

Optimizing Limousine Service with AI

Andy Hon Wai Chun

■ A common problem for companies with strong business growth is that it is hard to find enough experienced staff to support expansion needs. This problem is particularly pronounced for operations planners and controllers, who must be very highly knowledgeable and experienced with the business domain. This article is a case study of how one of the largest travel agencies in Hong Kong alleviated this problem by using AI to support decision making and problem solving so that its planners and controllers can work more effectively and efficiently to sustain business growth while maintaining consistent quality of service. AI is used in a mission-critical fleet management system (FMS) that supports the scheduling and management of a fleet of luxury limousines for business travelers. The AI problem was modeled as a constraint-satisfaction problem (CSP). The use of AI enabled the travel agency to sign up additional hotel partners, handle more orders, and expand its fleet with its existing team of planners and controllers. Using modern web 2.0 architecture and proven AI technology, the agency was able to achieve low-risk implementation and deployment success with concrete and measurable business benefits.

The fleet management system (FMS) was created for Airport Limousine Services Ltd. (Airport Limousine), a subsidiary of Swire Travel, one of the largest and oldest travel management companies in Hong Kong and widely recognized as the leader in service quality in the local travel industry. Swire Travel has more than 50 years of history and is part of the 145-year old worldwide Swire group.

The main business of Airport Limousine is a 24-hour luxury limousine and shuttle service that serves mainly the Hong Kong area (see figure 1), which has an area of 1104 square kilometers (426 square miles) with a population of 7 million in 2009. In comparison, the region serviced by Airport Limousine is similar to New York City—the five boroughs of New York occupy 786 square kilometers (303 square miles) with a population of 8.4 million as of 1 July 2009.¹

Airport Limousine's fleet consists of Mercedes-Benz sedans and stretch limousines as well as multipurpose vehicles (MPVs) and shuttle buses. It provides two main types of limousine services—airport transfer and on-hire services. The airport transfer service provides transfers to and from the Hong Kong Chek Lap Kok International Airport (figure 2) with one or more possible stopovers or pickup points in between. For airport transfers, Airport Limousine also provides a personalized meet-and-greet service and assists with VIP check-in procedures if needed. The limousine on-hire service, on the other hand, provides a vehicle and chauffeur to support any type of personal or business transportation needs, such as business appointments, plant site



*Figure 1. A View of the Business Districts in Hong Kong Island and Kowloon
— the Main Servicing areas for Airport Limousine's Fleet.*

Creative Commons photo by Base64, retouched by Carol Spears.



*Figure 2. Map Showing the Main Traffic Arteries Connecting the Hong Kong Chek Lap Kok
International Airport with Other Parts of Hong Kong SAR.*

Provided by OpenStreetMap under Creative Commons Attribution-ShareAlike 2.0 license.



Figure 3. Map Showing the Hong Kong International Airport and Its Relative Location to Main Destinations Served by Airport Limousine, such as Hong Kong Island, Kowloon, and the Mainland of China.

(Source: U.S. Central Intelligence Agency World Factbook)

visits, hospitality for overseas corporate clients, or personal events, such as weddings or banquets. Trips can also be "cross border" to the mainland of China (figure 3) using vehicles that have dual mainland China and Hong Kong Special Administrative Region (SAR) license plates. Since their

clientele consists mainly of business users and upscale market clients, quality of service and precise pickup and drop-off timing are very important. Specific details of each of these services are stored and displayed in the fleet management system.

Business Needs

Airport Limousine is young and fast growing with aggressive expansion plans. However, one of the major bottlenecks in expansion capability is the availability of skilled planners and controllers. Besides long and extensive training, the job can be very hectic and stressful at times. Planners and controllers need to take in information from many different channels, constantly communicate with limousine chauffeurs to record statuses and dispatch orders, and continuously perform scheduling and rescheduling decisions while trying to balance various business and operational criteria to maximize profit and productivity while maintaining service quality. A fair amount of problem solving is needed as traffic can be congested and lead to delays, clients might be late, flights might be delayed, and so on. Airport Limousine found it hard to train quickly an adequate number of knowledgeable planners and controllers who are up to these challenges to meet Airport Limousine's aggressive growth plans. Hence it decided to investigate the use of advanced technology, such as AI, to help solve some of these problems.

Using technology to streamline operations is not new to this limousine company. It has already been using many different advanced technologies to reduce planner and controller workload so that they can handle more orders. Currently the company has a web-based online reservation system that is used both by its business clients as well as hundreds of travel agents. Hotel orders, which are simpler, are recorded separately. A global positioning system (GPS) monitors the location of all vehicles on a map. A separate system retrieves airport flight statuses and arrival and departure times, as well as arrival hall information so that limousine chauffeurs know exactly where to meet up with their clients for airport pickup. Unfortunately, these systems were not integrated, and the planners and controllers had to operate several computers and view different monitors to get a complete picture.

System Objectives

The project goal for FMS is simple: use AI to help planners and controllers handle more orders and manage more vehicles while maintaining high service quality.

This high-level goal was then broken down into several finer and more concrete objectives. First, in order for any AI system to work, access to or integration with all necessary data and information sources was needed. The first objective of FMS was to provide a set of screens to consolidate all relevant data and information into one place. This included creating software to access order information from clients, travel agents, and hotels; flight

information, statuses, and arrival hall locations; limousine and chauffeur data; vehicle locations; and passenger statuses. Once all the data integration was done, this version of FMS was released to the planners and controllers as a decision-making support system for manual operations.

The second objective of the system is to help planners and controllers easily visualize the state of their world in one simple screen. Using data from different sources, we were able to create a Gantt chart to help planners and controllers visualize vehicle utilization and availability over time. This bird's-eye view allowed planners and controllers to make more informed and better decisions.

The third objective of the system is to be able to automate the vehicle dispatching and rescheduling function using AI while maintaining high service quality. This is done through constraints, heuristics, and an AI search algorithm. One of the original concerns is that AI will not have the same experience as human planners and controllers when it comes to estimating travel times. Accuracy in determining travel times is tightly linked to service quality, which is a key concern.

Application Description

FMS is a web-based application used by Airport Limousine's planners and controllers as well as its travel agents. Planners and controllers use the system to perform vehicle dispatching, while travel agents use FMS to view vehicle details and assignment statuses. Figure 4 is a high-level use case diagram for FMS.

The daily workflow is usually as follows. After midnight, the planner uploads daily orders to FMS for limousine services. This includes orders from clients, travel agents, and hotel concierges. Once uploaded, a spreadsheet (figure 5) displays details of orders so that they can be further edited if needed. In addition, FMS automatically connects airport transfer orders with real-time flight information and statuses. Orders with special needs, such as VIP, meet and assist, or stopovers, are highlighted with colored flags.

The planner may, at this time, also update availability for all limousines and vehicles. For example, some vehicles might be scheduled for maintenance and will not be available. Once all order and limousine data are updated, the planner then proceeds to assign vehicles to orders using AI. Orders are usually assigned vehicles in time periods of a few hours each. As orders are completed, chauffeurs report back on their statuses—when they reach the pickup point, when the client is on board, and when they have reached their destinations. All these statuses are updated into FMS by the controllers.

The FMS Gantt chart (figure 6) provides a bird's-

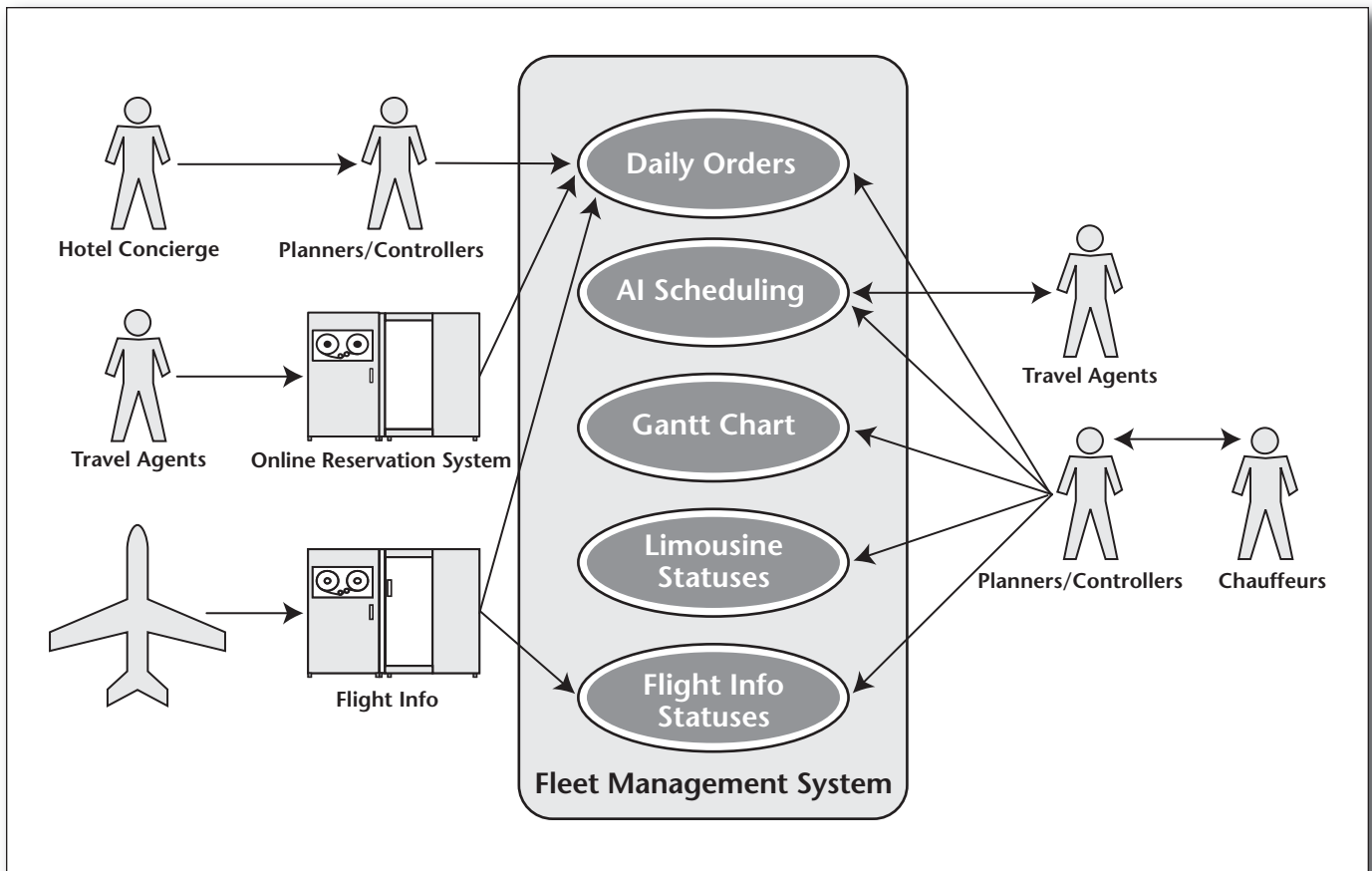


Figure 4. High-Level Use Case Diagram for FMS.

eye view of the entire day. The Gantt chart rows are organized according to vehicles and then the orders they are assigned, sorted by the pickup time. By doing so, it is very easy to see at a glance which vehicles are free and whether any vehicle has slack time. In addition, orders with special needs are also highlighted on the chart.

All orders are connected with live flight information. A separate flight information screen (figure 7) displays all flights that are associated with orders or simply all the flights of the day. Because the international airport is divided into different terminals and halls, FMS also displays hall and terminal assignment information to help chauffeurs figure out where to meet their passengers.

As new orders are received during the day, the AI engine can be invoked to provide proposals for vehicle assignments. Besides their own fleet of limousines and vehicles, Airport Limousine may also outsource some of the orders to third parties. Figure 8 shows a typical pop-up window with a list of prioritized AI proposals. The highest priority "AI Suggestion" list contains vehicles that satisfy all the hard or soft constraints and their current assignments. The "Available" list contains other

feasible alternatives. The final list contains other vehicles that are not available but can be re-assigned if needed.

System Architecture

The FMS application was designed with a modern architecture that is typical of most web 2.0 applications. Client-side is a JavaScript-based rich Internet application (RIA) that allows the designers to leverage open-source tools and libraries for the user interfaces and interactivities, such as spreadsheets and Gantt charts, and to focus on solving the AI problem. The server-side was coded in Python using a lightweight Python web application framework and AI platform.

The Python web application framework follows the model-view-controller (MVC) design pattern and provides a set of standard web framework features, such as user authentication, template engine, dispatching, and so on.

The AI platform provides a set of standard AI algorithms, such as rule processing, constraint solver, genetic algorithm, data mining, clustering, neural networks, and so on. The platform uses

SWIRE TRAVEL AIRPORT LIMOUSINE SERVICES LIMITED													
FILE ORDERS CARS FLIGHTS GANTT LOGOUT testing													
Date: 2010-01-11				Refresh	ARR	DEP	ALL	COMPLETED					
Order No	Driver	Car/Asgn	Status	Type	Car Type	Pickup	Flight	Fit Status	Passenger	Pick up location	Drop off location	Comments	Notes
SH1		372	PaxOnBoard	O/H	Sedan	06:30			M. [REDACTED] NG	SHEUNG SHUI	AIRPORT		RC [REDACTED] 3 [REDACTED]
CA00093870			PaxOnBoard	dep	Sedan	08:00	CA 1398	no match	Z. [REDACTED] MR	1 [REDACTED] D F [REDACTED] G K [REDACTED]	SHENZHEN AIRPORT		RC [REDACTED] 3 [REDACTED] SH [REDACTED] 16 [REDACTED] D 16 [REDACTED] 16 [REDACTED] D # [REDACTED] (Exp 11 [REDACTED] PC [REDACTED] T- SU [REDACTED]
SH3		394	PaxOnBoard	O/H	Sedan	10:00			M. [REDACTED] NG	SHERATON HOTEL	WEDDING		RC [REDACTED] 3 [REDACTED]
D1		388	PaxOnBoard	O/H	Sedan	10:30			T. [REDACTED] & K [REDACTED]	CONRAD HOTEL	WEDDING		WE [REDACTED]
D2		3909	PaxOnBoard	dep	Sedan	10:35			M. [REDACTED]	F [REDACTED] 4	AIRPORT		PC [REDACTED] 10 A [REDACTED]
CA00093910		388	Freezed	dep	Sedan	11:15			H. [REDACTED] T [REDACTED]	SHUN TAK CTR	AIRPORT	CAR CANCELLED, FULL CHARGE,	SH [REDACTED] T [REDACTED]
CA00093760		110	Freezed	dep	Sedan	12:00	CX 418	Sched: 14:05	H. [REDACTED] PHER P [REDACTED]	1 [REDACTED] H [REDACTED] Q [REDACTED] OAD, C [REDACTED] H [REDACTED] G	AIRPORT		
18245		274		dep	Sedan	12:00			M. [REDACTED]	S [REDACTED] H [REDACTED] 888	AIRPORT		
CA00093644		901	Freezed	dep	Sedan	12:05	IT 72	Sched: 14:05	R [REDACTED] T S [REDACTED]	4 [REDACTED] BANK T [REDACTED] C [REDACTED] OAD, H [REDACTED] G (V [REDACTED] BACK D [REDACTED]	AIRPORT		"WAIT AT MUARRY BLDG"
CA00093820		388	Freezed	dep	Sedan	12:05	CX 418	Sched: 14:05	H. [REDACTED] AN M [REDACTED]	S [REDACTED] I [REDACTED] ER, C [REDACTED] F [REDACTED] OAD, C [REDACTED] H [REDACTED] G	AIRPORT		- MUST SWIRE CAR - WAIT AT BACK DOOR
145057		617		dep	Sedan	12:15			M. [REDACTED]	C [REDACTED] OTEL R [REDACTED]	AIRPORT		
145058		476		dep	7 Seaters MPV	12:30			M. [REDACTED]	C [REDACTED] OTEL R [REDACTED]	AIRPORT		REQ VAN
18234				dep	Sedan	13:00			M. [REDACTED]	S [REDACTED] H [REDACTED] 1476	AIRPORT		
145059				dep	Sedan	13:30			M. [REDACTED]	C [REDACTED] OTEL R [REDACTED]	AIRPORT		
CA00093915			Freezed	dep	7 Seaters MPV	13:30			S [REDACTED] jo M [REDACTED]	SHUN TAK CENTRE	AIRPORT		

Figure 5. The FMS Orders Spreadsheet Screen.

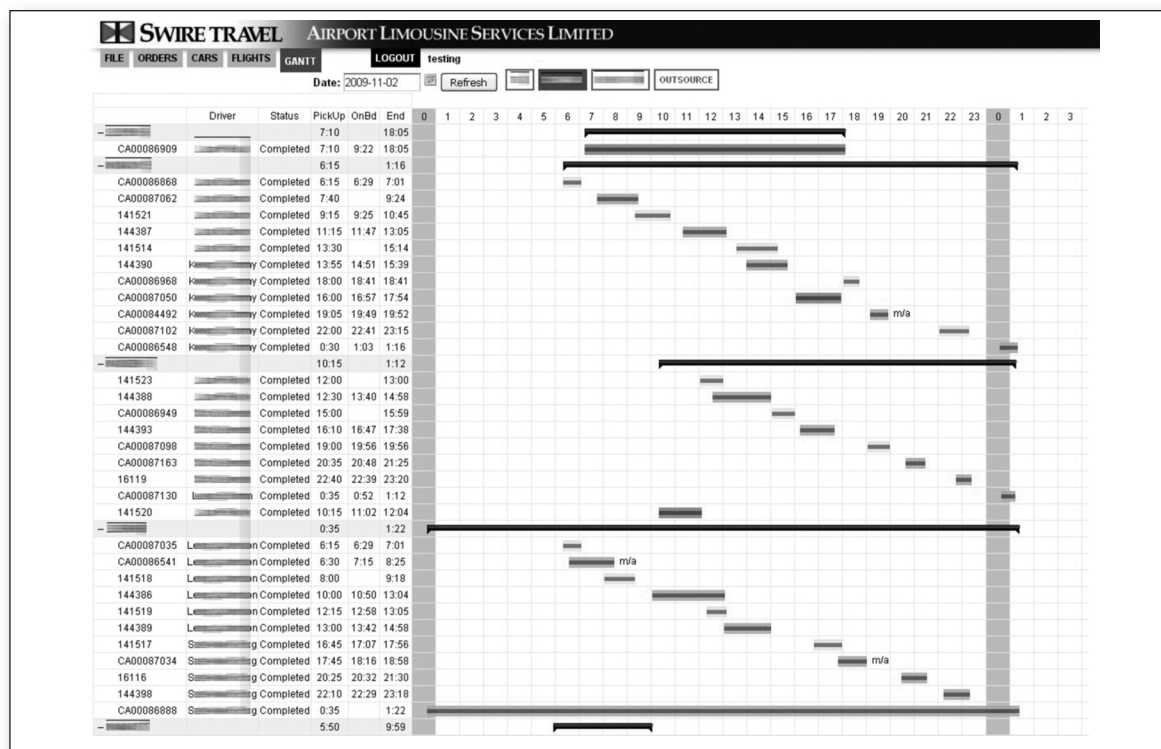


Figure 6. The FMS Gantt Chart Screen.

 SWIRE TRAVEL AIRPORT LIMOUSINE SERVICES LIMITED						
FILE	ORDERS	CARS	FLIGHTS	GANTT	LOGOUT	testing
Date: 2010-01-11 <input type="button" value="Refresh"/> <input type="button" value="ARR"/> <input type="button" value="DEP"/> <input type="button" value="ALL"/> <input type="button" value="COMPLETED"/>						
Flight No	Type	Time	City	Airline	Status	Hall/Terminal
CX 883,AA 6121	Arrival	05:40	Los Angeles	Cathay Pacific,AmericanAirlines	At gate 05:42	B
QF 029,AF 8073	Arrival	05:50	Melbourne	Qantas Airways,Air France	At gate 06:05	A
CX 746	Arrival	06:00	Jeddah,Dubai	Cathay Pacific	At gate 05:49	B
9W 072	Arrival	06:20	Delhi	Jet Airways	At gate 06:13	A
SQ 001	Arrival	07:10	San Francisco	Singapore Airlines	At gate 06:33	B
CX 156,AY 5850	Arrival	07:25	Brisbane	Cathay Pacific,FINNAIR	At gate 07:16	B
CX 881,AA 6119	Arrival	07:30	Los Angeles	Cathay Pacific,AmericanAirlines	At gate 07:21	B
CX 288	Arrival	07:50	Frankfurt	Cathay Pacific	At gate 10:00	B
CX 889,AA 6123	Arrival	07:50	New York/JFK,Vancouver	Cathay Pacific,AmericanAirlines	At gate 07:50	B
AY 067	Arrival	07:55	Helsinki	FINNAIR	At gate 08:43	A
CX 463	Arrival	08:00	Taipei	Cathay Pacific	At gate 07:47	B
PR 300	Arrival	10:10	Manila	Philippine Airlines	At gate 10:08	A
CX 684	Arrival	11:55	Dubai,Mumbai	Cathay Pacific	Est at 12:12	B
CX 710,AA 6094	Arrival	11:55	Singapore	Cathay Pacific,AmericanAirlines	Landed 11:47	B
CX 700,AA 6085	Arrival	12:10	Colombo,Bangkok	Cathay Pacific,AmericanAirlines	Est at 11:46	B
JL 731	Arrival	14:00	Tokyo	Japan Airlines	Est at 13:46	A

Figure 7. The FMS Flight Information Screen.

automatic code generation from XML to define the AI knowledge and parameters. The code generator supports Python, Java, and .NET and has already been used on a variety of deployed AI applications (Chun et al. 1999, Chun et al. 2000, Chun and Yeung 2004, Chun et al. 2005, Chun 2007, and Chun 2008).

Process Flow

The process flow for FMS is illustrated in figure 9. Under the “Planner/Controller” actor are the main application use cases—“Daily Orders,” “AI Scheduling,” and “Limousine Statuses.”

“Daily Orders” Use Case. Orders for next day are retrieved at midnight from the hotels and Airport Limousine’s reservation system. All order data are in English; some comments may be in Chinese. Flight information and flight statuses for each airport pickup and drop-off order are automatically retrieved from the airport system. In addition, individual orders are also received throughout the day, usually from hotels.

Order data submitted to FMS first go through an “autocorrecting” data preprocessing step before they are displayed in Gantt charts and spreadsheets. This step ensures that address information, which is in English, is accurate so that travel time prediction (described in a later section) can be made. The autocorrecting algorithm checks each address to see if it contains a known district or area. If not, a series of “transformers” are applied in order until a known district or area is found:










CA00093789 [AI] AI Suggestion:  528: no assignment  177 [CA00093354] 12:20-13:52 dep [CA00093905] 15:00-15:34 arr  120: no assignment  388 [CA00093733] 06:30-07:47 dep [CA00093366] 07:30-08:27 arr [CA00093901] 12:45-13:52 dep [CA00093852] 17:10-17:15 arr Available:  301 [CA00093695] 12:30-13:25 dep [CA00093861] 14:00-15:36 dep  119 [CA00093781] 12:35-13:25 dep [17238] 13:25-14:04 arr [CA00093668] 15:00-15:35 dep  539 [CA00093135] 07:00-07:47 dep [CA00090425] 09:00-16:03 O/H Not Available:  511 [CA00093904] 17:00-? O/H  192 [CA00093902] 18:00-? O/H	edan	19:10	KA 870	Sched: 21:10
	edan	20:15		
	edan	20:30		
	edan	21:00	AF 185	Sched: 23:35
	Seaters IPV	21:00		
	edan	21:45	CX 251	Dep 00:23 (10/1/2010 23:55)
	edan	21:55	CX 251	Dep 00:23 (10/1/2010 23:55)
	Seaters IPV	22:00	CX 255	Dep 01:39

Figure 8. The AI Proposal Pop-up.

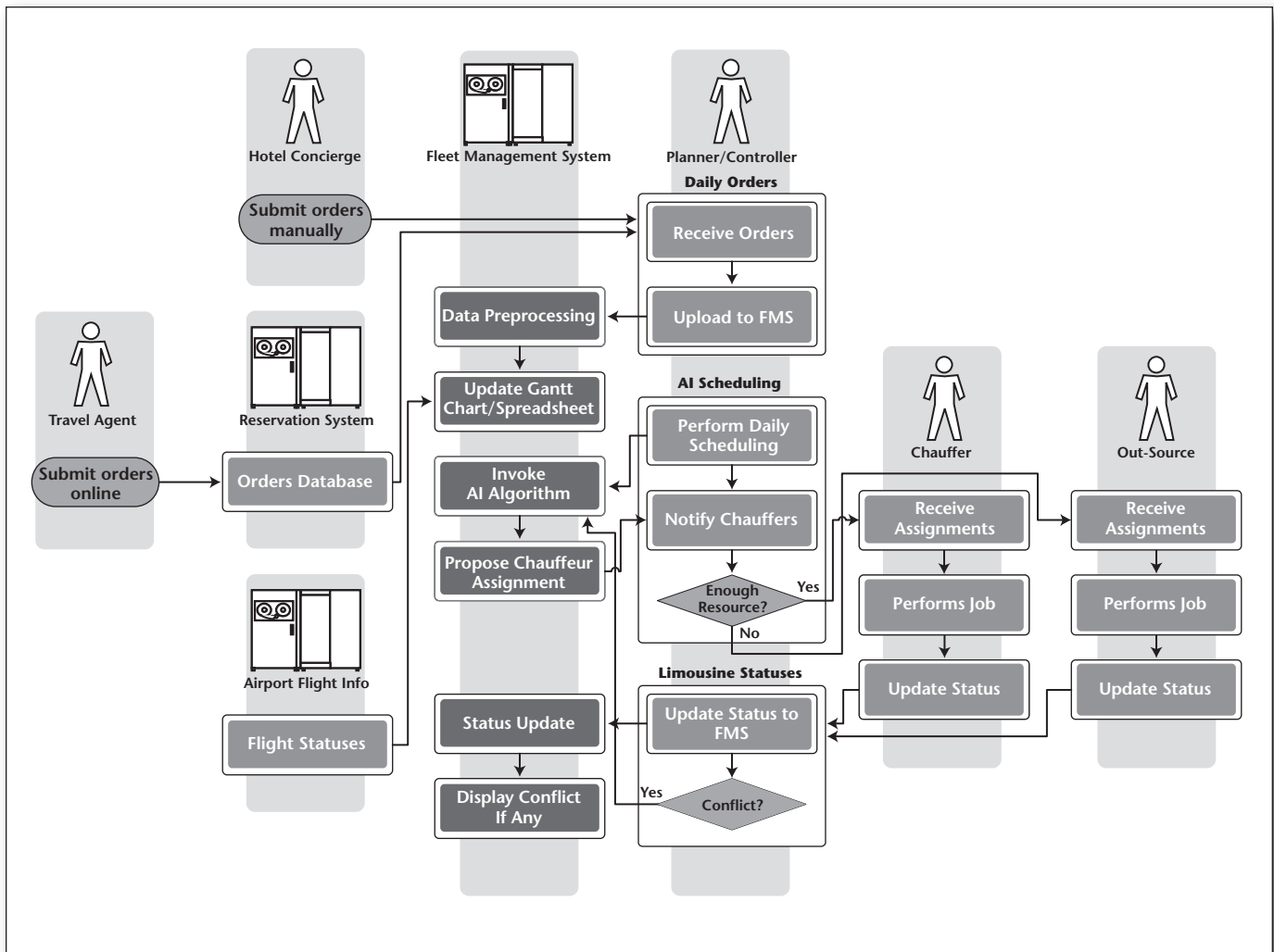


Figure 9. The FMS Process Flow.

1. *Spell-check*: FMS uses a variation of Norvig's spelling corrector algorithm (Norvig 2010), modified to work on a corpus of local location names.

2. *Normalize*: Abbreviations, informal names, and alternative names are then converted to standard formats.

3. *Map*: Names of buildings, landmarks, hotels, roads, streets, and so on are then mapped to known districts or areas.

4. *Similarity-check*: If the district or area still cannot be found, that information is then deduced from similar addresses taken from historical data that do have district or area information. Similarity is measured using a straightforward percentage of matched words and tokens. More complex algorithms based on Levenshtein distance (Levenshtein 1966) or soundex (NARA 2010) were not used, as Airport Limousine's clientele has similar demographic profiles and similar address samples were readily available in the historical data.

If no match is found after all "transformers"

have been applied, an "unknown" district and default travel time will be assigned.

"AI Scheduling" Use Case. Before the morning shift, the AI scheduling algorithm will be used to assign a limousine to each order based on rules and constraints (described later). The scheduling is usually done within a certain time window, such as the next two or three hours, as orders will change frequently. Chauffeurs, working in shifts, will be dispatched by phone. Based on AI proposals, the controller will confirm assignment and notify individual chauffeurs. The AI algorithm also identifies resource shortages. If Airport Limousine does not have enough limousines or chauffeurs, the orders will be outsourced to a pool of partners.

"Limousine Statuses" Use Case. As jobs are performed and orders completed, the chauffeurs will call the controller to update their locations and statuses, such as arrival time, passenger on-board time, and so on. Controllers enter these statuses

into FMS. Conflicts will automatically be highlighted and warnings displayed. The AI algorithm can be invoked again to resolve these conflicts.

AI Scheduling

In FMS, the vehicle scheduling problem is modeled as a constraint-satisfaction problem (Cohen 1990, Van Hentenryck 1989, Steele 1980, Kumar 1992, Tsang 1993). A solution is found through the assignment of values to variables subjected to a set of constraints.

CSP Model

CSP can be defined as consisting of a finite set of n variables v_1, v_2, \dots, v_n , a set of domains d_1, d_2, \dots, d_n , and a set of constraint relations c_1, c_2, \dots, c_m . Each d_i defines a finite set of values (labels) that variable v_i may be assigned. A constraint c_j specifies the consistent or inconsistent choices among variables and is defined as a subset of the Cartesian product: $c_j \subseteq d_1 \times d_2 \times \dots \times d_n$. The goal of a CSP algorithm is to find one tuple from $d_1 \times d_2 \times \dots \times d_n$ such that n assignments of values to variables satisfy all constraints simultaneously.

When the FMS limousine assignment problem is formulated as a CSP, each variable v_n represents a customer order than needs to be assigned a limousine. Each variable in FMS is associated with a "trip" object that defines the pickup time, pickup location, destination, stopover points, car type (capacity), and services.

The domain of each variable, d_n , will initially contain a set of all potential values, that is, representing all possible limousines. Each value is associated with a "limousine" object that contains details, such as car type, year, make, capacity, licenses, and other features.

The CSP constraints, c_j , are restrictions on how values, that is, limousines, can be assigned to variables, that is, orders. Constraints in FMS are classified into hard constraints, which cannot be violated, and soft constraints. Soft constraints are modeled as search heuristics.

Soft Constraints (Search Heuristics)

FMS uses both "select value" and "select variable" search heuristics to model soft constraints. The select variable heuristics control the ordering or selection of CSP variables, that is, v_n , to instantiate. The select value heuristics, on the other hand, control the preferred ordering of values within each domain, d_n , to use for instantiation of a variable v_n .

Select Variable Heuristic. In FMS, each variable v_n represents an order, created either by Airport Limousine travel agents, online customers, or hotel concierges on behalf of their guests. In selecting which variable to instantiate, FMS uses the following heuristics, in order of priority:

- (1) VIP orders (such as celebrities, politicians, and so on).
- (2) Cross-border orders (to the mainland of China).
- (3) On-hire orders (full day bookings).
- (4) Orders with preassigned vehicles (usually for special vehicles, such as stretch limos, buses, and so on).
- (5) All remaining orders.

Obviously, VIPs must be assigned first. In general, the number of VIPs in a day is manageable, so they will all be handled using Airport Limousine's own cars for quality assurance. Next in priority are orders that require crossing the border to the mainland of China. These orders require vehicles with two different license plates—one for Hong Kong and another for the mainland. On-hire orders are next in priority. These are orders that book a car for a whole day. For some orders, a preassigned vehicle will be specified. These are usually orders that require special car types.

Select Value Heuristic. Once a variable is selected, FMS assigns a value from the variable domain d_i . The variable, which represents a particular vehicle, is selected according to the following preference ordering:

- (1) Vehicle belonging to appropriate "team."
- (2) Vehicle is in vicinity.
- (3) Vehicle size and grade.
- (4) All remaining vehicles.

For hotels with a large number of daily orders, a special "team" of vehicles will be dedicated to serving their guests. Obviously, vehicles in appropriate teams should be selected first if available. Next in priority are vehicles already in the vicinity. For example, an airport pickup should first be assigned to a vehicle that is already at the airport. Likewise, a hotel pickup should be serviced by a vehicle that is already in the city. The next priority is the size and grade of the vehicle. Larger or better-equipped vehicles should be assigned only if no other vehicle is available.

Hard Constraints

Hard constraints are those that can never be violated. For FMS, the main hard constraints are vehicle type and vehicle availability. These hard constraints are encoded and verified using forward-chaining rules.

Vehicle types are general classes of cars, such as Mercedes-Benz sedans, stretch limousines, shuttle buses, cross border vehicles, and so on. Once booked, the vehicle type can be changed only with approval from the client. Within each class are variations, such as seating capacity, car features, year manufactured, specific model, and so on.

Obviously, a vehicle must be available in order to be assigned. This includes vehicle availability for the day (not in shop for service) or whether it is already assigned to service another order. Time

```

Let  V be a set of variables
     D be a set of domain values
     C be a set of constraints
     T be a time period

FMS_Scheduler(V, D, C, T):
  while un-instantiated variable in V within T do:
     $v_i = \text{SelectVariableHeuristics}(V, T)$ 
    foreach value  $x_i$  in  $\text{SelectValueHeuristics}(v_i, D)$  do:
      assign  $x_i$  to  $v_i$ 
      if  $\text{RuleViolation}(V, C)$ : continue
      else  $\text{Propagate}(x_i, v_i)$ : break
      end if
    end foreach
    if  $v_i$  is still unassigned: assign none to  $v_i$ 
  end while

SelectVariableHeuristics(V, T):
  returns the next un-instantiated variable from V within T based
  on a set of preference ordering

SelectValueHeuristics(D):
  returns the next value from D based on a set of preference
  ordering

RuleViolation(V, C):
  returns True if any hard constraints from C are violated

Propagate( $x_i, v_i$ ):
  propagate consequence of assigning  $x_i$  to  $v_i$ , such as availability
  and vehicle location and store in associated objects and their data
  structures

```

Figure 10. Pseudocode for the Scheduling Algorithm.

needed to travel between orders is also factored into availability. The algorithm to compute travel time is described later.

Scheduling Algorithm

The AI scheduling algorithm works on top of the CSP model. Although the problem was modeled as a CSP, it does not use a constraint satisfaction engine. Instead, it uses a hybrid design scheduling algorithm. For the FMS limousine scheduling problem, a schedule is generated by repeated selection of a variable, v_n (using the select variable heuristic), assigning a value (using the select value heuristic), and then verifying the solution with rules (using a rule engine) (Apt and Monfroy 2001). Figure 10 depicts the pseudocode for the scheduling algorithm.

In FMS, scheduling is done without backtracking and within a limited time window, such as two or three hours. A time window is used since the

limousine business is quite dynamic. Orders get added, canceled, and changed quite frequently. In addition, there may be frequent traffic and flight delays. In such an unpredictable environment, there is little benefit to using backtracking to optimize a solution, as any optimized solution will quickly become nonoptimal with these changes. Instead, the AI scheduling algorithm finds a fast solution following clearly defined heuristics and rules. This also has the side benefit of making the AI solution easier to be understood by the planners.

In the event that the AI scheduling algorithm cannot find a suitable vehicle to fill an order, that order will be flagged and a preapproved outsourcing company will be contacted by the planner to provide the service.

The use of select variable heuristics and select value heuristics in the CSP model is similar to the classical ordered frame instantiation approach (Engelman, Scarl, and Berg 1980) that prioritizes slots (CSP variables) to fill as well as potential slot fillers (CSP values). This approach is highly effective in reducing the search time required to find a satisfying solution relative to search over an unordered slots and slot fillers. Similar to the algorithm, Evertsz and Motta also proposed a hybrid rule and frame-based system (Evertsz and Motta 1991) to handle constraints.

Travel Time Prediction

Unlike traditional AI vehicle scheduling problems, the limousine scheduling problem requires precision in timing, particularly target pickup times. In FMS, a self-adapting module that automatically adjusts travel time based on historical values was created. Geographically, Hong Kong is divided into 18 districts (figure 11). These districts are further divided into areas. In total, there are roughly 200 areas in Hong Kong. These districts and areas are used as the basis for the drive time prediction algorithm.

The FMS travel time predication algorithm is a variation of the one proposed by Lee, Chowdhury, and Chang (2008). In FMS, there are two types of matrices, **S** and **H**. There is only one instance of matrix **S**, and it represents the “standard” travel time that is defined manually by planners. This is done once only when the system is set up, but can also be changed anytime later if needed. Matrix type **H** represents historical travel times and is computed as an average over a time window, such as one month. The time window captures seasonal or temporary fluctuations to travel times. There is a set of **H** matrices. Each matrix, $H_{\text{dow, tp}}$, represents the travel time for a particular day of the week (dow) and time period (tp). Similar to Lee, Chowdhury, and Chang (2008), the time-period is divid-

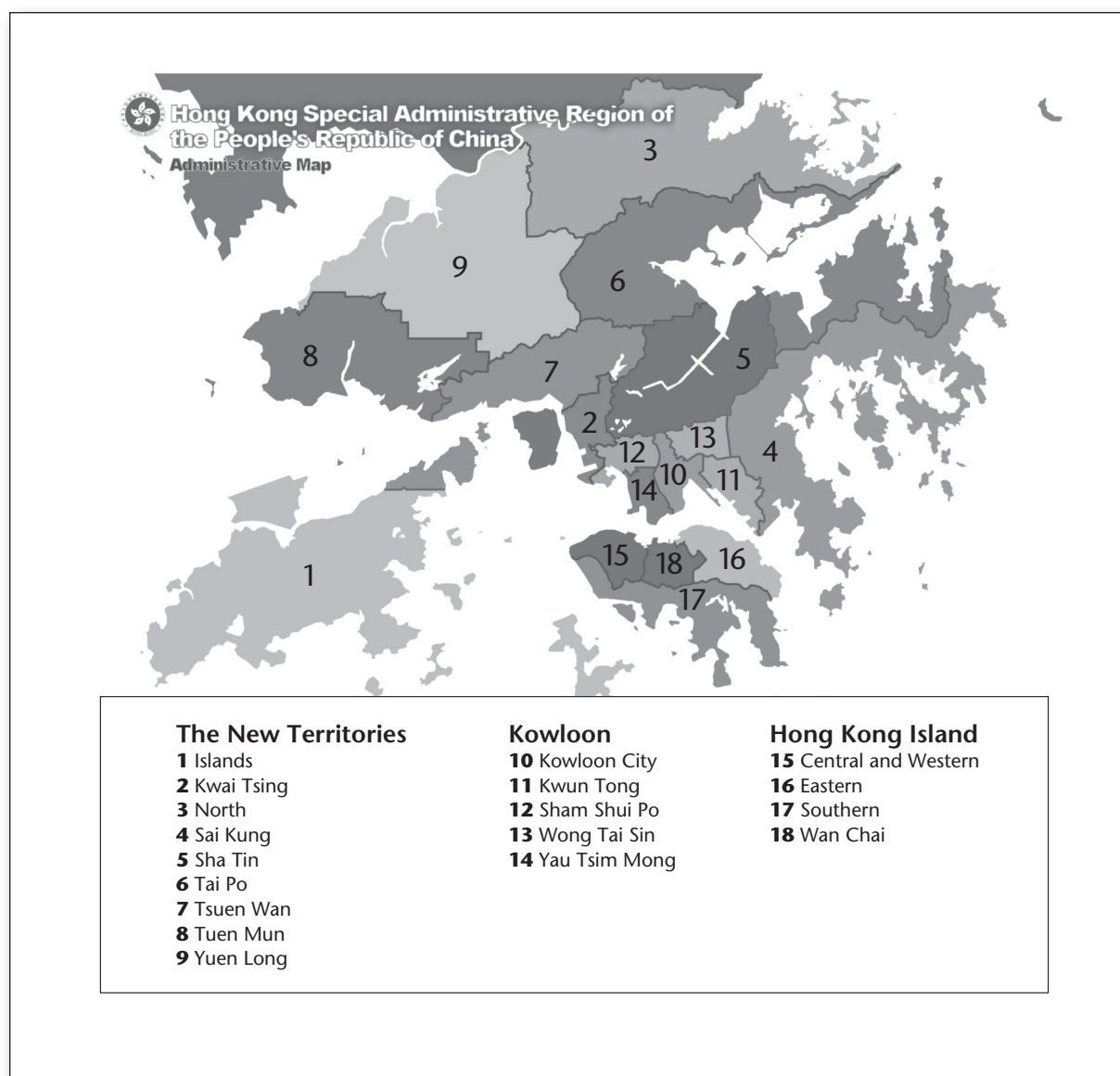


Figure 11. Map Showing the 18 Districts of Hong Kong.

Creative Commons diagram by Moddlyg.

ed into nine time ranges—morning rush hour, morning, early noon, lunch time, afternoon, evening, evening rush hour, night, and late night.

All matrices are $A \times A$, where A is the number of Hong Kong geographic areas recognized by FMS. Each element of the matrix, $t_{i,j}$, represents the average travel time from area a_i to area a_j where $i \neq j$. For matrix S , the values of $t_{i,j}$ are provided by human planners based on experience. For the $H_{\text{dow,tp}}$ matrices, the $t_{i,j}$ values are computed based on historical data. The predicted travel time used by the AI scheduling algorithm is the average of the values from matrices S and the particular $H_{\text{dow,tp}}$ for

the day of the week and time of a particular customer order. The averaging allows better control over the estimated travel time by the controllers.

Application Use and Payoff

FMS was deployed in two phases—first in January 2009 for manual assignment, and then in June 2009 for AI assignment. Phase 1 provided the core system with all the screens and features needed to support manual scheduling and vehicle dispatching. Phase 2 added AI vehicle scheduling support and a Gantt chart.

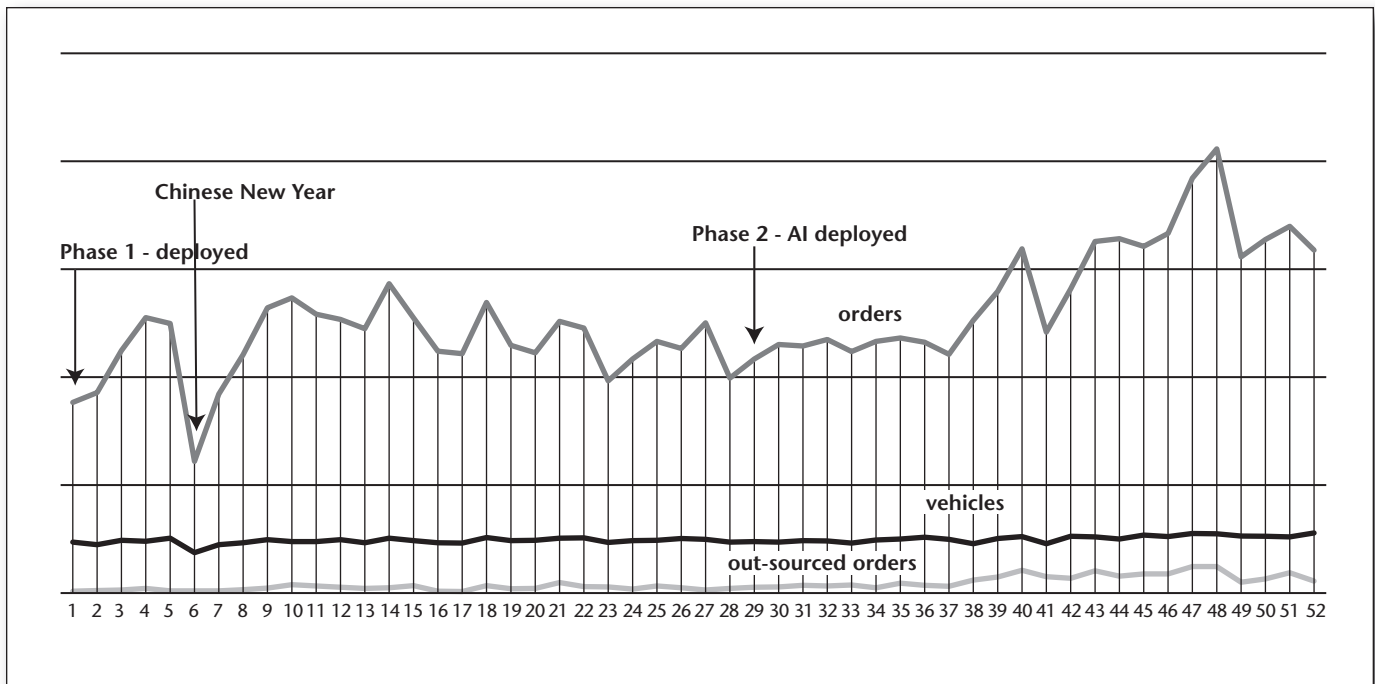


Figure 12. Plot Showing Deployment Schedule and How AI Supported Business Growth.

Application Payoff

The use of AI is believed to have made Airport Limousine more effective and efficient. Prior to AI deployment, Airport Limousine saw the lack of experienced planners and controllers as a major bottleneck to business growth. With AI, Airport Limousine was able to greatly increase the number of customer orders and vehicle utilization without additional planners and controllers.

Figure 12 shows the number of orders per week, the number of vehicles used to service these orders, and the number of outsourced orders. The horizontal scale is the week number. The vertical scale is the relative number of vehicles. Although the chart does not show data for the pre-FMS pencil-and-paper operation, it is similar to that of phase 1, which is FMS but without the AI features. The benefits of AI do not come in until phase 2. After AI was deployed, Airport Limousine was able to handle a 100 percent increase in the number of weekly orders without additional planners and controllers. This was unthinkable before.

Despite the over 100 percent increase in orders, the number of vehicles and drivers deployed remained fairly constant, as figure 12 indicates. With AI scheduling, Airport Limousine was able to optimize how its existing resources were deployed to meet increasing demands. Figure 13 shows the average number of orders serviced per vehicle per day after the phase 2 AI deployment. Roughly two more orders were handled per vehicle each day—a 40 percent improvement in utilization.

Although there is a slight increase in outsourcing due to the dramatic increase in business volume, the number of outsourced orders was still maintained to be within only a few percentage points of total orders throughout the year (see figure 11).

Besides the AI benefits, the data visualization provided by FMS also contributed to improving utilization. All information related to orders, vehicles, drivers, and flights are consolidated in one place and displayed in easy to digest Gantt charts and spreadsheets formats. Planners/controllers can instantly visualize operation statuses as well as assignment statuses and workload for each vehicle for the entire day. With the bird's-eye view and the AI proposal mechanism, Airport Limousine is better equipped to handle sudden changes, such as last-minute orders, and delays. This allows Airport Limousine to be more agile when dealing with business partners and customers.

Application Development

Development began in April 2008 and was performed by CityU Professional Services Limited, a nonprofit subsidiary of the City University of Hong Kong.

Application development was performed using a modern agile iterative approach and user-centered design. The result of the first iteration was released for user feedback in July 2008. After two more iterative refinements, phase 1 of the project was completed in December 2008 and was

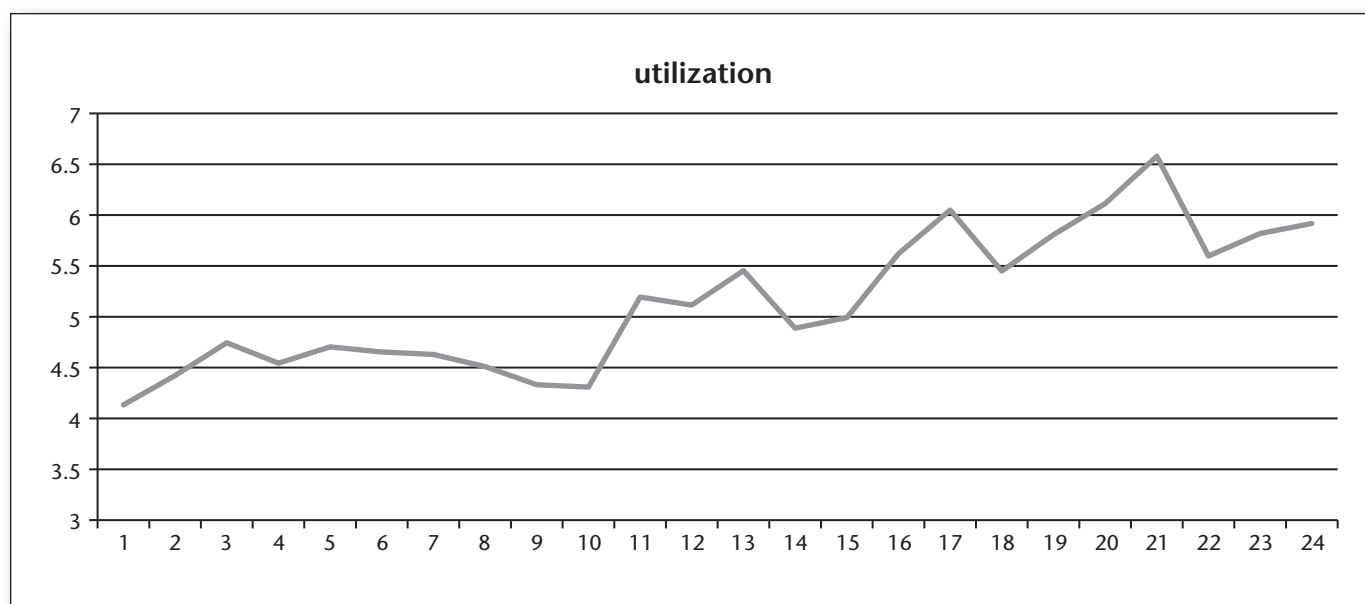


Figure 13. Average Number of Orders Serviced per Vehicle.

deployed 1 January 2009. Phase 1 provided core capabilities such as user-role management, a database to store vehicle and driver information, integration with booking data (from hotel, online web-based booking, and travel agents), integration with flight data (from airport), manual scheduling features (vehicle and driver assignments), data export features, and screens to display orders, vehicle location and statuses, driver assignments, and flight schedules and statuses.

The phase 2 AI work began in March 2009. The first AI release was deployed in early June 2009. After refinement, the final release was deployed in late July 2009. Phase 2 provided trip travel time estimation, AI scheduling capabilities, and a new Gantt chart screen.

Deployment

The phase 1–phase 2 deployment allows the designers to smooth the transition from a manual to an AI mode of operation. Phase 1 gives users time to get familiarized with the user interface and application features; decision making is still manual. During this stage, any remaining usability, integration, or system-level issues will be ironed out. Once the users are totally accustomed to using the application, phase 2 is deployed with AI. Because AI usually works behind the scene, this step of deployment is quite transparent to the user and will not require additional training.

This phase 1–phase 2 deployment method has been quite effective. Previously, when the entire AI system was deployed in one step, users were overwhelmed with both application and AI features. By

creating two different phases, potential deployment issues are isolated—namely the usability and system issues, the accuracy of the AI knowledge, and the quality of the AI-generated schedules. Isolating potential issues and solving them incrementally provides a more manageable model of AI deployment.

Maintenance

The frontline support and maintenance for FMS is provided by Airport Limousine's own IT team. Since FMS is a web-based solution, many of the traditional maintenance tasks, such as exporting data, are provided in "self-service" mode, for example, through downloading of Microsoft Excel files through the web. All site-specific configuration data are maintained as XML or Microsoft Excel files. For example, the limousine inventory and driver details are all stored as Excel files and uploaded whenever changed.

Because the AI rules and constraints are also coded in XML, ongoing maintenance is simplified. As described in the section on AI Scheduling, the rules and constraints are high level and do not deal with particular parameters. Hence, they are relatively static. If needed, additional constraints may be added, and their ordering may be adjusted to reflect changing priorities.

Related Work

There have been decades of work on different variations of the vehicle scheduling and routing problem (Ceder 2002, Wren 1998). These include

scheduling vehicles to pick up or deliver goods, mapping the shortest routes for vehicles to visit all destination points, dispatching vehicles from one or more depots, and so on. The constraints are usually the destinations, location of depots, travel distances, travel times, sizes of payload, capacity of vehicles, and so on. Many researchers model this problem as a type of traveling salesman problem. Some researchers offer solutions using constraint programming (Tsang 1993, Shaw 1998, Backer et al. 2000), linear programming (Foster and Ryan 1976), or expert systems (Waters 1990).

The limousine scheduling problem is a bit different from the traditional vehicle scheduling problem (VSP). For VSP, pickup and delivery times are usually flexible and can be scheduled. For limousines, pickup times are fixed. In fact, being on time is of utmost importance. The limousine scheduling problem is more dynamic; orders get cancelled and added throughout the day as well as the need to anticipate traffic delays.

Laurent and Hao (2007) proposed a combined vehicle and driver scheduling algorithm using a combination of constraint programming and simulated annealing. For their problem, driver scheduling was an issue. In this case, drivers are already associated with a team or vehicle. In Laurent and Hao's work, their limousines were dispatched to service roughly one job per day within the Paris area, whereas Airport Limousine needed to dispatch each limousine to service many jobs per day within the metropolitan Hong Kong area.

The limousine scheduling problem is related to taxi scheduling (Horn 2002). However, resources for taxi dispatching are a bit homogeneous and with less demand for special requests or value-added services. Borndörfer et al. (1999) outline a problem to schedule vehicles in a dial-a-ride system for handicapped people in Berlin—Telebus. The system operates 1000 to 1500 trips per day. They solved this massive problem by partitioning the trips and clustering them into smaller sets that can be scheduled using a branch-and-cut algorithm. Bus vehicle and driver scheduling are also related to this work (Huisman and Wagelmans 2006, Ismail and Ang 2005). However, the bus problem is mainly driven by a rigid timetable and fixed stops.

The designers also performed a survey of commercially available limousine software (LMS,² CorporateCarOnline,³ Limo-Aide,⁴ Trip Tracker,⁵ and LimoNexus⁶). As far as can be told, most of them provided only screens for a human planner to input schedules. Most relied on a traditional client-server model and were not as flexible or user friendly. However, most of these software packages do provide a fully integrated system that ranges from online reservation, scheduling, and dispatching, to billing.

Future Work

The current FMS uses AI for vehicle allocation only. Potentially, AI can be applied to optimize other areas of operations, such as scheduling, which shift a chauffeur should report to work, and the assignment of a chauffeur to a particular vehicle. Future enhancement will allow chauffeurs direct access to FMS through mobile devices so that they can retrieve assignments and client details (such as addresses, phones, and so on) themselves as well as update their current locations and statuses. This will ensure more accurate and timely data input into FMS as well as further reduce the workload and communication overhead for the planners and controllers.

Conclusion

This article is a case study of how AI was successfully used for mission-critical limousine scheduling using a modern web 2.0 architecture to improve efficiency and resource utilization, allowing the limousine company to increase business without sacrificing quality of service. The use of AI enabled Airport Limousine to handle a 100 percent growth in number of weekly business orders and achieve 40 percent improvement in resource utilization.

Notes

1. Source: United States Census Bureau.
2. See the Limousine Management Systems website, www.lmsgold.com.
3. See CorporateCarOnline, www.limosinesoftware.com.
4. See the Comp-Easy Software LLC website (www.compez.com/Software/Limo_Aide_tm_/limo_aide_tm_.html).
5. See the Trip Tracker website, www.triptracker.com.
6. See the BlueNexus website (www.bluenexus.com/limonexus.html).

References

- Apt, K. R., and Monfroy, E. 2001. Constraint Programming Viewed as Rule-Based Programming. *Theory and Practice of Logic Programming* 1(6)(November): 713–750.
- Backer, B. D.; Furnon, V.; Shaw, P.; Kilby, P.; and Prosser, P. 2000. Solving Vehicle Routing Problems Using Constraint Programming and Metaheuristics. *Journal of Heuristics* 6(4)(September): 501–523.
- Borndörfer, R.; Grötschel, M.; Klostermeier, F.; Küttner, C. 1999. Telebus Berlin: Vehicle Scheduling in a Dial-a-Ride System. In *Proceedings of the 7th International Workshop on Computer-Aided Transit Scheduling*, 391–422. Berlin: Springer-Verlag.
- Burbeck, S. 1992. Application Programming in Smalltalk-80: How to Use Model-View-Controller (MVC). University of Illinois in Urbana-Champaign (UIUC) Smalltalk Archive. (www.cs.uiuc.edu/users/smarch/st•docs/mvc.html).

Ceder, A. 2002. Urban Transit Scheduling: Framework, Review and Examples. *Journal of Urban Planning and Development* 128(4): 225–244.

Chun, H. W. 2007. Using AI for e-Government Automatic Assessment of Immigration Application Forms. In *Proceedings of the 19th Conference on Innovative Applications of Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Chun, H. W. 2008. Using AI for Olympic Equestrian Event Preparation. In *Proceedings of the 20th Conference on Innovative Applications of Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Chun, H. W.; Chan H. C.; Tsang M. F.; and Yeung, W. M. 1999. HKIA SAS: A Constraint-Based Airport Stand Allocation System Developed with Software Components. In *Proceedings of the Eleventh Conference on Innovative Applications of Artificial Intelligence*, 786–793. Menlo Park, CA: AAAI Press.

Chun, H. W.; Chan H. C.; Lam P. S.; Tsang M. F.; Wong, J.; and Yeung W. M. 2000. Nurse Rostering at the Hospital Authority of Hong Kong. In *Proceedings of the Twelfth Conference on Innovative Applications of Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Chun, H. W., and Yeung, W. M. 2004. Rule-Based Approach to the Validation of Subway Engineering Work Allocation Plans. In *Proceeding of the International Conference on Computing, Communications, and Control Technologies*, August 14–17. Winter Garden, FL: International Institute of Informatics and Systemics.

Chun, H. W.; Yeung, W. M.; Lam, P. S.; Lai, D.; Keefe, R.; Lam, J.; and Chan, H. 2005. Scheduling Engineering Works for the MTR Corporation in Hong Kong. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence*. Menlo Park, CA: AAAI Press.

Cohen, J. 1990. Constraint Logic Programming. *Communications of the ACM* 33(7): 52–68.

Engelman, C.; Scarl, E. A.; and Berg, C. H. 1980. Interactive Frame Instantiation. In *Proceedings of the First National Conference on Artificial Intelligence*, 184–196. Menlo Park, CA: AAAI Press.

Evertsz, R., and Motta, E. 1991. The Abstract Interpretation of Hybrid Rule/Frame-Based Systems. In *Trends in Artificial Intelligences*, Lecture Notes in Computer Science 549/1991, 147–156. Berlin: Springer-Verlag.

Foster, B. A., and Ryan, D. M. 1976. An Integer Programming Approach to the Vehicle Scheduling Problem. *Operational Research Quarterly* Part 1 27(2): 367–384.

Horn, M. E. T. 2002. Fleet Scheduling and Dispatching for Demand-Responsive Passenger Services. *Transportation Research*, Part C, 10(1): 35–63.

Huisman, D., and Wagelmans, A. P. M. 2006. A Solution Approach for Dynamic Vehicle and Crew Scheduling. *European Journal of Operational Research* 172(2)(16 July): 453–471.

Ismail, Z., and Ang, P. S. 2005. Integer Programming Approach in Bus Scheduling and Collection Optimization. *Jurnal Teknologi* 43 (C): 1–14.

Kumar, V. 1992. Algorithms for Constraint Satisfaction Problems: A Survey. *AI Magazine* 13(1): 32–44.

Laurent, B., and Hao, J. 2007. Simultaneous Vehicle and Driver Scheduling: A Case Study in a Limousine Rental Company. *Computers and Industrial Engineering*. 53(3) (Oct. 2007): 542–558.

Support AAAI Open Access

AAAI wishes to thank you for your ongoing support of the open access initiative and all AAAI programs through the continuation of your AAAI membership. We count on you to help us deliver the latest information about artificial intelligence to the scientific community. To enable us to continue this effort, we invite you to consider an additional gift to AAAI. For information on how you can contribute to the open access initiative, please see <http://www.aaai.org> and click on “Gifts.”

Lee, H.; Chowdhury, N. K.; and Chang, J. 2008. A New Travel Time Prediction Method for Intelligent Transportation Systems. In *Knowledge-Based Intelligent Information and Engineering Systems*, Lecture Notes in Computer Science 6277, ed. I. Lovrek, R. J. Howlett, and L. C. Jain. Berlin: Springer.

Levenshtein V. I. 1966 Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady* 10(8): 707–10.

Norvig, P. 2010. How to Write a Spelling Corrector. Unpublished ms. (Retrieved from norvig.com/spell-correct.html.)

Shaw, P. 1998. Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In *Proceedings of the 4th international Conference on Principles and Practice of Constraint Programming*, Lecture Notes in Computer Science 1520, ed. M. J. Maher and J. Puget, 417–431. Berlin: Springer-Verlag.

Steele Jr., G. L., The Definition and Implementation of a Computer Programming Language Based on Constraints, Ph.D. Thesis, MIT, 1980.

Tsang, E. 1993. *Foundations of Constraint Satisfaction*. Boston: Academic Press.

U.S. National Archives and Records Administration (NARA). 2010. The Soundex Indexing System. Washington, D.C.: U.S. National Archives and Records Administration. (www.archives.gov/research/census/soundex.html).

Van Hentenryck, P. 1989. *Constraint Satisfaction in Logic Programming*. Cambridge, MA: The MIT Press.

Waters, C. D. J. 1990. Expert Systems for Vehicle Scheduling. *Journal of the Operational Research Society* 41(6): 505–515.

Wren, A. 1998. Heuristics Ancient and Modern: Transport Scheduling Through the Ages. *Journal of Heuristics* 4(1): 87–100.

Andy Chun is the chief information officer at the City University of Hong Kong and an associate professor in the Department of Computer Science. His research interests include web technologies; search engine optimization; scheduling, rostering, optimization; business intelligence, data mining; text mining; and distributed architectures. He received a B.S. from the Illinois Institute of Technology and an M.S. and a Ph.D. in electrical engineering from the University of Illinois at Urbana-Champaign.