd	ATC	–
84	fd	METEO	–
85	pss	SEC	–
86	pss	SEC	–
87	gs	AIRP	–

code	concept	flight/aircraft problem	crew problem
88	fd	ATC	–
89	fd	ATC	–
91	AMS	ROT	–
92	pss	SEC	–
93	AMS	ROT	–
94	CTS	CREW	ROT
95	CTS	CREW	ROT
96	fd	ATC	–
97	gs	OTH	–
98	gs	OTH	–
99	gs	OTH	–

C.2 TAP Delay Code Classification

Table C.2: Manual TAP delay code classification.

code	concept	flight/aircraft problem	crew problem
829	gs	AIRP	–
831	mss	MAINT	–
832	mss	MAINT	–
833	gs	AIRP	–
835	pss	SEC	–
836	pss	SEC	–
837	pss	SEC	–
838	pss	COMM	–
841	pss	HAND	–
843	gs	HAND	–
848	AMS	MAINT	–
849	AMS	MAINT	–
850	DOV	CREW	ROT
851	DOV	CREW	ROT
852	fd	METEO	–
853	fd	ATC	–
854	pss	SEC	–
855	gs	AIRP	–
856	gs	HAND	–
857	fd	ROT	–
858	AMS	ROT	–
860	pss	COMM	–
861	fd	METEO	–
865	pss	COMM	–
868	pss	HAND	–
870	pss	HAND	–
871	gs	AIRP	–
877	pss	HAND	–
884	gs	AIRP	–
885	gs	AIRP	–
887	gs	AIRP	–
888	pss	COMM	–
893	pss	COMM	–
897	pss	HAND	–

code	concept	flight/aircraft problem	crew problem
904	pss	COMM	–
905	pss	HAND	–
908	pss	SEC	–
909	pss	COMM	–
911	gs	HAND	–
912	gs	HAND	–
914	gs	HAND	–
919	gs	HAND	–
923	gs	HAND	–
936	gs	HAND	–
940	gs	AIRP	–
941	gs	AIRP	–
942	gs	AIRP	–
946	gs	AIRP	–
947	gs	AIRP	–
948	gs	AIRP	–
949	gs	AIRP	–
951	gs	AIRP	–
952	gs	AIRP	–
953	gs	AIRP	–
954	gs	AIRP	–
955	gs	OTH	–
956	gs	OTH	–
959	mss	MAINT	–
963	mss	MAINT	–
964	mss	MAINT	–
968	mss	MAINT	–
975	mss	MAINT	–
976	mss	MAINT	–
980	gs	MAINT	–
984	fd	ATC	–
985	fd	ATC	–
987	fd	ATC	–
988	fd	OTH	–

code	concept	flight/aircraft problem	crew problem
990	fd	ATC	–
991	gs	OTH	–
992	CTS	CREW	INDUTY
993	CTS	CREW	INDUTY
998	DOV	CREW	SIGN
999	fd	CREW	OTH
1000	CTS	CREW	OTH
1001	DOV	CREW	SIGN
1002	fd	CREW	OTH
1003	pss	COMM	–
1004	fd	ROT	–
1006	fd	ATC	–
1008	fd	ATC	–
1009	gs	AIRP	–
1011	gs	METEO	–
1013	fd	ATC	–

code	concept	flight/aircraft problem	crew problem
1014	fd	METEO	–
1015	pss	SEC	–
1016	pss	SEC	–
1017	gs	AIRP	–
1018	fd	ATC	–
1019	AMS	ROT	–
1021	AMS	ROT	–
1022	CTS	CREW	ROT
1023	CTS	CREW	ROT
1024	CTS	CREW	ROT
1025	CTS	CREW	ROT
1026	fd	ATC	–
1028	gs	OTH	–
1029	gs	OTH	–
1031	DOV	CREW	RULES
1032	AMS	OTH	–
1035	AMS	OTH	–

Appendix D

Operational Database Fields

Following subsection 3.2.2 this appendix contains all the operational database column names of the files provided by TAP which, in turn, are described on subsection 3.1.5.

As mentioned on the text body, the *op_airc_roster* file holding the aircraft roster had the same information as the *op_flight* file. Consequently there is no reason for it to appear here, as it was not considered on further analysis.

According to table 3.13, the *id*, *label* and *redundant* database fields of the tables bellow were filled by automatic analysis, while *description*, *rem.* (removable) and *regex* required human decision.

A final word goes for the (not depicted) *values* field, containing a list with the relative frequency of the instances associated with each column. Attending to the huge values list produced, we opted for keep the tables comprehensive and not depict it.

D.1 *Op_flight* File Label Analysis

Table D.1: Extended *op_flight* file labels and details.

id	label	description	rem.	regex	redundant
0	OP_FLIGHTSID	Database ID	yes		
1	PHASE	Record phase	no	`/^[0-9]{2}$...`	3
2	PERIOD_TIME	Date related to record	yes		
3	UPDATE_TIME	DB record time	yes		1
4	FLT_DATE	Date of the flight	no	`/^([0-9]{2}...`	
5	FLT_NBR	Flight number	no	`/^[0-9]{2,4...`	
6	CARR_CD	Carrier code	yes		
7	FROM_AIRP_CD	Departure airport code	no	`/^[A-Z]{3}$...`	
8	OP_SUFFIX	—	yes		14
9	FLEG_STAT	Flight record state	no	`/^(S|X|[1-9...`	
10	TO_AIRP_CD	Arrival airport code	no	`/^[A-Z]{3}$...`	
11	ACTYP_CD	Aircraft type code	no	`/^([A-Z]|[0...`	
12	AIRC_OWNR_CD	Aircraft owner code	yes		21;22
13	OPER_FLT_NBR	Operational flight number	yes		
14	OPER_OP_SUFFIX	—	yes		8
15	SCHD_DEP_DATE	Scheduled departure time	no	`/^([0-9]{2}...`	
16	SCHD_ARR_DATE	Scheduled arrival time	no	`/^([0-9]{2}...`	
17	SRVCE_TYP	—	yes		

18	FRST_SEATS	First class seats bought	yes		
19	BUS_SEATS	Business seats bought	no	/^[0-9]+$/	
20	ECON_SEATS	Economic seats bought	no	/^[0-9]+$/	
21	CABN_CARR_CD	—	yes		12;22
22	CKPT_CARR_CD	—	yes		12;22
23	LCL_FLT_DATE	—	yes		
24	AIRC_REG	Aircraft registry/plate	no	/^(([A-Z]|[...	
25	TRFFC_RSTR_LST	—	yes		
26	CLS_SALE_LST	—	yes		
27	INFLT_SRVCE_LST	—	yes		
28	LEASE_FLG	—	yes		
29	DEP_TERM_CD	Departure terminal code	yes		
30	ARR_TERM_CD	Arrival terminal code	yes		
31	FICTITIOUS_EXT	—	yes		
32	OLD_DEP_AIRP	Old departure airport	yes		
33	OLD_ARR_AIRP	Old arrival airport	yes		
34	DIV_IND	—	yes		44;45;46;66
35	RESCHD_IND	Reschedule indicator	yes		
36	EST_OFFBLK_DATE	Estimated takeoff time	no	/^(([0-9]{2...	
37	EST_AIRB_DATE	Estimated airborne time	yes		
38	EST_LNDNG_DATE	Estimated landing time	yes		
39	EST_ONBLK_DATE	Estimated arrival time	no	/^(([0-9]{2...	
40	ACTL_OFFBLK_DATE	Real takeoff time	no	/^(([0-9]{2...	
41	ACTL_AIRB_DATE	Real airborne time	yes		
42	ACTL_LNDNG_DATE	Real landing time	yes		
43	ACTL_ONBLK_DATE	Real arrival time	no	/^(([0-9]{2...	
44	TRVLING_TECH	—	yes		34;45;46;66
45	SPARES_PACK	—	yes		34;44;46;66
46	TECH_STOP	—	yes		34;44;45;66
47	NEXT_INFO_DATE	—	yes		
48	RET_RAMP_DATE	—	yes		
49	RET_LNDNG_DATE	—	yes		
50	RET_ONBLK_DATE	—	yes		
51	CURR_LEASE_FLG	—	yes		
52	CURR_ACTYP_CD	—	yes		
53	CURR_OWNR_CD	—	yes		
54	REM	—	yes		
55	ADMIN_CARR_CD	—	yes		
56	FRST_MEALS	First class meals	yes		
57	BUS_MEALS	Business class meals	yes		
58	ECON_MEALS	Economic class meals	yes		
59	FRST_SALE_SEATS	First class seats available	yes		
60	BUS_SALE_SEATS	Business seats available	no	/^[0-9]+$/	
61	ECON_SALE_SEATS	Economic seats available	no	/^[0-9]+$/	
62	CYCLS	—	yes		
63	MANL_SALE_CONFIG	—	yes		
64	ADMIN_FLT_DATE	—	yes		
65	SCHD_FLT_TYP	Schedule flight type	yes		
66	HIGHLIGHT	—	yes		34;44;45;46
67	MANL_SCHD_FLT_TYP	—	yes		
68	ENGNRS	—	yes		
69	LCL_ADMIN_FLT_DATE	—	yes		
70	DLY_DEP_MIN	Departure delay (minutes)	no	/^-?[0-9]+$...	
71	DLY_ARR_MIN	Arrival delay (minutes)	no	/^-?[0-9]+$...	

D.2 *Op_flight_dep_dly* File Label Analysis

Table D.2: Extended *op_flight_dep_dly* file labels and details.

id	label	description	rem.	regex	redundant
0	OP_FLIGHTS_DEP_DLYID	Database ID	yes		
1	PHASE	Record phase	yes		
2	PERIOD_TIME	Date related to record	yes		
3	UPDATE_TIME	DB record time	yes		
4	FLT_DATE	Date of the flight	no	`/^([0-9]{2}...`	
5	FLT_NBR	Flight number	no	`/^[0-9]{2,4...`	
6	CARR_CD	Carrier code	yes		
7	FROM_AIRP_CD	Departure airport code	no	`/^[A-Z]{3}$...`	
8	OP_SUFFIX	—	yes		
9	STAT	Flight record state	no	`/^(S|X|[1-9...`	
10	DEP_DLY_TYP	TAP delay code	no	`/^[0-9]{3,4...`	13;17;24
11	DATE_TIME	Time of delay registration	no	`/^[0-9]{2}-...`	
12	MINS	Delay minutes	no	`/^[0-9]+$/`	
13	SYSTEM_PKEY	System primary key	yes		10;17;24
14	DLY_CD	IATA delay code	no	`/^[0-9]{2}$...`	18;20;21;22
15	DLY_SUB_CD	Delay sub code	yes		19;23
16	RSPBL_CD	Responsible code	no	`/^[A-Z]{2}$...`	26;27
17	DESCR	Delay description	no	`/^.+$/`	10;13;24
18	CHNG_REASN_CD	Change reason code	yes		14;20;21;22
19	CHNG_REASN_SUB_CD	Change reason sub code	yes		15;23
20	CHNG_REASN_CD_1	Change reason code 1	yes		14;18;21;22
21	DESCR_1	Delay description 1	yes		14;18;20;22
22	CHNG_REASN_CD_2	Change reason code 2	yes		14;18;20;21
23	CHNG_REASN_SUB_CD_1	Change reason sub code 1	yes		15;19
24	DESCR_2	Delay description 2	yes		10;13;17
25	IATA_CHNG_REASN_CD	IATA change reason code	yes		
26	RSPBL_CD_1	Responsible code 1	yes		16;27
27	DESCR_3	Delay resp. description	no	`/^.+$/`	16;26
28	RSRC_NAME	Resource name	no	`/^([A-Z])+$...`	
29	RSRC_AREA	Resource category	no	`/^([A-Z])+$...`	
30	RSRC_TEL_NBR	Resource telephone number	yes		
31	REM	Delay additional comments	yes		

D.3 *Op_crew_roster* File Label Analysis

Table D.3: Extended *op_crew_roster* file labels and details.

id	label	description	rem.	regex	redundant
0	OP_CREW_ROSTERID	Database ID	yes		
1	PHASE	Record phase	no	`/^[0-9]{2}$...`	3
2	PERIOD_TIME	Date related to record	yes		
3	UPDATE_TIME	DB record time	yes		1
4	CRW_GRP_CD	Crew group code	no	`/^1|2$/`	
5	ACTV_TYP	Activity type	no	`/^[0-9]{1,2...`	
6	EMPL_NBR	Employee number	no	`/^[0-9]{5}..`	
7	STRT_DATE	Activity start date	no	`/^([0-9]{2}...`	
8	END_DATE	Activity end date	no	`/^([0-9]{2}...`	
9	ACTV_STAT	Activity attribution phase	no	`/^(S|T)$/`	
10	ASSGND_USR_ID	— usr	yes		
11	ASSGN_TYP	—	yes		

12	GRND_ACTV_FK	— foreign key	yes		
13	LEAV_TYP	Absence type	no	/^([A-Z]{4}...	
14	LEAV_CD	Absence code	no	/^([A-Z]{3,...	
15	PRNG_CD	Pairing code	no	/^.*$/	49
16	PRNG_FROM_DATE	Pairing from date	no	/^(([0-9]{2...	48
17	PRNG_LVL	Pairing level	yes		51
18	EMPL_TYP	Employee type	yes		52
19	RSRV_ACTV_FK	— foreign key	yes		
20	DUTY_MINS	Expected duty minutes	no	/^-?[0-9]+$...	
21	CRED_MINS	Credits in minutes	yes		
22	POSN_CD	Position in flight code	no	/^(([A-Z]|[...	
23	FLEET_CD	Fleet type	no	/^(([A-Z]|[...	
24	ACTL_SGNON_DATE	Real sign-on date	no	/^(([0-9]{2...	
25	ACTL_SGNOFF_DATE	Real sign-off date	no	/^(([0-9]{2...	
26	ACTL_CRED_MINS	Real credits in minutes	yes		
27	POOL_CD	Type of aircraft code	no	/^([A-Z]|[0...	
28	RANK_CD	Crew rank or position	no	/^[A-Z]{3}$...	
29	ASSGND_DATE	Time of pairing assignment	no	/^([0-9]{2}...	
30	EMPL_PREF_LEV	—	yes		
31	CRED_FLG	—	yes		
32	PUBLISH_DATE	Planning publishing date	yes		
33	PUBLISH_USR_ID	— usr	yes		
34	NOTIFY_DATE	Crew notify date	no	/^(([0-9]{2...	
35	NOTIFY_USR_ID	— usr	yes		
36	REM	Additional comments	yes		
37	FLYNG_MINS	Expected flying minutes	no	/^-?[0-9]+$...	
38	ACTL_DUTY_MINS	Real duty minutes	no	/^[0-9]+$/	
39	ACTL_FLYNG_MINS	Real flying minutes	no	/^[0-9]+$/	
40	BID_TYP	—	yes		
41	RNWL_ACTV_FK	— foreign key	yes		
42	CRSE_ACTV_KEY	— foreign key	yes		
43	SERIAL_NBR	—	yes		
44	CRED_FLG_REASN_CD	—	yes		
45	DSRPTD_NBR	—	yes		
46	RELEASED_USR_ID	— usr	yes		
47	DUTY_GRP_SEQ_NBR	—	yes		
48	PRNG_DATE	Pairing date	yes		16
49	PRNG_CD_1	Pairing code 1	yes		15
50	CRW_GRP_CD_1	Crew group code 1	yes		
51	PRNG_LVL_1	Pairing level 1	yes		17
52	EMPL_TYP_1	Employee type 1	yes		18
53	STRT_DATE_1	Start date 1	yes		
54	FLT_SET_NAME	Pairing preffix	yes		
55	END_DATE_1	End date 1	yes		
56	PRNG_TYP	Pairing type	yes		
57	FLT_DATE	Date of the flight	no	/^([0-9]{2}...	
58	FLT_NBR	Flight number	no	/^(-1)|([0-...	
59	OP_SUFFIX	—	yes		
60	CARR_CD	Carrier code	yes		
61	FROM_AIRP_CD	Departure airport code	no	/^([A-Z]{3}...	
62	TO_AIRP_CD	Arrival airport code	no	/^([A-Z]{3}...	
63	HOTEL_CD	Night sleep hotel code	yes		
64	PAX_CARR_CD	Crew as pax carrier code	yes		
65	PAX_FLT_NBR	Crew as pax flight number	yes		
66	PAX_OP_SUFFIX	—	yes		
67	PAX_COST	Crew as pax cost	yes		

68	ACTYP_CD	Aircraft type code	no	`/^(([A-Z]|[...`	
69	PNTS	—	yes		
70	FLEG_STAT	Flight record state	no	`/^((S|X|[1-...`	
71	FDT_END_IND	—	yes		
72	NOTIFY_DATE_1	Crew notify date 1	yes		
73	NOTIFY_USR_ID_1	— usr	yes		
74	DSRPTN_DATE	—	yes		
75	REINF_BY	—	yes		
76	RTE_IND	—	yes		
77	SRVC_TYP_IND	—	yes		

Appendix E

Brahms and JAPI Tutorials

Being an academic and research tool, *Brahms* penetration is weak when compared to other multi-agent systems or simulation engines. A short user base prevents articles, tutorials or examples about *Brahms* usage to be widely available on the internet, making it difficult for newcomers to understand or get familiar with the tool.

This appendix intends to aggregate a set of tutorials purportedly written along our research study as a contribution to the growing *Brahms* community. They are mostly concerned with *Brahms* installation, model compilation and execution and advanced JAPI features.

E.1 Installing Brahms on Mac OS X

The first step in using a software tool is to download, install and configure it. By the time of this writing, the Brahms download page provided two versions of the Brahms: an alpha, not extensively tested, and a stable (older). For this tutorial the later was chosen, version 1.3.2.

Brahms is only available for research purposes, so registration is required in order to download the installer. A varying time after registration an email should be received containing download credentials and a license file.

Requirements

- Mac OS X 10.5.8
- Brahms Agent Environment v1.3.2

Procedure

1. After registration, download the Brahms installer.
2. Unzip the downloaded file, *setup.zip*, by clicking on it.
3. Open the *Terminal* (console).
4. Following UNIX best practices and in order to allow simultaneous installations of other versions, the goal is to install Brahms in the `/usr/local/brahms-1.3.2` directory. Create it.

```
$> sudo mkdir /usr/local/brahms-1.3.2
```

5. Now, set the right ownership on the folder.

```
$> sudo chown `echo $USER`:staff /usr/local/brahms-1.3.2
```

6. Click on the *setup.app* file and proceed with installation. When prompted for "Choose Install Folder" click "Choose..." and then press Command-Shift-G. Enter `/usr/local/brahms-1.3.2/` in the box and complete installation.
7. For convenience, Brahms binaries should be added to the `PATH`. The environment variable `BRAHMS_HOME` should be also set. Open the *.bash_profile* file using your favorite text editor, for instance:

   ```
   $> vim ~/.bash_profile
   ```

8. Copy and paste the next instructions:

   ```
   export PATH="$PATH:/usr/local/brahms-1.3.2/AgentEnvironment/bin"
   export BRAHMS_HOME="/usr/local/brahms-1.3.2/AgentEnvironment"
   alias bc="/usr/local/brahms-1.3.2/AgentEnvironment/bin/bc"
   ```

   The *alias* instruction is required because Mac OS X already ships with a tool invoked using `bc` command, an thus colliding with the Brahms compiler (also `bc`).
9. Copy the license file, *brahms.lic*, to `/usr/local/brahms-1.3.2/AgentEnvironment`. Assuming it is on `/Desktop`:

   ```
   $> mv ~/Desktop/brahms.lic /usr/local/brahms-1.3.2/AgentEnvironment
   ```

10. Quit and reopen *Terminal* (to refresh environment variables).
11. Test Brahms installation by issuing the Brahms Virtual Machine.

    ```
    $> bvm
    ```

    The list of `bvm` options should appear but if something went wrong a *Java Exception* is thrown.

E.2 Installing Brahms on Ubuntu (server)

In Brahms download page the Linux version of Brahms Agent Environment already ships with Sun JRE 1.6 and was solely tested in Fedora Core 8. When it comes to Linux, Ubuntu is undoubtedly the most popular distribution and you may want to install Java JRE/JDK using a system-wide package manager, such as `aptitude` or `apt-get`.

Given the cross-platform nature of Java, and if you followed the previous tutorial, installing Brahms on a Ubuntu headless server, remotely administered through SSH, is fairly easy.

This tutorial arose from the need of having a remote, always available installation of Brahms controlled through a web browser.

Requirements

- Headless Ubuntu server
- Java JDK 5
- Brahms Mac OS X installation

Procedure

1. Brahms should have been installed on Mac OS X according to previous instructions.
2. Create a "tar gunziped" archive of the whole installation folder.

   ```
   $> sudo tar -zcvf brahms-1.3.2.tar.gz /usr/local/brahms-1.3.2
   ```

3. Upload the archive to Ubuntu server home folder.

```
$> scp brahms-1.3.2.tar.gz <user>@<host>:/home/<user>/
```

This tutorial tries to be as generic as possible, so you should replace `<user>` and `<host>` by your username and host/IP address of your server. Additional `scp` parameters might be needed, such as a *private key* path or a custom port. Alternatively, you may also use `ftp`.

4. Optionally, remove the created "tar gzip" archive as it's no longer needed.

```
$> sudo rm brahms-1.3.2.tar.gz
```

5. Establish a secure shell connection with your server.

```
$> ssh <user>@<host>
```

Again, replace `<user>` and `<host>` by proper values.

6. Determine if Java is installed on server and which version.

```
$> java -version
```

7. If last command returned a "command not found" or Java version is 4 or bellow, install Java JDK 5.

```
$> sudo apt-get install sun-java5-jdk
```

You may chose to install JDK 6 but since Mac OS X ships with JDK 5, this version was preferred.

8. Move the uploaded archive (2) from home folder to Ubuntu `/usr/local` directory.

```
$> sudo mv brahms-1.3.2 /usr/local
```

9. Extract the *brahms-1.3.2.tar.gz* archive, maintaining the folder tree structure.

```
$> sudo tar -zxvf /usr/local/brahms-1.3.2.tar.gz
```

10. Remove the now unneeded archive.

```
$> sudo rm /usr/local/brahms-1.3.2.tar.gz
```

11. Installation is complete but you should add Brahms binaries to the `PATH` and set `BRAHMS_HOME` environment variable. Open the *.bashrc* file using your favorite text editor, for instance:

```
$> vim ~/.bashrc
```

12. Copy and paste the next instructions:

```
export PATH="$PATH:/usr/local/brahms-1.3.2/AgentEnvironment/bin"
export BRAHMS_HOME="/usr/local/brahms-1.3.2/AgentEnvironment"
```

13. Reload *.bashrc* and test Brahms installation.

```
$> source ~/.bashrc && bvm
```

The list of `bvm` options should appear but if something went wrong a *Java Exception* is thrown.

E.3 Running the Example Simulation (GUI)

In the previous tutorials we see how to install and setup Brahms. In the first case, Mac OS X provides a graphical environment and Brahms ships with graphical tools, such as the Brahms Composer.

This tutorial will show how to compile and run the Brahms simulation provided in the official Brahms tutorial using the graphical Brahms Composer on Mac OS X.

Requirements

- Brahms installation (Mac OS X)

- Brahms environment variables set
- Brahms Tutorial Files

Procedure

1. Refer back to Brahms Tutorial 01 on how to install a Brahms on Mac OS X and set environment variables.
2. Download the Atm Tutorial files (on Brahms webpage).
3. Extract the archive to your Desktop. You will end with an *AtmModel* folder in `/Desktop`.
4. Open *Terminal*, change current working directory and check if old Brahms compiled XML files are present.

```
$> cd ~/Desktop/AtmModel
$> find . -name "*.xml"
```

5. If a list of XML files appears, remove them. (In the past, Brahms compiler output files had the XML extension. Currently it generates BCC files.)

```
$> find . -name "*.xml" -delete
```

6. Let's now open Brahms Composer, it is located in the *bin* folder under `$BRAHMS_HOME`.

```
$> open $BRAHMS_HOME/bin/Composer.app
```

7. A new graphical window will open asking to "Open Brahms Model", select "Import" and in the combo-box navigate to *Desktop*.
8. Open *AtmModel* > *final_source* > *gov* > *nasa* > *arc* > *brahms* > *atm*, and in the end select the *AtmModel.b* file.
9. Keep the option to create a build directory selected and click "Import".
10. All the simulation files will be loaded, and you can navigate over the model concepts using the left-side panel of Brahms Composer. Before running the simulation, the Brahms model must be built/compiled. To do so, click on the blue down-arrow in the topmost toolbar or choose "Build" > "Build Model" in main menu. The message "Successfull Build" should appear in the status bar.
11. Now you may execute the model, by clicking on the blue right arrow in the toolbar or choosing "Run" > "Run Model" in the menu.
12. The Brahms Composer right panel should switch to the "VM Log" viewer and some messages are displayed.
13. During the *AtmModel* execution, several simulation output files were produced, namely:

```
$BRAHMS_HOME/Databases/AtmModel_<date>_<time>.txt
$BRAHMS_HOME/logs/EventInformation_<date>_<time>.txt
$BRAHMS_HOME/logs/vm_<date>_<time>.log
```

Where `<date>` and `<time>` correspond to the date and time of simulation execution.

E.4 Running the Example Simulation (CLI)

In the previous tutorial the provided example simulation was compiled and executed using Brahms Composer (a GUI front-end). This time we use Brahms command line tools to perform the same tasks.

Requirements

- Brahms installation
- Brahms environment variables set
- Brahms Tutorial Files

Procedure

1. Refer back to Brahms Tutorial 01 or Brahms Tutorial 02 on how to install and set Brahms environment variables through command line.
2. Download the Atm Tutorial files (on Brahms webpage).
3. Extract the archive to your *Desktop*. You will end with an *AtmModel* folder in `/Desktop`.
4. Open *Terminal*, change current working directory and check if old Brahms compiled XML files are present.

```
$> cd ~/Desktop/AtmModel
$> find . -name "*.xml"
```

5. If a list of XML files appears, remove them. (In the past, Brahms compiler output files had the XML extension. Currently it generates BCC files.)

```
$> find . -name "*.xml" -delete
```

6. There are four Brahms projects in the *AtmModel* folder: *final_source*, *lesson_7_source*, *lesson_8_source*, *lesson_9_source*; corresponding to incremental development stages of the tutorial. First, get some info about `bc` (Brahms Compiler) tool, then compile the final source.

```
$> bc -?
$> bc -lp $BRAHMS_HOME/Models/lib -dtd $BRAHMS_HOME/DTD \
    -source final_source final_source/gov/nasa/arc/brahms/atm/AtmModel.b
```

   If all went well, you will see the message "Exporting model as XML" and some BCC files will appear in */Desktop/AtmModel/final_source/gov/nasa/arc/brahms/atm*.

7. Now it's time to run our simulation. First you need to add the *final_source* folder path to the *vm.cfg* file located in `$BRAHMS_HOME/config`. Open the file and change the line:

```
library_path=/usr/local/brahms-1.3.2/AgentEnvironment/Models/lib;
/Users/<homefolder>/Desktop/AtmModel/final_source
```

   Remember to change `<homefolder>` by the proper value. If you don't know it, the `pwd` command might come to rescue.

8. Now start the simulation using the `bvm` (Brahms Virtual Machine) tool.

```
$> bvm -cf $BRAHMS_HOME/config/vm.cfg gov.nasa.arc.brahms.atm.AtmModel
```

   You can add the `-ui` option if you want to graphically pause/play the simulation.

9. After running the model, you will find Brahms output simulation files in `logs` and `Databases` folders under `$BRAHMS_HOME`.

E.5 Introducing Brahms JAPI

Now that we have seen how to compile and run a Brahms model, it is time to move to more advanced topics.

This tutorial intends to show how to work with Brahms JAPI, *Java Application Programming Interface*. In the previous tutorials we used the *AtmModel* example, available at the Brahms website. This time, we will also use an example shipped with Brahms installation, located in the examples folder under `$BRAHMS_HOME`.

According to the shipped *ReadmeJAPI.txt*:

> The Java API (JAPI) contains all that is need to develop external activities, aka Java activities and external agents. The Java activities allows a model builder to perform tasks that are not supported by the Brahms language itself. The Java activities have to written using the Java language. External agents can be used to wrap external processes as an agent that needs to interact with other agent and objects present in the virtual machine.

Requirements

- Brahms installation
- Sun JDK 1.5 or above

Procedure

1. Copy and rename the example project folder, to the *Desktop*. This way it will allow you to make changes without messing the original code.

```
$> cp -r $BRAHMS_HOME/examples/japi ~/Desktop/myjapi
```

2. As you may see, the project is ideally structured by separating Brahms from Java code and source from built files. Let's do some cleanup first.

```
$> cd ~/Desktop/myjapi && \
     rm -r java/build/* && \
     rm java/lib/*
```

3. The Java source folder contains two packages, *jact* and *jagt*, you may delete the later, as we won't need it for this tutorial.

```
$> rm -r java/src/gov/nasa/arc/brahms/jagt
```

4. Now we are ready to compile Java code making sure to add Brahms libraries, namely *vm.jar* and *common.jar*, to the CLASSPATH.

```
$> javac -cp $BRAHMS_HOME/lib/vm.jar:$BRAHMS_HOME/lib/common.jar \
     -d java/build \
     java/src/gov/nasa/arc/brahms/jact/*.java \
     java/src/gov/nasa/arc/brahms/jact/string/*.java
```

5. Create a JAR, named *japitutorial.jar*, containing the built class files inside the *lib* folder.

```
$> jar cvf java/lib/japitutorial.jar -C java/build .
```

6. For Brahms models make use of the Java classes, the JAR archive has to be copied to `$BRAHMS_HOME/deploy` folder.

```
$> cp -f java/lib/japitutorial.jar $BRAHMS_HOME/deploy
```

7. Now it's time to turn into Brahms model, starting by compiling it.

```
$> bc -lp $BRAHMS_HOME/Models/lib -dtd $BRAHMS_HOME/DTD \
     -source brahms/JACExample/source -d brahms/JACExample/build \
     brahms/JACExample/source/example/JACExample.b
```

8. Append the Brahms *build* folder path to the *vm.cfg* file located in `$BRAHMS_HOME/config`. Open the file and change the line (if you are following along, it will look like):

   ```
   library_path=/usr/local/brahms-1.3.2/AgentEnvironment/Models/lib;
   /Users/<homefolder>/Desktop/myjapi/brahms/JACExample/build
   ```

   Where `<homefolder>` must be changed to the proper value. If you don't know it, the `pwd` command might help.

9. Now that everything is setup and in place, execute Brahms model using the `bvm` tool.

   ```
   $> bvm -ui -cf $BRAHMS_HOME/config/vm.cfg example.JACExample
   ```

   As you will see, there are messages printed from the Java code.

10. By the end of simulation, you may want to clean Java built files.

    ```
    $> rm -r java/build/* java/lib/* $BRAHMS_HOME/deploy/japitutorial.jar
    ```

11. And Brahms counterparts.

    ```
    $> rm -r brahms/JACExample/build/*
    ```

12. As stated in the previous tutorials, Brahms output simulation files might be found in `logs` and `Databases` folders under `$BRAHMS_HOME`, you may delete them as well.