2009 Ninth IEEE International Conference on Bioinformatics and Bioengineering

Wedjat: A Mobile Phone Based Medicine In-take Reminder and Monitor

Mei-Ying Wang, John K. Zao
Computer Science Department
National Chiao Tung University
Hsinchu, Taiwan, Republic of China
chesnut.cs96g@nctu.edu.tw, jkzao@cs.nctu.edu.tw

P.H. Tsai, J.W.S. Liu
Institute of Information Science
Academia Sinica
Taipei, Taiwan, Republic of China
peipei@iis.sinica.edu.tw, janeliu@iis.sinics.edu.tw

Abstract— **Out-patient medication administration has been identified as the most error-prone procedure in modern health-care. Under or over doses due to erratic in-takes, drug-drug or drug-food interactions caused by un-reconciled prescriptions and the absence of in-take enforcement and monitoring mechanisms have caused medication errors to become the common cases of all medical errors. Most medication administration errors were made when patients bought different prescribed and over-the-counter medicines from several drug stores and use them at home without little or no guidance. Elderly or chronically ill patients are particularly susceptible to these mistakes.**

In this paper, we introduce *Wedjat*, a smart phone application designed to help patients avoiding these mistakes. *Wedjat* can remind its users to take the correct medicines on time and record the in-take schedules for later review by healthcare professionals. *Wedjat* has two distinguished features: (1) it can alert the patients about potential drug-drug/drug-food interactions and plan a proper in-take schedule to avoid these interactions; (2) it can revise the in-take schedule automatically when a dose was missed. In both cases, the software always tries to produce the simplest schedule with least number of in-takes. *Wedjat* is equipped with user friendly interfaces to help its users to recognize the proper medicines and obtain the correct instructions of taking these drugs. It can maintain the medicine in-take records on board, synchronize them with a database on a host machine or upload them onto a Personal Heath Record (PHR) system. A proof-of-concept prototype of *Wedjat* has been implemented on Window Mobile platform and will be migrated onto Android for Google Phones. This paper introduces the system concept and design principles of *Wedjat* with emphasis on its medication scheduling algorithms and the modular implementation of mobile computing application.

***Keywords*— telemonitoring, medication error prevention, mobile computing, real-time scheduling.**

I. INTRODUCTION

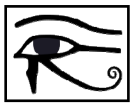
According to a landmark study on medical errors conducted by the US Institute of Medicine in 1999 [1], *medication errors (ME)* and *adverse drug reactions (ADR)* are the most common cases among all medical errors. These *adverse drug events (ADE)* incurred significant tolls in terms of patient fatality, financial costs (including additional medical expenses, lost income and productivity), and damages to the reputation and morale of healthcare professionals. Most of these errors are nonetheless preventable [2]. One study found 530,000 preventable ADEs among Medicare out-patients each year.

Although errors can occur in every step of medication process during medicine procurement, prescription, dispensing and administration, they happen most frequently in the *prescription* and *administration* stages. In the past decade, increasingly wide-spread use of computerized physician order entry (CPOE) [3], clinical decision support systems (CDSS) [4] and electronic medical records (EMR) [5] along with better procedures to dispense medicine has helped to eliminate a large proportion (up to 80%) of prescription errors, which account for half of all medication errors. In comparison, little progress has been made in the prevention of administration errors, which are caused by improper and failed use of prescribed medicine. Consequently, medicine administration errors have become the prevalent cause of ADEs. They accounted for 25%–40% of all medication errors and were the main reason for admission of elderly into nursing homes [6].

Out-patient medication administration has been identified as the most error prone procedure amidst the entire medication process. Most of these errors were made when patients bought different prescribed and over-the-counter (OTC) medicines from several drug stores and use them at home without little or no guidance. Common causes of these errors include: (1) *irregular medicine in-takes* due to the patient's busy or erratic lifestyles, (2) *complicated in-take schedules* due to many medicines and doses taken by the patient, (3) *adverse drug reactions* caused by un-reconciled prescriptions obtained from different sources, (4) *lack of knowledge* about proper use of medicines, (5) *lack of consultation* with healthcare providers when confusion arises and (6) *lack of monitoring mechanisms* to keep track of patient's medicine in-take. Recently, telemedicine, especially *telemonitoring* techniques, has been investigated as a cost-effective approach to control quality of care (QoC) in out-patient medication administration [7,8,9,10]. By sending in-take reminders to the patient (even producing the proper medicine from a medicine dispenser) and then recording patient's responses, Health Maintenance Organizations (HMO) hope to reducing cost of service while improving quality of care. Communication between HMOs and patients is established through wired or wireless Internet connections. Although these efforts represent progress in the right direction, the medicine dispensers thus made are bulky, expensive and prone to dispensing errors. A handy alternative solution can be provided by installing a *medication reminder and monitor* on a smart mobile phone

978-0-7695-3656-9/09 $25.00 © 2009 IEEE
DOI 10.1109/BIBE.2009.60

and then using it along with a traditional mechanical "pill box". Such a solution will be cheaper (excluding the incurred cost of the smart phone) and may result in deeper penetration into the consumer market.



In this paper, we introduce *Wedjat*, a smart phone application designed to help patients to avoid medicine administration errors. The software is named after the "Eye of Egyptian God Horus" [insert] from which the prescription symbol R_x was derived. *Wedjat* can perform the following three primary functions:

1. *Issue medicine in-take reminders* — *Wedjat* will issue an alert approximately 5 – 15 minutes (preset by user) before the scheduled time to take certain medicine(s). The alert will be issued repetitively until it is cancelled by the user. Scheduling of in-take alerts is performed by a real-time process/resource scheduling algorithm [Section IV] that can satisfy time constraints according to medicine in-take directions and drug-drug/drug-food interactions. This function is integrated with the calendar and planner applications installed on most smart phones.
2. *Provide medicine identification and in-take directions* — *Wedjat* has a built-in database containing crucial information about the medicines (including their photo images, in-take directions and precautions) and the healthcare providers (including physicians, pharmacists and HMOs) relevant to its user. All these data can be retrieved with the touch of a button while *Wedjat* is in use.
3. *Maintain medicine in-take records* — *Wedjat* will record the time at which its user cancels an in-take alert and regard that at the time that specific medicine(s) was taken. These medicine in-take records can be stored on board, synchronize with the database on a host machine and/or uploaded onto a Personal Heath Record (PHR) system.

Wedjat has two distinguished features: (1) it can alert the patients about potential drug-drug/drug-food interactions and plan a proper in-take schedule to avoid these interactions; (2) it can revise the in-take schedule automatically when a dose was missed. In both cases, the software always tries to produce the simplest schedule with least number of in-takes. A proof-of-concept prototype of *Wedjat* has been implemented on the Window Mobile platform and will be migrated onto Android for Google Phones.

The rest of this paper presents the design concepts of *Wedjat* with emphasis on its medication scheduling algorithms and modular implementation. An overview of the *Wedjat* system is given in Section II. It is followed by discussions of the formal specification of the time constraints [Section III] and the two algorithms used in medicine in-take scheduling [Section IV]. Actual implementation of the mobile computing application is discussed in Section V. Related work is compared in Section before accomplishments and future work are summarized in a brief conclusion.

II. System Overview

A. System Architecture

Figure 1 shows the system architecture overview including mobile phone with Wedjat, Indivo including Indivo Server, Indivo Application Interface and Encrypted Data Store, Hospital, Pharmacy and other Medical Services. The mobile phone is the end point of the system and communicates with user directly. Wedjat on mobile phone uses wireless to connect to Indivo Server and then download medicine scheduling specification as the input of Wedjat. Wedjat schedules the medicines in-take according to MSS and writes the whole in-take schedule to Calendar application on and notes down user's medicines in-take records and uploads the in-take records to Indivo. If doctors or pharmacists change the content of prescription such as increase dosage of some medicines, the Wedjat can download the latest version of MSS from Indivo and reschedule. Indivo is a distributed, web-based, personally controlled health records (PCHR) system for information exchange and communication. It follows the open standard PCHR and provides API is accessible on the Internet. The patient can easily take his own medical treatment records. Pharmacies and hospitals are the data provider of the system. They upload medical records to Indivo and view or down load medical records only when they have user's permission. So that they can really know and control the patient's condition and improve the medical service quality.

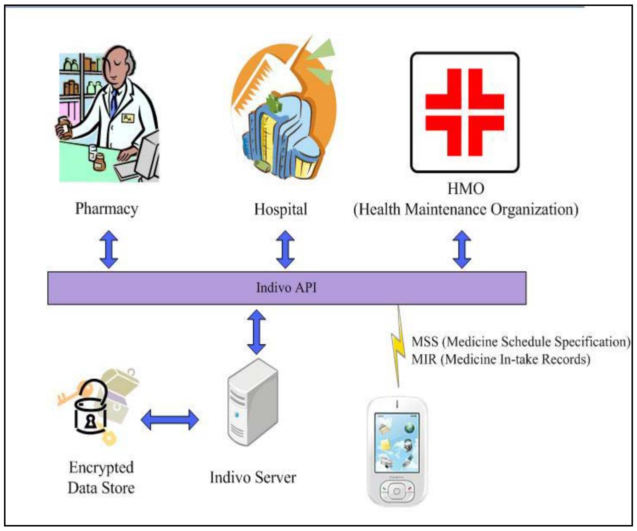

Pharmacy
Hospital
HMO
(Health Maintenance Organization)
Indivo API
MSS (Medicine Schedule Specification)
MIR (Medicine In-take Records)
Encrypted
Data Store
Indivo Server

Figure 1: *Wedjat* System Architecture

B. Operation Scenarios

After Wedjat writes medicines in-take schedule to Calendar, it set the every medicine in-take as a calendar event. The

Calendar will pop out a message 15 minutes before medicine in-take time to reminder user. If user clicks OK, Wedjat will take down the records. When user delays or miss some doses, Wedjat checks the schedule presently is feasible or not and readjust the schedule and rewrites to Calendar. If feasible schedule is impossible, Wedjat sends warning message to user to tell him (her) the situation is urgent or not. If urgent, it suggest user to make contact with doctor immediately.

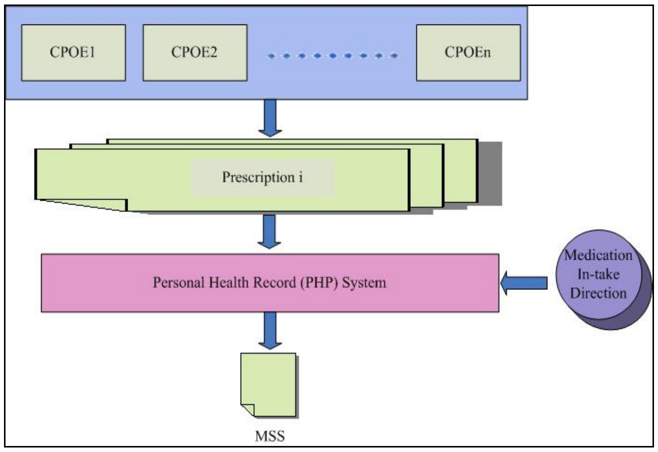

CPOE1
CPOE2
CPOEn
Prescription i
Personal Health Record (PHP) System
Medication In-take Direction
MSS

Figure 1: MSS Production Process

III. Medication Prescription and Scheduling Specification

A. *Medication Prescription Compilation*

Figure 1 shows the data flow in the production of Medication Scheduling Specification (MSS). A patient may be taken care by several physicians, each of whom issues prescription using a different CPOE system. Tools provided by Personal Health Record Systems such as Indivo may be used to transcribe medicine prescriptions and in-take directions to standard format and then merged to produce a medication scheduling specification (MSS) for a specific patient. The MSS XML file will be sent to *Wedjat* via RSS subscription.

B. *Medication Scheduling Specification*

Table 1 shows the scheduling specification of a medicine. Such a specification consists of three parts: Prescription Parameters (PP), Dosage Parameters (DP) and Interaction Parameters (IP).

Prescription Parameters (PP)	
Medicine Identifier	M
Medicine Dose	g
Medicine Form	Capsule/Tablet/…
Medicine Amount	n
Therapy Duration	T
Dosage Parameters (DP)	
Nominal Minimum and Maximum Dose	[dmin, dmax]
Nominal Minimum and Maximum Separations	[nsmin, nsmax]
Maximum Intake B over interval R	(B, R)
Minimum Intake L over interval P	(L, P)
Absolute Minimum and Maximum Dose	[Dmin, Dmax]
Absolute Minimum and Maximum Separations	[asmin, asmax]
Interaction Parameters (IP) <List>	
Interferer Identifier	N
Minimum Separation from M to N	minToInterferer
Minimum Separation from N to M	minFrInterferer

Table 1: Sample Medication Scheduling Specification

1) *Prescription Parameters (PP)*

The first part gives information *Wedjat* needs to know about the medicine including its name or identifier M and the *duration that* the patient shall take this medicine. The medicine comes in dose g *(*medicine dose); dose size parameters of the medicine are given in terms of integer multiples of g. The medicine form indicates the unit of dosage, such as capsule, tablet…etc. The total amount of medicine patient need to in-take is medicine amount. The part also provides other relevant attributes such as a picture image of the medicine for verification purpose. The Horus uses the same time resolution for all medicines. All separation parameters expressed are in terms of multiples of Horus time resolution. We use one hour hereafter unless stated otherwise.

2) *Dosage Parameters (DP)*

The dosage parameters part specifies constraints on dose size and separation (i.e., the length of time interval between any two consecutive doses) for scheduling the medicine when the medicine is taken alone. Take the direction of Advil for example: a part of it reads “Take 1 gel caplet every 4 to 6 hours. If pain or fever does not respond to 1 capsule, 2 capsules may be used.” So, its nominal dose size and separation ranges are [1,2] and [4,6], respectively.

Supply Rate (B, R) of M: It indicates that the *intake* (i.e., the total size of all doses) within any time interval of length R must be no more than B. For example, the supply rate of Advil is (6,24) because its direction also says “Do not exceed 6 gel capsules in 24 hours.”

Demand Rate (L, P) of M: It indicates that the intake within any interval of length P must be at least L. Many medicines (e.g., antibiotic and insulin) have demand rate constraint to ensure that at least the minimum required amount is at work.

The DP part may also include *absolute dose size range* `[Dmin, Dmax]` and *absolute separation range* `[asmin, asmax]`. These constraints are hard. By making the ranges wider than the corresponding nominal ranges, the direction allows some flexibility in scheduling.

3) Interaction Parameters (IP)

We refer to a medicine or food that interacts with *M* to the extent as to require some changes in how *M* is to be administered as an *interferer N* of *M*. The IP section of *M* contains an entry for each of its interferers.

The entry of an interferer *N* may also define additional separation constraints, each of which specifies a required time separations between each dose of *M* and any dose of the interferer *N*. Table 1 lists only the *minimum separation* `minToInterferer` *from the medicine to interferer* for each dose *M* scheduled before any dose of *N* and the *minimum separation* `minFrInterferer` *from the interferer to the medicine* for each dose of *N* scheduled before some dose of *M*. Take Fosamax as an example. This medicine for prevention and treatment of brittle bone decease must be taken on empty stomach, and the user should not take anything within 30 minutes after taking the medicine. Hence the minimum separation parameters to and from any interferer of Fosamax are half an hour and 6 hours, respectively. As we will see in Section 7, the required separations between doses of interferers make scheduling more difficult.

Table 1 leaves off additional constraints due to medicine interaction, including precedence constraints that restrict the order in which doses of some interacting medicines are taken, and maximum separation constraints that ensure interferers are taken sufficiently close together. These constraints are discussed and illustrated in [11].

IV. MEDICINE IN-TAKE SCHEDULING ALGORITHMS

This section provides an overview of the algorithms for scheduling multiple medicines [13]. The algorithms work with fixed dose sizes. As stated earlier, that a valid dose size exists has already been assured when the user's MSS was generated. By first choosing a valid dose size for each medicine, the scheduler then focuses on finding times for individual doses to meet all intra-medicine and inter-medicine separation constraints.

A. Scheduling Models

1) Resource Model

The design of the scheduler is based on the resource model [12,13] that uses a virtual processor *PM* and a virtual resource *RM* for each medicine *M* to keep track of when the user is available to take the medicine. When computing a schedule, the scheduler treats each dose of each medicine *M* as if it is a job on processor *PM* and the sequence of doses of the medicine as a task *M*. A job starts when the corresponding dose should be retrieved by the user. A schedule for the medicine is a list of the time instants at which jobs of task *M* start.

2) Resource Allocation Rules

The scheduler maintains correct separations between doses of *M* by scheduling each of the corresponding jobs non-preemptively on *PM* for this amount of time whenever possible, but it may schedule the job for a smaller amount time in the range from the absolute minimum separation `asmin(M)` of *M* to `nsmin(M)`. In addition, the maximum absolute separation between consecutive doses of *M* is enforced by imposing a relative deadline for each of the jobs.

For each medicine *M* that has interferers, the scheduler uses the virtual resource *RM* and resources of the interferers to help it maintain inter-medicine separations. Simply put, a job of *M* can start at a time *t* only when the resource *RM* is free at *t*. The scheduler allocates the resource *RM* to each job of *N* for `minFrInterferer` units of time each time when it schedules a job on *PN*. Thus, each job of the interferer *blocks* jobs of *M* from starting for this amount of time. The *worst-case blocking time* of the medicine *M* is the maximum over all of its interferers of the minimum separations from interferer to *M*.

3) Priority Schemes

Both OMAT and ODAT algorithms operate based on priorities. One of the better priority schemes is the *Most Victimized First* (*MVF*) scheme which gives priorities to tasks based on their worst case blocking times; the longer the worst case blocking time, the higher the priority.

Other priority schemes based on separation and interference characteristics of the medicines include the *Most Interferers First* (*MIF*) scheme and the *Shortest Separation Difference First* (*SSDF*) schemes. They give higher priorities to tasks corresponding to medicines with larger numbers of interferers or larger differences between the maximum and minimum nominal separations, respectively. We also experimented with the classical real-time priority schemes *Rate Monotonic* (*RM*) and *Earliest Deadline First* (*EDF*) schemes. The former gives higher priorities to tasks with shorter periods. The latter gives priorities to jobs based on their absolute deadlines; the earlier the deadline, the higher the priority. Clearly, EDF scheme is suitable for ODAT algorithms only.

B. Scheduling Algorithms

Figure 10 provides the pseudo-code description of basic OMAT and ODAT algorithms. The inputs are MSS of all medicines and `PrioritySchemes` used by the algorithms. The Boolean output variable `feasible` indicates whether the algorithm succeeded in finding a feasible schedule when it terminates. If it succeeded (i.e., `feasible` is TRUE), the elements of the `feasibleSchedule[]` array point to the schedules —i.e., the lists of job start time— of all medicine in the MSS.

1) One-Medicine-at-A-time (OMAT)

After creating instances of data structures described above, the scheduler considers medicines in non-increasing priority order: It schedules individual dose of each medicine *M* as a separate job, starting from the first job until it either fails to find an available (start) time for some job of *M* before the duration T of the medicine or has successfully generated a list of start times for all the jobs within the duration. An available time for a job is an instant between the release time and deadline of the job at which both *PM* and *RM* is free. The search for the earliest of such instants and the bookkeeping chores of the scheduler are described by Table2. In the former case, the scheduler returns immediately with feasible set to FALSE. In the latter case, it sets the element in `feasibleSchedule[]` for the medicine to point to the newly generated list and then move on to work on the medicine next in priority order.

```
Input:
{ List MedicationScheduleSpec MSS[];
  List MedicineIntakePriority MIP[]; }
Output:
{ Boolean  feasible;
  List MedicineIntakeSchedule MIS[]; }
Procedure:
{ For every Medicine in MSS[] {
    Create JobModel, ResourceModel, ScheduleList;
    Assign priority to MSS[] based on MIP[]
    Sort MSS[] by priority order
  }
  For every Medicine in MSS[]{
    lastestStartTime = 0;
    currentMedicineFeasible = TRUE;
    Do while (currentMedicineFeasible == TRUE)
    {
      Call FindAvailableTime()
        to get earliest available instant x;
        If (x is found)
          If (x >= T of Mi) break;
            Insert x into ScheduleList of Mi;
          For (x <= k < x+ nsmin)
            Set process[k] of Mi to 1
          For every interferer N of Mi in MSS[]
            For(k >=x - minFrInterferer &&
              k < x + minToInterferer of Ni) {
              Set resource[k] of Ni to 1
              latestStartTime = x; }
        else x is not found {
          currentMedicineFeasible = FALSE;
          feasible = FALSE; }
    }
    If (feasible == TRUE)
      Insert ScheduleList of Mi to MIS[];
  }
  Return feasible and MIS[];
}
```

Table2: One-Medicine-At-a-Time

Basic OMAT algorithms tend to schedule doses of higher priority medicines close together, leaving little or no time for doses of lower priority medicines. A simple enhancement is to schedule doses of medicines as close as to their respective deadlines as possible. We call algorithms with this enhancement *advanced OMAT* algorithms.

2) One-Dose-at-A-Time (ODAT)

Table 3 shows that the scheduler does the same work when it schedules individual jobs regardless of the algorithm it uses. An ODAT scheduler assigns priorities to jobs according to `PrioritySchemes` when the jobs are released. It uses the array `releaseTimesCurrentJobs[]` to keep track when the current job of each medicine is released and ready to be scheduled. The schedule of a medicine *M* is *complete* when the possible start time of the job of *M* currently being scheduled is later than the minimum duration of *M*. The scheduler continues to schedule jobs as they are released in priority order until either it fails to find a possible start time for the job currently being scheduled or the schedules of all medicines in the given MSS are complete.

```
Input:
{
  List MedicationScheduleSpec MSS[];
  List MedicineIntakePriority MIP[];
                        }
Output:
{
  Boolean  feasible;
  List MedicineIntakeSchedule MIS[];
}
Procedure:
{ For every Medicine in MSS[]
  { Create JobModel, ResourceModel, ScheduleList;
    ScheduleComplete = 0;
    ReleaseTimeCurrentJobs[numberMedicines] = {0};
  }
  While(feasible == TRUE &&
    scheduleComplete < numberMedicines)
  { Find job j with
      earliest release time and highest priority
      from JobModel among MSS[];
    Do
    { Call FindAvailableTime()
        to get earliest available instant x;
      If (x is found)
        If (x >= T of Mi) scheduleComplete++
          break;
        Insert x into ScheduleList of Mi;
        For (x <= k < x+ nsmin )
          Set process[k] of Mi to 1
        For every interferer N of Mi in MSS[]
          For(k >= x -minFrInterferer &&
            K < x + minToInterferer of Ni)
            Set resource[k] of Ni to 1
        releaseTime of j of Mi = x + nsmin;
      else x is not found
        feasible = FALSE;}
    }
  }
  Return feasible and MIS[];
}
```

Table 3: One-Dose-At-a-Time

V. Implementation of In-take Reminder and Monitor

This section documents the implementation details of Wedjat including its platform, software design and user interfaces.

A. System Platform

Wedjat is developed on .net framework with windows mobile 6.0 SDK and integrate with Calendar application through the Calendar application interface released by Microsoft. After *Wedjat* schedule all medicines, the system writes the medicine in-take schedule to Calendar application.

As shown in Figure 2, *Wedjat* inputs are user preferences and MSS of all medicines including medicine directions and user prescriptions. *Wedjat* download MSS of all medicines taken by its user from an Indivo personal health record (PHR) server. In addition, user set his/her preferences such as sleep time and busy time using the *Wedjat* graphic user interface. If feasible, the scheduler will avoid scheduling medicine in-takes when the user is busy or resting.

Wedjat outputs include full in-take schedule, revised medicine in-take schedule and medicine in-take records. After *Wedjat* create a feasible full in-take schedule, it writes the schedule to the Calendar application through an application programming interface (API). The Calendar application plays the role of reminder and interacts with user directly. If the user miss or delay some dose, the system will readjust the in-take schedule and rewrite to Calendar application. The readjust schedule is revised medicine in-take schedule. When the user clicks OK to confirm the dosage in-take, the system takes down it as medicine in-take records.

B. Software Design

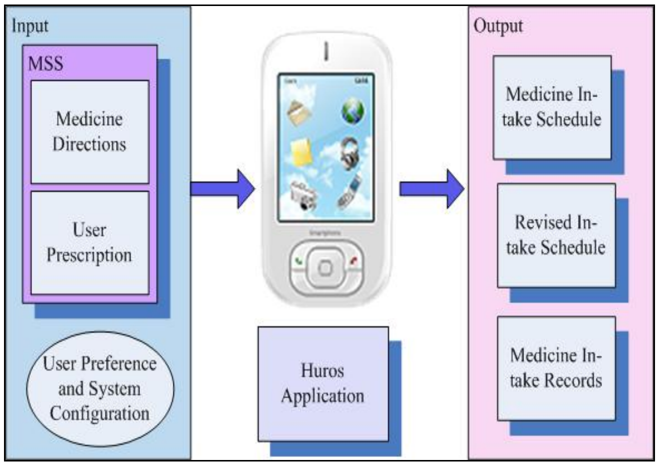


Figure 2: *Wedjat* Input/Output flows

Figure 3 shows the implementation architecture. The *Wedjat* downloads the MSS of user in-take medicines from Indivo. The user sets the preference and then starts the medication scheduler. If the Medication Scheduler creates feasible schedule successfully, it shows some message to graphic user interface and writes the schedule to Calendar application. After that the Medication Scheduler monitors the user take medicine on time or not and readjust the in-take schedule.

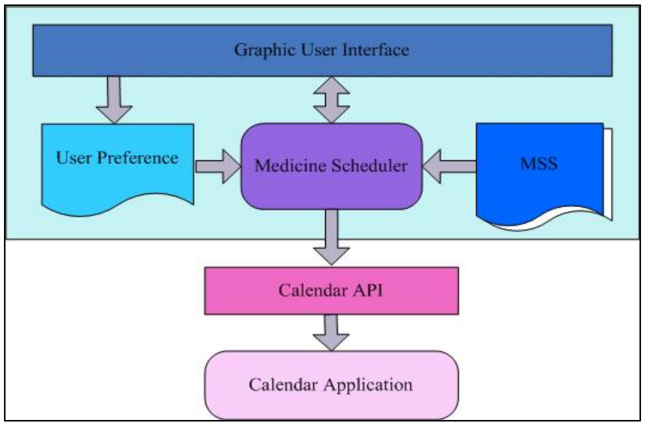


Figure 3 : *Wedjat* Functional Modules

The class diagrams of *Wedjat* are classified to three categories: Medication Schedule Specification, Medicine Priority and Medication Scheduler. As the Figure 4 shows, the data structure of MSS includes Dosage Parameters, Interaction Parameters and Interferer Parameters. It holds all related constraints values. The Medicine Priority defines five schemes of priority including EDF, MIF, MVF, RM and SSDF. The EDF is used by ODAT only. Medication Scheduler is the core of *Wedjat*. It creates a `JobModel` object, a `ResourceModel` object and a `Schedule` object for each medicine. The `JobModel` object holds the values of jobs parameters of each medicine M. The `executionTime` field is set to the nominal minimum separation `nsmin` of M initially. The `ResourceModel` maintains two integer arrays *resource* and *processor*. The initial values of every element of the two arrays are 0 which indicates the *resource* and *processor* are free. When a job is scheduled on processor, the *processor* is set to 1 for k elements in *processor* array which indicates the processor is occupied by the job for the length of execution time k. The *resource* of M is occupied by the interacting medicines or food N of M. Only when the *processor* and M are free for the length of execution time and the *resource* is free at the start instance of job, the job can be scheduled and executed. The `Schedule` array is the outputs of Medication Scheduler to show the full in-take schedule. The Medication Scheduler supports three kinds of scheduling algorithms including ODAT, OMAT with as soon as possible and OMAT with as last as possible. The other methods in Medication Scheduler are private methods used for Medication Scheduler only. The graphic user interface designed in *Wedjat* version. 1 is shows as Figure 6.

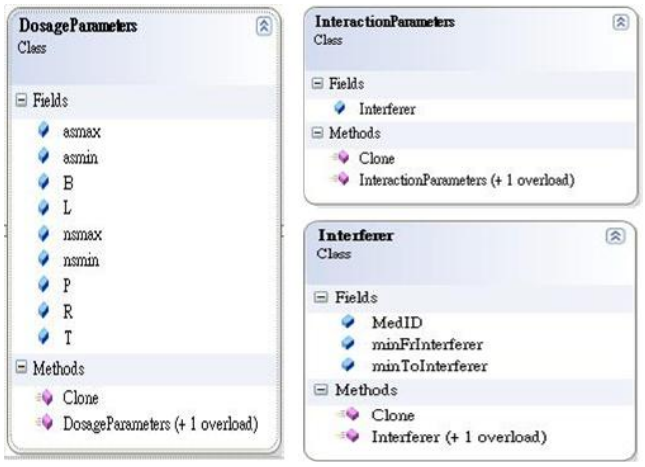


Figure 4 : Medication Scheduling Specification (MSS) Classes

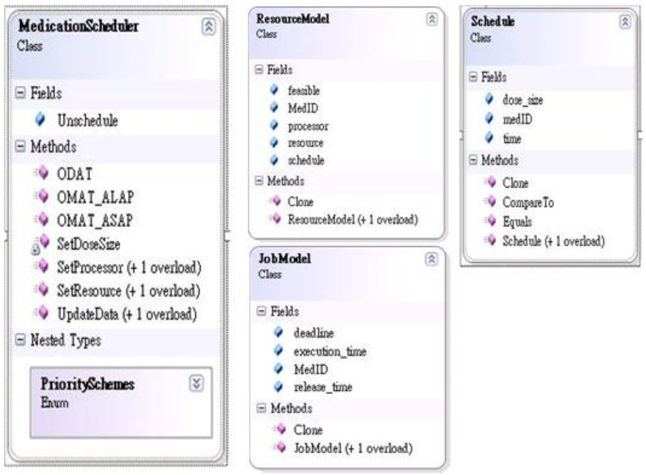


Figure 5 : Medication Scheduler Classes

C. User Interfaces

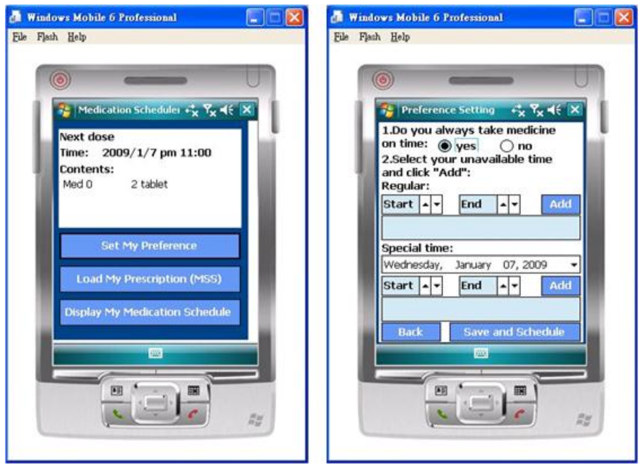
The graphic user interface we design for *Wedjat* version.1 is shows as Figure 6. The first page shows three buttons for user to set preference or load a MSS file or display the result schedule and a text show the next dose user have to intake. The second one shows the preference setting page. The user can set the unavailable time or special time which they don't want to take medicine.

VI. Related Work

As mentioned in the introduction, most attempts to reduce medicine administration errors have focused their efforts on developing "medicine dispensers". Most commercial products [14,15,16,17] are low cost, manual operating devices. A weakness shared by these devices is that their users must load medicine doses into these devices and then program their operation. Naturally, such a cumbersome operation is susceptible to human errors. Automated medicine dispensers [7,8,9,10] are definitely a step taken in the right direction. Some of these devices will be installed in the homes for the elderly or chronically ill. *Wedjat* represents a novel attempt to integrate healthcare support with mobile computing. As next generation smart phones such as iPhone™ and Google Phones™ become popular, such an application has the potential to reap a huge market share.

VII. Conclusions

In this paper, we presented the design ideas behind *Wedjat*, a medicine in-take reminder and monitor installed on a smart phone. This mobile computing application combines mobile phone based telemonitoring techniques with real-time scheduling algorithms to offer ubiquitous services to numerous out-patients. Notable accomplishments include the development of OMAT/ODAT medication scheduling algorithms and the implementation of an integrated mobile computing application. Plenty of work remains to be done after this initial effort: first and foremost, a thorough integration of *Wedjat* with electronic medical records (eMAR) and electronic personal health records (ePHR). Advance scheduling algorithms including those that can produce incremental changes to existing schedules should be investigated. The possibility of exercising on-the-fly changes of prescriptions in response to patient's conditions may also be explored.



Next dose
Time: 2009/1/7 pm 11:00
Contents:
Med 0 2 tablet

1.Do you always take medicine on time: [✓ yes] no
2.Select your unavailable time and click "Add":
Regular:
Special time:
Wednesday, January 07, 2009

Figure 6 : *Wedjat* Graphic User Interfaces

References

[1] Kohn LT, Corrigan JM, Donaldson MS (Ed.), *To Err Is Human: Building a Safer Health System*. Compiled by the Committee on Quality of Health Care in America, the Institute of Medicine, US National Academies; published by National Academy Press, 1999.

[2] *Preventing Medication Errors*. Quality Chasm Series compiled by the Institute of Medicine, US National Academies; published by National Academy Press, July 2006.

[3] Koppel B, et al., “Role of Computerized Physician Order Entry Systems in Facilitating Medication Errors,” *Journal of American Medical Association*, Vol. 293, No. 10, 2005.

[4] Kuperman GJ, Bobb A, Payne TH, et al. “Medication Related Clinical Decision Support in Computerized Provider Order Entry Systems: A Review” Journal of American Medical Informatics Association, 2007.

[5] Baron RJ, Fabens EL, Schiffman M, Wolf E, “Electronic Health Records: Just around the corner? Or over the cliff?” *Annuals of Internal Medicine*, 2005.

[6] Wertheimer AI and Santella TM, “Medication Compliance Research,” *Journal of Applied Research in Clinical and Experimental Therapeutics*, 2003.

[7] Wan D, “Magic Medicine Cabinet: A Situated Portal for Consumer Healthcare,” *Proceedings of 1st International Symposium on Handheld and Ubiquitous Computing (HUC ’99)*, September 1999.

[8] Murray MD, “Automated medication dispensing devices,” Chapter 11 in *Making Healthcare Safer: a Critical Analysis of Patient Safety*, 01-E58, Agent for Healthcare Research and Quality, 2001.

[9] Governo M, Riva V, Fiorini P, Nugent C, “MEDICATE Tele-assistance System”, *11th International Conference on Advance Robotics*. June 2003.

[10] Liu JWS, Shih CS, Tsai PH, Yeh HC, Hsiu PC, Yu CY, Chang WH, “End-User Support for Error Free Medication Process,” Proceedings of High-Confidence Medication Device Software and Systems, pp. 34 – 45, June 2007.

[11] Schmidt DC, et al, “Leader/Followers: A Design Pattern for Efficient Multithreaded Event De-multiplexing and Dispatching,”

[12] P. H. Tsai, H. C. Yeh, C. Y. Yu, P. C. Hsiu, C. S. Shih and J. W. S. Liu (2006), Compliance Enforcement of Temporal and Dosage Constraints, Proceedings of IEEE Real-Time Systems Symposium, December 2006.

[13] Tsai, P. H., C. S. Shih, and J. W. S. Liu, “Algorithms for scheduling multiple interacting medications,” Institute of Information Science Academia Sinica, Taiwan, Technical Report TR-IIS-08-001, April 2008.

[14] e-Pill: http://www.epill.com/dispenser.html.

[15] MD2: http://www.epill.com/md2.html.

[16] Pill boxes: http://www.dynamicliving.com/automated_medication_dispenser.htm.

[17] Rx Showcase, http://www.rxinsider.com/prescription_dispensing_automation.htm.