Using AI for Olympic Equestrian Event Preparation

Andy Hon Wai CHUN

Associate Professor
City University of Hong Kong
Department of Computer Science
Tat Chee Avenue, Kowloon Tong
Hong Kong SAR
andy.chun@cityu.edu.hk

Abstract

This paper describes our experience in using modern Web 2.0 architecture, lightweight Python frameworks, and rapid prototyping to create an AI rostering and workforce management system to help prepare for the 2008 Beijing Olympic Equestrian Events, which will be held in Hong Kong. As part of the 2008 Olympics preparation, a scaled down "test" event – "The Good Luck Beijing – HKSAR 10th Anniversary Cup" was held in August 2007 as a dressed rehearsal for the Olympics Equestrian Events. To schedule and manage all the volunteers and staffs of this event, a Web-based AI "Workforce Management System" (WMS) was created. By leveraging on our existing Web-based application framework and AI platform, we were able to complete the entire AI project under very tight time constraints and helped Hong Kong successfully pass the readiness requirements of the International Olympic Committee. This paper focuses on our experience with using AI to support the Good Luck Beijing equestrian games. It describes the AI development approach we took as well as the AI architecture for the HR-XML compliant rostering system.

Introduction

Most of the 2008 Olympic competitions will be held in Beijing. However, a few competitions will be held in other cities in China. For example, Hong Kong will host all the Olympic and Paralympic Equestrian Events (COC 2007a), which include individual and team competitions in dressage, jumping and eventing. The city has a long history of organizing horse racing events as well as world-class equine care facilities and medical services for horses. These events will take place at two different competition venues – a core venue for jumping and dressage and another venue for the cross country event (HKEquestrian 2007a, Beijing2008 2007c).

The "Good Luck Beijing" Games

To prepare for the 2008 Games and as a test of the organizer's readiness, a scaled-downed version of the Olympics was held in August/September 2007. In HK, these "test" games were called the "Good Luck Beijing — HKSAR 10th Anniversary Cup" to concurrently celebrate the 10th anniversary of the HK hand over. These games tested all aspects related to the actual Olympic Games, such as competition venues, logistics, traffic control, medical services and security (COC 2007b, 2007c).

Manpower Needs

Although the actual Good Luck Beijing equestrian events span only a few days, staffs were needed long before and after the events to handle different publicity/marketing activities, logistics, venue preparation/shutdown, and various paperwork. To support all the work needed before, during and after the games, roughly a thousand volunteers and part-time staffs were needed. Ensuring adequate levels of manpower resource to handle all types of event-related activities is the main AI task.

The AI Problem

There were many challenges to this workforce management problem. The AI problem is to assign, for each shift and job duty, an appropriate number of adequately skilled volunteers/staffs, while optimizing several factors. Firstly, since the workforce consists of volunteers and part-time staff, their availability varied greatly. Secondly, volunteers and staff were needed to service a wide variety of tasks, ranging from medical and security services to liaising with media. It was important that staff assigned had relevant experiences and/or trainings. Thirdly, although there are two main competition venues, staff might need to work at a variety of locations outside the competition venues. It was important to reduce the amount of traveling needed. Fourthly, unlike other rostering problems, most of the work shifts (start time and duration) were not fixed and varied daily depending on job nature and workload.

Project Objectives

The WMS project had several key objectives. Firstly, and most importantly, is to create efficient and effective daily

rosters by optimizing the use of their human resources. Secondly, is to help the organizers cope with unexpected delays and event cancellations. (Coincidentally, a typhoon passed through HK just before the test events and caused some minor delays and schedule changes.) Thirdly, the system should act as the portal to disseminate all information related to schedules and assignments. Fourthly, rostering data must be made available for potential use by other systems, such as systems used in Beijing. The fifth objective, which is also most crucial, is to be able to complete the entire project within less than two months' time.

Originally, the HK event organizers had thought that an off-the-shelf software package might be adequate. However, they quickly realize that their scheduling needs were quite unique. After an extensive search for off-the-shelf software failed to identify one that satisfies the organizer's needs, the organizers commissioned the City University of HK (CityU) to create a system around mid-May with deployment by end of June. Even though CityU had quite a long history of creating mission critical AI scheduling and rostering systems for HK (Chun 1999, 2000, 2005, 2007), it was still quite a challenge to put a system together, with relatively demanding IT requirements, within such a short time. This paper outlines the approach we took and how we successfully deploy the system in time and helped Hong Kong pass the International Olympic Committee's readiness requirements.

Application Description

The Workforce Management System (WMS) for the Good Luck Beijing equestrian events is a typical Web 2.0 JavaScript-based rich internet application (RIA). Server-side services were implemented using our own lightweight Python application framework and template engine together with our existing AI platform.

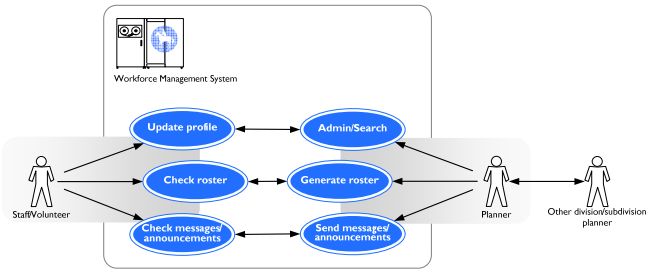

Figure 1. High-level Use case Diagram for WMS

The WMS is a secure role-based application with 2 main classes of users – volunteers/staffs, and administrators/ planners. Volunteers/staffs have access to WMS features to update their profiles, such as addresses, phone numbers, etc., as well as availability and uniform size information. They can of course use WMS to check their rosters/ schedules, as well as individual/group messages and announcements from the administrators/planners. Figure 1 is a high-level use case diagram for WMS.

Administrators/planners have access to WMS administrative features to search and retrieve information related to individual volunteers/staffs, define shifts and generate rosters, and send messages/announcements.

The WMS users are divided into 9 "divisions" and 38 "subdivisions" that correspond to the function and activities they might be performing. For example, the Competition division is further divided into 15 subdivisions – ambulance assistant, arena party, cross points, cooling team, data entry, fence assistant, information officer, medical control program, quarantine control point, radio operator, runner, scorer, stable management, training arena assistant, and vet assistant. Volunteers/staff are assigned to the subdivisions based on their qualifications and trainings. Rostering is usually done within each subdivision. However, if manpower is short, borrowing staff from another subdivision is permitted.

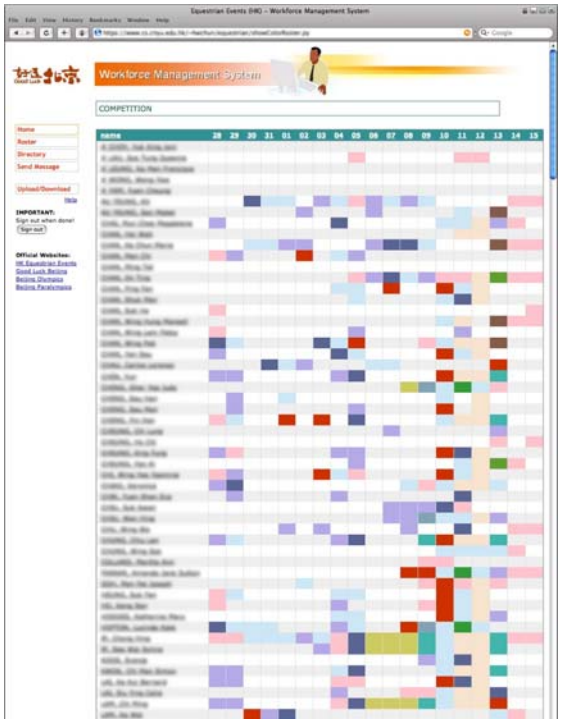

Figure 2. The WMS Roster Screen

Figure 2 is an example of a typical "administrator" screen in the WMS; this particular screen shows part of the roster for one of the divisions. The cells are colored to indicate the first shift the volunteer/staff is assigned to for a particular day. White cells indicate that a volunteer/staff is not available on that day. Red indicates a conflict; either the staff is not available or overlapping assignments were made. Pink indicates that the volunteer/staff is available but not assigned any duties. Mousing over the cell displays details of the assignment. Clicking on any cell will display a form to allow the administrator to edit the assignment.

System Architecture

The Workforce Management System follows a Model-View-Controller (MVC) (Burbeck 1992) architectural design pattern, which is popular for many modern web-based applications. For implementation, we used our own lightweight Python-based MVC framework called the "FOA Application Framework," coupled with our own lightweight "FOA Template Engine" to implement the views within MVC. The WMS operates within a Solaris environment using Apache as the web server. Figure 3 shows the overall architecture for WMS.

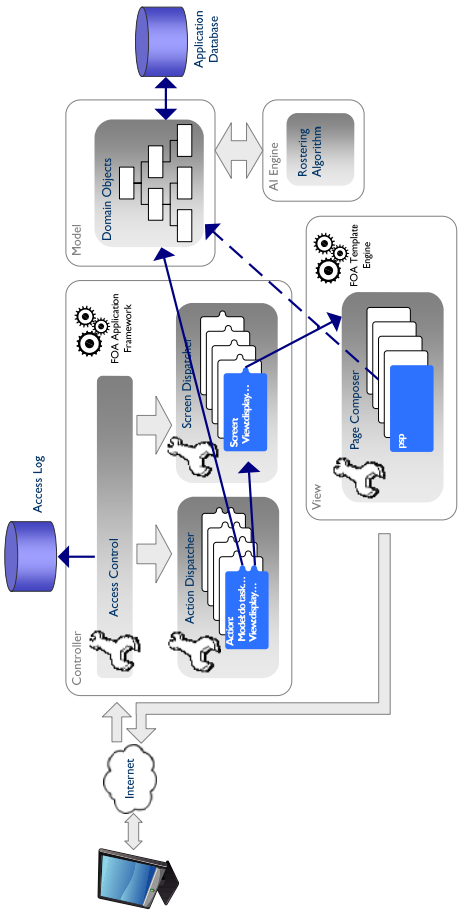

Figure 3. The Architecture of the WMS.

Both the application framework and template engine makes heavy use of Python meta-programming using "decorators" to simplify application coding.

FOA Template Engine

The main task for the template engine is to provide mechanisms to define screen templates for dynamic web pages. Our lightweight template engine operates on top of the Apache web server using the mod_python module and Python Server Pages (PSP). However, PSP technology alone only allows one to embed server-side Python code into HTML pages. It does not provide a well-defined mechanism to define page layout, page components nor dynamic page composition, i.e., how components of a page change depending on desired view or user profile. These tasks are performed by the template engine, which allows us to quickly and very easily create a template layout, define the components of the template, and have dynamic content be filled in using PSP. A new screen can be added within minutes. This greatly improves productivity during application development; our experiments show that we achieve a 90% savings in code size.

FOA Application Framework

The application framework implements an MVC architecture and provides secured role-based action dispatching and screen dispatching. The main function of the application framework is to act as the "Controller" in the MVC model. The "View" is provided by the previously described template engine. And the "Model" is provided by domain objects, which we will explain later.

As the "controller," the application framework provides three key functions – role- based access control, role-based action dispatching and role-based screen dispatching. The action/screen dispatching mechanism is similar in concept to Jakarta Struts.

The way our application framework works is that each click (or page access) must first go through the "Access Control" module to be authenticated, role privilege checked, and then logged. Then, depending on the action, the framework may either call the Action or Screen Dispatcher.

If user action is a simple screen retrieval, the Screen Dispatcher will request the template engine to create the dynamic page, potentially using data from the domain "Model." The dispatchable screen is defined using a "decorator" called "screenHandler." For example, the following two lines of code define a screen that can be accessed by users with "admin" role or higher only.

```
@screenHandler('admin', menu=adminMenu)
def reporting(req, **kwargs): pass
```

The screen will be created using the configured template and will be customized with "adminMenu" replacing the default menu and dynamic content will be taken from "reporting.psp." Some of the WMS screens, which require large volume data access, used AJAX to improve user experience.

If user action requires operations on the "Model", the Action Dispatcher will be used. It first dispatches the requested action to the domain Model and then creates the result page via the template engine. The following illustrates how easy it is to add an action using "actionHandler" decorator provided by our framework. This particular action is accessible to users with "admin" role or higher only. It first asks the EquestrianModel to perform the actual "addAsgn" action. Once done, it will display the "showroster" screen via the Screen Dispatcher.

```
@actionHandler('admin', next=showroster)
def addAsgn(req, **kwargs):
  EquestrianModel.addAsgn(req, **kwargs)
```

Using our frameworks, all the screens for the application were up within a week. Besides the screens, WMS also needed a Model and an AI rostering engine.

The "Model" represents the persistent domain objects, such as staffs, shifts and organizational structure. Since one of the project objectives is to ensure that data can easily be exchanged with external systems, we used a model that is based on open standards: HR-XML Release 2.5 (HR-XML 2007) – a library of XML schemas developed by the HR-XML Consortium, Inc. To create a Python-based object model from the XML schemas, we used a great tool created by Dave Kuhlman, called "generateDS" that automatically generates Python classes from XML schema documents (generateDS 2007). Using this approach, a standards-based domain object model was created instantly. GenerateDS also provides automatic import/export mechanisms to facilitate XML data exchanges that WMS needed.

Once the model was created, methods were added for persistency and to perform actions required for rostering. This may include invoking the "AI Engine" to perform the necessary AI processing. Like the rest of WMS, the "AI Engine" was also developed using Python using our AI framework and is described in the following Section.

Uses of AI Technology

There is, of course, a large body of AI research work related to staff rostering, especially for the problems of nurse rostering and airline crew scheduling (Barnhart et al. 2003, Butchers et al. 2001). For example, (Burke et al. 2004) provides an extensive survey on current state-of-the-art techniques in nurse rostering as well as general healthcare personnel scheduling. The survey summaries healthcare rostering/scheduling approaches into the following categories – linear and integer programming (Warner 1976, 1990), goal programming/multi-criteria (Jaszkiewicz 1997), declarative/constraint programming (Darmoni et al. 1995, Meisels et al. 1996, Meyer auf'm Hofe 1997), knowledge-based/expert systems (Smith et al. 1979), heuristics (Smith and Wiggins 1977), simulated annealing (Brusco and Jacobs 1995), tabu search (Dowsland and Thompson 2000, Burke et al. 1999), and genetic algorithms (Easton and Mansour 1993).

Even though there is a large body of rostering research, the number of actual deployed hospital rostering systems is still small. According to Burke et al., several areas need more work, for example, supporting multi-criteria reasoning to better reflect the multi-objective environment of the workplace; more flexible algorithms to allow for dynamic changes and cope with unforeseen events; easier to use with little or no parameters to adjust; better human/computer interactions; better leveraging on problem decomposition and problem-specific information, etc.

However, rostering for Olympic equestrian events is somewhat different from traditional rostering problems, such as nurse or airline crew rostering. Firstly, practically all the staff are part-time, secondly, shifts vary significantly from day to day, and thirdly, schedule changes are quite common to reflect the dynamic nature of the events. Unfortunately, volunteer rostering for previous Olympics have not been documented well and a search on the web did not retrieve any recent research paper in this area. On the other hand, the equestrian event problem share some similarities to rostering/scheduling for workers in fast-food chains or waiters/waitresses in restaurants. For example, (Fukuyama et al. 2006) used an interesting "rent and loan" metaphor to generate a fair schedule for part-time workers, such as part-time waiters/waitresses in restaurants. The "rent and loan" metaphor is an algorithm for the credit assignment problem to balanced roster assignment against personal shift requests.

For our WMS project, we modeled the rostering problem as a constraint-satisfaction problem (CSP); whereby a solution is found through the satisfaction of a number of constraints or criteria (Tsang 1993).

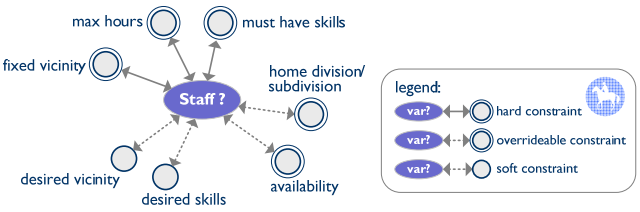

Figure 4. The structure of the rostering constraints.

The Rostering Constraints

Different from past approaches, our equestrian rostering constraints were coded and verified using an XML-based rule engine that is part of our AI platform. Rosters are then generated by repeated application of rules which generated labeling for the search algorithm (Apt 2001). The labels represent evaluation scores based on rule execution results and weighted by performance assessments taken during training or actual duty assignments.

Constraints in WMS are divided into 3 categories (see Figure 4) – hard, overrideable, and soft. Hard constraints are those that can never be violated. Overrideable constraints are those that can be overridden by human planners, possibly after confirmation with volunteer or other staff. Soft constraints represent preferences that can be violated if necessary.

The following are the key hard constraints considered by WMS:

- **max hours** – cannot assign person to more than the max no. hours per day a person is permitted to work
- **must have skills** – cannot assign person to duties that he/she does not have adequate skills to perform

- **fixed vicinity** – person to be assigned must be at (for convenience) or not at (for rotation) a given venue, possibly from a previous duty assignment

The following are the key overrideable constraints:

- **availability** – cannot assign duty to person if he/she is not available (can be overridden if confirmed with volunteer/staff involved)
- **home division/subdivision** – cannot assign volunteers/staffs from other division/subdivision (can be overridden if approved by staff in charge of the division/subdivision involved)

The following are the key soft constraints:

- **desired vicinity** – prefer assigning person who is already at or not at a particular desired venue, possibly from a previous duty assignment
- **desired skills** – prefer assigning a person who has the given desired skills

These constraints are coded as rules in RDF/XML and then converted to Python using a "rule compiler". Our AI platform's "rule compiler" support the generation of rule engines for 3 popular application platforms – .NET, Java EE and LAMP. For example, we used the same RDF/XML rule syntax for a .NET subway maintenance scheduling system (Chun 2004, 2005) and a Java-based e-Government form processing system (Chun 2007). We call this type of AI development approach as a "non-intrusive" XML-based approach where all knowledge and configurations are coded using only XML documents without affecting objects in the existing domain model. Automatic code-generation techniques were then used to dynamically generate the actual engines. This greatly shortens development time and minimizes potential coding errors (Tabet et al. 2000).

The WMS Rostering Algorithm

Figure 5 shows the process flow of the WMS rostering algorithm. Each division/subdivision defines a set of shifts/duties that need to be filled each day. For each of these manpower needs, the rostering algorithm then uses the rule engine to determine which staffs satisfy the rostering constraints, such as available, skills and vicinity. The result is a pool of volunteers/staffs that satisfies all the hard constraints. In addition, associated with each volunteer/staff is a score to indicate the "quality" of the match. The quality depends on weightings from violations of soft constraints and training assessment scores as well as possible performance evaluation scores.

The pool is sorted according to this score. If adequate numbers of staffs are available, they will be assigned. Human intervention will be needed if there are not enough staffs that meet all the constraints. The human planner will need to see if any of the overrideable constraints can be overridden. For example, calling volunteers up to check if they can make themselves available or borrowing people for a different division/subdivision. If this fails, the planner will need to see if additional part-time staff can be hired.

Innovations

Besides the non-intrusive XML-based AI rule coding, the AI algorithm is innovative in that it caters to a variety of different staff rostering and scheduling needs at the same time. With 9 divisions and 38 subdivisions, the rostering needs can be quite different. Some divisions/subdivisions with consistent workloads and staff availabilities may decide to roster using traditional shifts while others may break shifts into smaller units to better match volunteer availabilities. Other divisions/subdivisions may decide to roster and perform job assignments at the same time to ensure rotation and vicinity constraints were met. In addition, shift start times and durations may be different from day to day. Our rostering algorithm supports a very flexible shift definition that allows for these variances together with mechanisms to combine shorter unit shifts and job assignments to form full shifts.

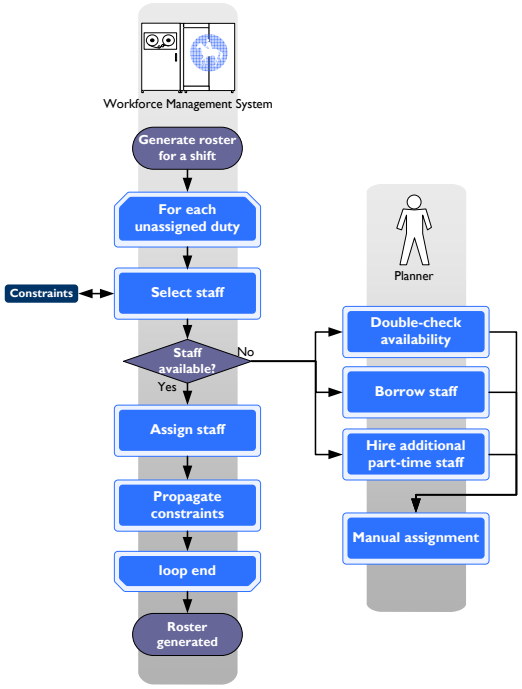

Figure 5. Process flow diagram for AI rostering

Application Use and Payoff

The organizers began using WMS in late June 2007, initially for data update and then for rostering and messaging; the system was fully deployed by mid-July 2007. The Good Luck Beijing equestrian games were held in mid-August 2007. Roughly a thousand volunteers/staffs used the WMS daily as the main form of communication for rosters, duties, schedules, instructions, messages, notices, etc.

Application Payoff

The key benefits from using WMS were:

- **Ensured games ran smoothly** – Using WMS, the event organizer was able to ensure that each and every games-related activity – before, during, and after the games – had adequate level of staffing support. It also ensured that all assigned staff had adequate skills with the required training/qualifications. Without WMS, it would have taken a tremendous amount of effort and manpower to create rosters for a thousand people manually; not to mention the time needed to notify each of the staff of their rosters and updates each day.
- **Provided agility –** Because WMS allowed the organizer to quickly add or change shifts and duties, to quickly find available and qualified staff to fill these duties, and to easily notify them of changes, WMS provided great agility to the event organizer in handling unexpected workloads, changes, or delays. Without WMS, the planners would have had to sift through stacks of paperwork to figure out who would be available, if they had any conflicting assignments, and whether they had adequate training/ qualifications.
- **Kept everyone in sync** – Because WMS acted as the central focal point for all information related to rosters, duties, schedules, updates, messages, etc., all volunteers/staffs acquired a habit of checking with WMS daily to get any last minute updates/changes. The use of WMS ensured the entire workforce was kept in sync at all times. Without WMS, the organizer would have had to hire a team of operators to call up people whenever there were changes.

Results

The main objective for using WMS was to ensure that the Good Luck Beijing equestrian games run smoothly and to get the International Olympic Committee's approval on Hong Kong readiness preparation. In terms of meeting these objectives, WMS served its intended purpose very well. Both the Hong Kong and the International Olympic Committees were very satisfied with the overall performance of the Good Luck Beijing games (HKDigest 2007). The Director of Corporate Administration, Dr. Horace Yuen, complimented CityU on our efforts in creating the WMS: "*[CityU] was very professional in understanding user requirements… readily available Web-based platform… shortened development time.*"

Application Development and Deployment

The design and development of WMS began around mid-May 2007 with the first release by end of June 2007. The project was performed by the CityU Professional Services Limited, a non-profit subsidiary of the City University of Hong Kong.

Since the WMS project was not really planned ahead of time, getting resources together at the last minute was a bit of a challenge. Fortunately, we were able to get a good size team together in time. The team consists of a project leader from the Equestrian Events who coordinated all the requirements from all the divisions. Several key users/planners from the Equestrian Events provided input as well as feedback on usability testing. The AI development was led by Dr. Andy Chun, who also performed the design and was the lead coder, a system administrator who handled the systems configuration and routine maintenance work, and software engineers who performed testing.

The total development time was roughly 6 weeks. Prior to project commencement, we spent 2 days to design and construct a click-through prototype using our FOA Application Framework and Template Engine. This gave the Equestrian Events a much better understanding of what the final system would feel like and higher confidence in going ahead with the development. The first two weeks after project commencement were used to define the requirements and to perform data preparation, which included creating the HR-XML compliant domain model. Once the domain model was in place, it only took roughly a week to come up with a detailed second prototype. Rostering constraints were implemented in the fourth week, when we hooked up our AI engine and rostering algorithm. The last two weeks were then spent on refining the user interface and interactivity as well as thoroughly testing the application.

Since the system was built on top of our existing Web frameworks and AI libraries, the main design and development tasks were to ensure the AI constraints were properly coded and to match the Web-based process flow with the operational needs to support the equestrian events.

Deployment

Despite the short development cycle, the deployment was still broken down in 3 stages to ensure the project rolled out smoothly. At end of June, it was first "soft" launched to all volunteers/staff with restricted features. They were allowed to view and modify their own profile, availability and other information, such as uniform sizes. This early release gave everyone more time to familiarize themselves with WMS and to ensure that the data in our database was up to date.

The second stage was a week after, when all the planners began to use the system to create the rosters for their respective divisions/subdivisions. This allowed us to evaluate the performance of the "admin" features and fine-tune different usage patterns between different departments. The third stage is the official launch in mid-July when all volunteers/staff began to check their actual rosters and job assignments. This is also when the Equestrian Events started to use WMS for daily Web-based communication with and announcements to all the volunteers/staffs.

Evaluation

The system was highly tested and evaluated because it is to be widely deployed and used by a large number of people. The rollout in stages allowed us to assess and evaluate

different features and performance in a more manageable pace. Comparing with other approaches and alternatives that were considered by the Equestrian Events, the system performed very well.

Had our system been not available, the main alternative was to manually schedule and manage staff using individual spreadsheets, one for each division/subdivision. In fact, the planners were doing exactly just that before our system was deployed. As one can easily imagine, there were many problems with the manual approach. Firstly, it was hard to determine which spreadsheets were most up to date; different versions of overlapping data were floating around. A large pool of staff would also be needed to manually call each and every volunteer/staff to notify them of their schedules/assignments daily and whenever there were changes.

Our system brought many benefits to the management of staffs for the event. By centralizing data in one place, it firstly ensured data consistency and avoided potential confusion in schedules and assignments. Secondly, it represented a tremendous savings in manpower. Instead of a team of staff dedicated to calling up or answering calls from volunteers/staffs, all notifications were automated and online. Time needed to produce rosters was also greatly reduced. Instead of tediously sifting through sheets and sheets of spreadsheet data to find appropriate staff to assign, the AI rostering algorithm generates assignments within a fraction of a second.

Maintenance

Since the WMS was offered in a SaaS (software as a service) model (Wikipedia 2007), all system-level maintenance works were performed by CityU. Many of the traditional maintenance tasks, such as exporting data for backup, are provided in "self-service" mode, i.e. Web-based downloads, allowing the event organizer total control over their own data. For the AI knowledge part, no real maintenance was needed since the rules and constraints governing staff rostering were relatively static and did not change. However, around the clock system-level support was provided during the crucial days of the actual equestrian games.

Application-level maintenance was performed by the equestrian event planners. These tasks include adding/modifying volunteer/staff profiles, changing availabilities, defining jobs/tasks, re-assigning jobs/tasks if a staff/volunteer becomes unavailable, etc. These are easily done using Web-based forms. Real-time attendance records, however, are maintained by a separate system.

Conclusion

This paper described how we used modern Web 2.0 architecture together with reusable application frameworks and an AI platform to create a workforce management system to support Olympics equestrian event preparation within a very short time. The resulting Workforce Management System was used daily by all volunteers/staffs during the 2007 Good Luck Beijing equestrian games in Hong Kong to retrieve messages, instructions, rosters, and duty assignments. AI ensured that the best use of available resources was made. It also provided the event organizer with agility in handling last minute changes and unexpected delays. With the help of WMS, the HK organizers passed the International Olympic Committee's preparation requirements in flying colors.

References

Apt, K.R., Monfroy, E. 2001, "Constraint programming viewed as rule-based programming," *Theory and Practice of Logic Programming*, v.1 n.6, p.713-750, November 2001.

Barnhart, C., Cohn, A.M., Johnson, E.L., Klabjan, D., Nemhauser, G.L., and Vance, P.H. 2003. "Airline Crew Scheduling," In *Handbook of Transportation Science*, 2nd ed., Kluwer's International Series.

Beijing2008. 2007a. The Beijing 2008 Olympic Games Web Site, "Hong Kong passes Olympic equestrian test," 14 Aug 2007. Available at en.beijing2008.cn/goodluckbj/equestrian/s214126194/n214129339.shtml

Beijing2008. 2007b. The Beijing 2008 Olympic Games Web Site, "Preparation for 'Good Luck Beijing' events under way," 21 July 2007. Available at en.beijing2008.cn/61/66/article214026661.shtml

Beijing2008. 2007c. The Beijing 2008 Olympic Games Web Site, "HK Well-prepared for Olympics Equestrian Trial Event for 'Good Luck Beijing' events under way," 9 August 2007. Available at en.beijing2008.cn/news/sports/headlines/equestrian/n214123959.shtml

BTMBeijing. 2007. The Bejing This Month Web Site, "Ready for Testing," 31 July 2007. Available at www.btmbeijing.com/contents/en/btm/2007-08/olympics/testing

Brusco, M. J. and Jacobs, L.W. 1995. "Cost analysis of alternative formulations for personnel scheduling in continuously operating organisations," *European Journal of Operational Research*, 86, pp.249–261.

Burbeck, S. 1992. "Application Programming in Smalltalk-80: How to use Model-View-Controller (MVC)." University of Illinois in Urbana-Champaign (UIUC) Smalltalk Archive. Available at st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html

Burke, E. K., Causmaecker, P. De, and Vanden Berghe, G. 1999. "A hybrid tabu search algorithm for the Nurse rostering problem," in B. McKay et al. (eds.), *Simulated Evolution and Learning*, Springer, Lecture Notes in Artificial Intelligence, vol. 1585, Springer, pp. 187–194.

Burke, E.K., De Causmaecker, P., Vanden Berghe, G., and Van Landeghem, H. 2004. The State of the Art of Nurse Rostering,

Journal of Scheduling, Vol. 7 Issue 6, Nov/Dec 2004.

Butchers, E.R., Day, P.R., Goldie, A.P., Miller, S., Meyer, J.A., Ryan, D.M., Scott, A.C., and Wallace, C.A. 2001. "Optimized Crew Scheduling at Air New Zealand," *Interfaces*, Vol. 31, No. 1, Jan-Feb 2001, pp. 30-56.

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*, pp. 786-793, Orlando, July, 1999.

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*, Austin, August, 2000.

Chun, H.W. and Yeung, Wai Ming, 2004. Rule-based Approach to the Validation of Subway Engineering Work Allocation Plans, In the *Proceeding of the International Conference on Computing, Communications and Control Technologies*, August 14-17, 2004, Austin, Texas, USA.

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*, Pittsburgh, July, 2005.

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*, Vancouver, July, 2007.

COC. 2007a. The Chinese Olympic Committee Web Site, "Preparation for Olympic Games Moving Forward as Planned." Available at en.olympic.cn/08beijing/bocog/2006-08-09/903952.html

COC. 2007b. The Chinese Olympic Committee Web Site, "BOCOG Sums Up Experience in Volunteer Services," 31 August 2007. Available at en.olympic.cn/08beijing/bocog/2007-08-31/1246192.html

COC. 2007c. The Chinese Olympic Committee Web Site, "BOCOG: Good Luck Beijing events go on smoothly," 25 August 2007. Available at en.olympic.cn/08beijing/bocog/2007-08-25/1241284.html

Darmoni, S. J., Fajner, A., Mahe, N., Leforestier, A., Stelian, O., Vondracek, M., and Baldenweck, M. 1995. "Horoplan: Computer assisted nurse scheduling using constraint based programming," *Journal of the Society for Health Systems*, 5, pp.41–54.

Dowsland, K. A. and Thompson, J.M. 2000. "Solving a nurse scheduling problem with knapsacks, networks and tabu search," *Journal of the Operational Research Society*, 51, pp.825–833.

Easton, F. and Mansour, N. 1993. "A distributed genetic algorithm for employee staffing and scheduling problems," in *International Conference on Genetic Algorithms*, San Mateo, 1993, pp.360–367.

Fukuyama, M. Shimakage, M. and Hazeyama, A. 2006. A Scheduling Method Based on the Rent and Loan Information, *IEICE Transactions on Information and Systems*, Vol.E89-D, No.2, pp.798-805.

generateDS. 2007. The GenerateDS Web Site. Available at www.rexx.com/~dkuhlman/generateDS.html

HKDigest. 2007. The Hong Kong Digest Web Site, "IOC Gives Hong Kong Vote of Confidence," Available at www.hketousa.gov.hk/ny/e-newsletter/07july/Equestrian.htm

HKEquestrian. 2007a. The Hong Kong Equestrian Events Web Site. "Olympic Equestrian – Events." Available at www.equestrian2008.org/eng/olympic_e.aspx

HR-XML. 2007. The HR-XML 2.5 Release XML schemas. Available at www.hr-xml.org/hr-xml/wms/hr-xml-1-org/index.php?id={E00DA03B685A0DD18FB6A08AF0923DE0|139|2}

Jaszkiewicz, A. 1997. "A metaheuristic approach to multiple objective nurse scheduling," *Foundations of Computing and Decision Sciences*, 22(3), pp.169–184.

Meisels, A., Gudes, E., and Solotorevski, G. 1996. "Employee timetabling, constraint networks and knowledge-based rules: A mixed approach," in E. K. Burke and P. Ross (eds.), *Practice and Theory of Automated Timetabling*, First International Conference, Edinburgh, Springer, Lecture Notes in Computer Science, Vol. 1153, pp.93–105.

Meyer auf'm Hofe, H. 1997. ConPlan/SIEDAplan: Personnel assignment as a problem of hierarchical constraint satisfaction, In *Proceedings of the Third International Conference on the Practical Application of Constraint Technology*, London, pp. 257–271.

Smith, L. D., Bird, D. and Wiggins, A. 1979. "A computerised system to schedule nurses that recognises staff preferences," *Hospital & Health Service Administration*, pp.19–35.

Smith, L. D. and Wiggins, A. 1977. "A computer-based Nurse scheduling system," *Computers and Operations Research*, 4(3), pp.195–212.

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

Warner, M. 1976. "Scheduling Nursing personnel according to Nursing preference: A mathematical programming approach," *Operations Research*, 24, pp.842–856.

Warner, M., Keller, B. and Martel, S. 1990. "Automated Nurse scheduling," *Journal of the Society for Health Systems*, 2(2). pp.66–80.

WTOPnews. 2007. The WTOPnews.com Web Site, "Equestrian Event Gets Trial in Hong Kong," 2 Aug 2007. Available at www.wtopnews.com/?nid=393&sid=1207330

Wikipedia. 2007. Wikipedia Web Site, "Software as a Service," 5 Dec 2007. Available at http://en.wikipedia.org/wiki/Software_as_a_Service