Efficient Resource Scheduling Architecture for Fast Provisioning of Multimedia Service

Dong Cheul Lee†, Byungjoo Park‡, Ki Eung Kim†, and Jae Jin Lee†,

‡KT Network Technology Laboratory, Yuseong-Gu, Daejeon, Republic of Korea

1. Introduction

In a telecom, field workers visit many customer sites to provide multimedia services and repair faults. Scheduling the workers is hard because there are many tasks and the number of the workers is not sufficient. To tackle this problem, KT had transformed the field workers into ubiquitous workers and used work team based scheduling system. However, this approach was not sufficient to reduce frequent worker’s arrival time changes and to increase operational efficiency. Therefore, we proposed individual worker scheduling system for managing the worker’s schedule efficiently. We also developed an optimal worker selection algorithm based on statistical information when an operator assigns a task to the worker.

Key words: u-Worker, scheduling, provisioning, multimedia service
managed by team hourly. After the operator had selected a suitable team among the teams which satisfying a work region constraint and a maximum allocation count constraint, the task was assigned to the team so that any available u-Workers in the team can dispatch the task and offer the service to the customer. While they were doing their tasks, work managers have been monitoring the progress states of the tasks, and re-scheduled the delayed task.
However, as we mentioned before, this scheduling approach has several disadvantages since the scheduling process is based on the work team. Therefore, we propose individual u-Worker based scheduling process which assigns tasks to the worker directly as described in Fig. 1. In this process, when a customer requests a service, the operator searches suitable u-Workers to deliver the service. Since work regions are managed by u-Worker in this process, the selected workers should have been in charge of the work region which includes the customer site. Work regions are small enough not to make the workers travel for a long time. Also, the workers who have available time slot can receive the tasks and the time slot is managed per 15 minutes.

3. System architecture

To tackle KT’s u-Worker scheduling challenge, we designed the system architecture as shown in Fig. 2. The Schedule Visualizer works as an interface between operators and the scheduling system and visualizes the schedules of u-Workers. It gets schedule data from the Schedule Manager after the Schedule Manager computes an optimal solution for selecting the workers using the Human Resource Manager, the Task Duration Estimator, and the Driving Time Estimator. The workers input their day off, holiday duty, night shift, and business trip information to the Human Resource Manager. The Task Duration Estimator estimates the duration of a task before the worker actually does the task. Also, the Driving Time Estimator estimates the workers’ moving time from one customer site to another before they actually move.

3.1. Task Duration Estimator

When a customer requests a service to a call center operator, the operator should be able to estimate the duration of the task to reserve the worker’s schedule. Arrival time of the worker can be easily known since a customer wants specific visit time. However, the finish time of the task is hard to estimate since the duration of a task can vary with the worker’s experience, service type, and provided network facilities. The Task Duration Estimator can estimate the duration of a task by using statistical data. It calculates and stores the mean duration of previous tasks for each task type and u-Worker. The task types can be classified by IP TV provisioning and IP TV repair. And the network facilities are classified into FTTH, hybrid FTTH, xDSL, IP-xDSL, and Fast Ethernet. Furthermore, the duration can be varied by the day of the week and a time zone. The time zone is classified into forenoon, afternoon, and night. As a result, the mean duration $mdur(n)$ is managed by u-Worker ID, the service type, the facility type, the day of the week, and the time zone. And it is calculated as follows:

$$mdur(n) = \frac{1}{n} \sum_{i=1}^{n} finishtime_i - arrivaltime_i$$

$finishtime_i$ means the time when the worker had completed the $i$th task and it can be obtained when the workers had finished their task and sent the result to the work managers. $arrivaltime_i$ means the time when the worker had arrived at a customer site for the $i$th task and it can be obtained when the workers had arrived at a customer site and notify work managers of their arrival time. Whereas $n$ is the total number of tasks with same worker ID, service type, facility type, day of the week, and time zone for latest one month.

3.2. Driving Time Estimator

Even though call center operators can estimate the duration of the task by using the Task Duration Estimator, they should also consider departure time of a u-Worker from a previous customer site. Therefore, they should also estimate the time to travel from a previous customer site to a next customer site. However, the driving time is hard to estimate since it can vary with moving distance, moving velocity. Also, there can be many uncertainties such as weather and traffic conditions.

The Driving Time Estimator estimates the driving time of the worker from one customer site to another by using statistical data. It calculates and stores mean velocity for each worker. Like the Task Duration Estimator, the driving time can be varied by the day of the week and a time zone. As a result, the mean velocity $mvel(n)$ is managed by u-Worker ID, the day of the week, and the time zone. And it is calculated as follows:
started the first task on a day, week, and a time zone for latest one month. If the workers
the total number of tasks with same worker ID, day of the
Also, driving distance between two points by using the GIS.
converts the addresses to coordinates and computes the
addresses of customers, the Driving Time Estimator
with traffic function enabled. Since we can get the
distances from the previous customer site for each worker
to the new task’s location when the operator assigns a new
task. Because the Driving Time Estimator can get the
driving distance and the mean velocity of the worker, it
can estimate the driving time by dividing the mean
velocity into the driving distance.

3.3. Human Resource Manager
The Human Resource Manager manages a day off, a
holiday duty, a night shift, and business trip information of
u-Workers. It is used to search the available workers
at a specific date by the Schedule Manager. If they had a
day off or a business trip on a certain day, they are
excluded from a resource pool at that time. On the other
hand, if they put in a holiday duty or the night shift, they
are included in the pool at that time.

3.4. Schedule Manager
Call center operators should search a suitable u-Worker
when they assign a task. If they choose a worker without
any consideration at their convenience, they can not
manage worker’s schedule efficiently. However, finding a
suitable worker by themselves is hard due to there are
many factors to consider.
The Schedule Manager searches a near optimal worker for
a task. It uses find_worker() algorithm to find a solution as
described in Fig. 3. The input value of the algorithm
consists of arrival time, service type, and work region.
These values can be obtained while the operator talks over
with a customer. It uses the Driving Time Estimator and
the Task Duration Estimator to estimate the departure time
and finish time for each worker. Also, the Human
Resource Manager is used for excluding workers who
don’t work on weekday and including workers who work
on holiday or at night.

\[
dist_i = \sum_{i=1}^{n} dist_{i-1} \text{arrivaltime}_i - \text{finishtime}_{i-1} / n \quad (2)
\]

dist_, means the previous worker’s driving distance from i-
1th customer site to i-th site. To find dist_, we used GIS
with traffic function enabled. Since we can get the
addresses of customers, the Driving Time Estimator
converts the addresses to coordinates and computes the
driving distance between two points by using the GIS. n is
the total number of tasks with same worker ID, day of the
week, and a time zone for latest one month. If the workers
started the first task on a day, finishtime_, means nine
O’clock since they had started their work at that time.
Also, dist_, means the distance from their local office to a
customer site. Likewise, the GIS also provide the driving
distance from the previous customer site for each worker
to the new task’s location when the operator assigns a new
task. Because the Driving Time Estimator can get the
driving distance and the mean velocity of the worker, it
can estimate the driving time by dividing the mean
velocity into the driving distance.

3.5. Schedule Visualizer
Call center operators should be able to identify and
monitor u-Worker’s schedule. The Schedule Visualizer
display worker’s schedule and interacts with the operators.
It represents the schedule as a Gantt chart or on GIS map
so that the operator can identify the distribution of the
tasks. It also helps the work managers identify the task
that is not likely to be completed on time so that they can
rearrange the next task of the worker.

```python
# Algorithm find_worker

Algorithm find_worker(arrivaltime, servicetype, workregion)
BEGIN
1. DEFINE wlist, templist as worker’s list;
2. IF(arrivaltime==holiday) THEN store workers who are
   responsible for a holiday duty into wlist;
   ELSE IF(arrivaltime==night) THEN store workers who
   are responsible for a night shift into wlist;
   ELSE store workers who don’t have a day off or are not
   on a business trip into wlist;
3. exclude workers who are not in charge of the
   servicetype from wlist;
4. exclude workers who are not responsible for the
   workregion from wlist;
5. templist=wlist;
6. DEFINE stime as the departure time for the task, ftime
   as the finish time of the task;
7. FOR(each worker t in templist)
   BEGIN
   stime=arrivaltime+DrivingTimeEstimator();
   ftime=arrivaltime+TaskDurationEstimator();
   IF time between stime and ftime is overlapped with
   other schedules of t THEN exclude t from templist;
   END
8. IF templist is NULL THEN RETURN wlist;
9. IF templist has workers having no task THEN exclude
   workers who have tasks from templist;
10. delete all workers from templist except top 5 workers
    who have shortest (driving time+task duration);
11. IF(gap between (driving time+task duration) of
    workers in templist is within 10 min) THEN delete all
    workers from templist except a worker who has minimum
    number of tasks;
12. ELSE delete all workers from templist except a worker
    who has shortest (driving time+task duration);
13. RETURN templist;

Fig. 3 Algorithm find_worker()
```
4. Implementation

The u-Worker scheduling system is implemented in KT’s NeOSS-WM (New Operations Support System-Workforce Management). NeOSS-WM consists of PC client, PDA client, application server, EAI server, and database server as shown in Fig. 4.

The PC client is used by call center operators and work managers at local offices. They can get a suitable u-Worker when assigning a task to a worker. They can also monitor the current status of the tasks whether there are delayed tasks. It is implemented as C/S method and works on .NET framework.

U-Workers use the PDA client when they are doing their tasks. They notify the application server of their arrival and finish time by using the client. If an operator has assigned a task to a worker, the application server sends a short message to the worker so that they can identify their schedule and retrieve detail information from the server by using the client. The PDA client communicates with the application server on WiBro or CDMA2000 1x or EVDO network. It uses Windows Mobile .NET platform and MS SQL CE 2.0 DBMS. And it is implemented with C# .NET on .NET framework.

The Driving Time Estimator and the Task Duration Estimator and the Schedule Manager are implemented in the application server as COM+ components. The server communicates with other systems or the clients via the EAI server. BizTalk Server 2004 is used for the EAI server. Both servers operate on Windows Server 2003 platform and load balanced by using L4 switch and protected by a firewall. Server applications are implemented with C# .NET.

The database server stores tasks, schedules, and u-Workers information from the application server and it uses MS SQL Server 2000 as a DBMS. Periodically executed stored procedures are registered as SQL Server jobs for calculating and storing the average durations of the tasks and the average driving velocities of the workers.

5. Conclusions

We proposed the scheduling system for u-Workers especially to address KT’s worker scheduling challenge for fast provisioning of multimedia service. To solve this problem, we designed five modules which are the Task Duration Estimator, the Driving Time Estimator, the Schedule Visualizer, the Human Resource Manager, and Schedule Manager. They were implemented in NeOSS-WM and the system has been used for scheduling KT’s u-Workers successfully.

As a future work, we will improve NeOSS-WM to solve not only KT’s workforce scheduling problems but also other resource scheduling problems.

References


Dong Cheul Lee received the B.S. and M.S. degrees in Computer Science and Engineering from Pohang University of Science and Technology, Pohang, Korea in 2002 and 2004, respectively. He has been a junior researcher in the KT Network Technology Laboratory, Korea since 2004. His research interest includes algorithm and application of mobile communications, including routing protocol design and network engineering in the Broadband Convergence Network, workforce scheduling issues for network service provisioning, and task scheduling algorithms for GRID environment. He is a member of the TM Forum, IASTED, KICS, KISS.

Byungjoo Park received the B.S. degree in Electronics Engineering from Yonsei University, Seoul, Korea in 2002, and the M.S. and Ph.D. degrees (First Class Honors) in Electrical and Computer Engineering from University of Florida, Gainesville, USA, in 2004 and 2007, respectively. He is currently a senior researcher in the KT Network Technology Laboratory, Korea. He is a member of the IEEE, IEICE, IEEE, KIIE and KICS. His primary research interests include theory and application of mobile computing, including protocol design and performance analysis in the next generation wireless/mobile networks. Since 2004, his activities have focused on IPv6, IPv6 mobility management, End-to-End QoS provisions and cross-layer optimization for efficient mobility support on IEEE 802 wireless networks. He has published approximately 25 research papers on mobile computing. He is an honor society member of Tau Beta Pi and Eta Kappa Nu.

Ki Eung Kim received the B.S and M.S. degrees in Electronics Engineering from Dankook University, Korea in 1984 and 1986, respectively. Since he joined KT in 1989, he had researched various networks including ATM, PSTN, IP, and Wireless network. Currently, he is a principal researcher and the task leader in IP Backbone Task, Network Technology Laboratory, KT, Korea. His research interest covers network traffic engineering, circuit switching technique, Quality of Service, routing algorithms, and network management system. Since 2007, his activities have focused on control and management method on IP backbone networks in emergency.

Jae Jin Lee is an assistant vice president and the group leader in IP Network Research Group, Network Technology Laboratory, KT, Korea. He received the B.S. and M.S. degrees in Electrical Engineering from Kyungpook National University, Daegu, Korea in 1985 and 1987, respectively, and the Ph.D. degree in Electrical Engineering from Korea University, Seoul, Korea, in 1997. Since he joined KT in 1987, for 21 years he worked for the research and development of various IP network management systems including PSTN, ATM, Fixed and Mobile Internet NMSs. His research interest include Broadband Convergence Network, routing algorithms, traffic network management engineering, fixed and mobile IP network architecture and NGOSS (Next Generation Operation Support Systems).