International Journal of Informatics Society

06/14 Vol. 6 No.1 ISSN 1883-4566

Informatics Society

Aims and Scope

The purpose of this journal is to provide an open forum to publish high quality research papers in the areas of informatics and related fields to promote the exchange of research ideas, experiences and results.

Informatics is the systematic study of Information and the application of research methods to study Information systems and services. It deals primarily with human aspects of information, such as its quality and value as a resource. Informatics also referred to as Information science, studies the structure, algorithms, behavior, and interactions of natural and artificial systems that store, process, access and communicate information. It also develops its own conceptual and theoretical foundations and utilizes foundations developed in other fields. The advent of computers, its ubiquity and ease to use has led to the study of informatics that has computational, cognitive and social aspects, including study of the social impact of information technologies.

The characteristic of informatics' context is amalgamation of technologies. For creating an informatics product, it is necessary to integrate many technologies, such as mathematics, linguistics, engineering and other emerging new fields.

Guest Editor's Message

Kozo Okano

Guest Editor of Sixteenth Issue of International Journal of Informatics Society

We are delighted to have the sixteenth and special of the International Journal of Informatics Society (IJIS) published. This issue includes selected papers from the Seventh International Workshop on Informatics (IWIN2013), which was held at Stockholm, Sweden, Sep 1-4, 2013. The workshop was the seventh event for the Informatics Society, and was intended to bring together researchers and practitioners to share and exchange their experiences, discuss challenges and present original ideas in all aspects of informatics and computer networks. In the workshop 24 papers were presented at five technical sessions. The workshop was complete in success. It highlighted the lasts research results in the area of networking, business systems, education systems, design methodology, groupware and social systems.

Each paper submitted IWIN2013 was reviewed in terms of technical content and scientific rigor, novelty, originality and quality of presentation by at least two reviewers. From those reviews 15 papers are selected for publication candidates of IJIS Journal. This fourteenth includes four papers of them. The selected papers have been reviewed form their original paper presented in IWIN and accepted as publication of IJIS. The papers were improved based on reviewers' comments.

We hope that the issue would be interest to many researchers as well as engineers and practitioners in this area.

We publish the journal in print as well as in an electronic form over Internet. This way, the paper will be available on a global basis.

Kozo Okano received his BE, ME, and Ph. D degrees in Information and Computer Sciences from Osaka University, in 1990, 1992, and 1995, respectively. Since 2002 has been an associate professor in the Graduate School of Information Science and Technology, Osaka University. In 2002 and 2003, he was a visiting researcher and visiting lecturer at the Department of Computer Science, University of Kent at Canterbury and the School of Computer Science, University of Birmingham, respectively. His research interests include formal methods and their application to Software Engineering. He is a senior member of IEICE and a member of IPSJ.

Extending Battery Lifetime in Smartphones with Power Efficient Task Management and Energy Aware Design Tool

Hiroshi Inamura[†], Takeshi Kamiyama[†], Teppei Konishi[†], and Ken Ohta[†]

[†]Research Laboratories, NTT DOCOMO, Inc.
inamura@nttdocomo.com, kamiyamat@nttdocomo.co.jp, teppei.konishi.xt@nttdocomo.com, ohtak@nttdocomo.co.jp

Abstract - In the use of smartphone, the battery lifetime is important factor that determines how long users can use the applications on the device. The application developers need to be aware of energy consumption for their application. In order to extend the battery lifetime, we recognize two issues; 1) Allowing developers to know the power-efficiency of their applications in real user environments for applications in design, 2) Optimizing the system to be power-efficient in the execution of background tasks for existing applications. We implemented a tool for developers to visualize how their code consumes the energy, with low overhead for data collection in runtime. We also improved the power efficiency in the background task management in Android OS and it shows up to 40% power reduction in call waiting time.

Keywords: Power Management, Android, Background Task, Global Synchronization, Power Modeling

1. INTRODUCTION

With the wider use of smartphone, it has become important to improve the battery lifetime of devices. Comparing to the featurephone, the smartphones are flexible both in development and deployment of 3rd party applications. Among major platforms of smartphones, a large percentage of devices are running Android OS [1]. Users are attracted to smartphones for the variety of applications available through application market. It is preferable all the applications are power efficient. The application developers need to be aware of energy consumption for their application. In order to extend the battery lifetime, we recognize two issues;

1) **Allowing developers to know the energy-efficiency of their applications in real user environments for applications in design**. The power consumption of device is determined by the utilization or the state of each hardware components such as CPU, which depends on the behavior of the application software. It is possible to prolong the battery lifetime by eliminating inefficiency from the applications, such as too frequent data transfer, for example. As a case study on a particular application, Furusho [2] showed that the energy consumption could be decreased by optimizing the application's behavior through an analysis of real users' actions. Given the competitiveness of the smartphone market, it is critical that developers be aware of how much power their applications consume and of the "Power Cost" of each function. The energy visualization integrated in SDK tool will be great help for developer to know the "Power Cost" in early stage of design.
2) **Optimizing the system to be power-efficient in the execution of background tasks for existing applications.** The background task (BG task) is a type of application component that does care-taker for processes with long waiting time and the role of service provider. A class of BG task has arbitrariness for the timing to be executed and does not cause any visible change to the user interaction. We can utilize this characteristic to shift the execution timing to minimize the utilization of hardware components. On the managing the execution timing, we need to consider the possibility of the synchronization of task executions among device, since it may leads network congestion if the communicating tasks are involved.

In this paper we describe an approach to extending the battery lifetime with our solutions for these two issues. With the combination of solutions, the extension can be made both for applications to be designed as well as existing applications on the market. We implemented a software tool for developer to visualize how their code consumes the energy, with low overhead for data collection in runtime. Also, we improved the power efficiency in the background task management in Android OS and it shows up to 40% power reduction for call waiting time.

2. ALLOWING DEVELOPERS TO KNOW THE POWER-EFFICIENCY OF THEIR APPLICATIONS

Software-based energy profiling demands an accurate power estimation scheme. The scheme should satisfy the following points. **a) Accurate estimation based on modeled characteristics of hardware components. b) Estimation is based on the data obtained through actual usage of applications.**

We propose an energy profiler for Android applications [3] based on a power model of devices and data obtained during applications usage.

To achieve a), we extend the existing power model proposed in [3]**[5]** to take account of new features of hardware components such as multi-core CPUs with dynamic frequency scaling and 3G/LTE RRC State transition. Regression analysis is used to allow the model to catch the relationship between power consumption and the behavior of each hardware component. In other words, the model is an ensemble of formulas, each with its own coefficients and parameters that express the power consumption of each component.

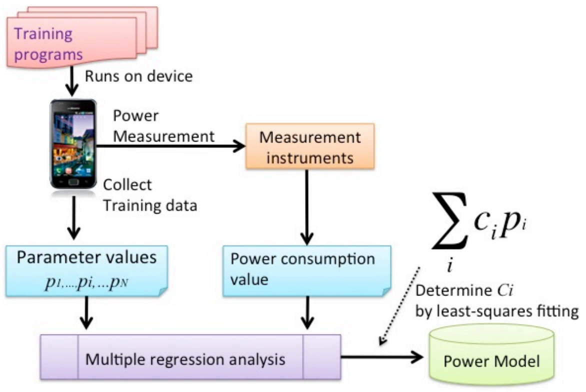


Figure 1: The process of power characterization

To achieve b), we log target applications while running to obtain the data needed generate the model parameters. The data reflects the impact of actual application usage on real devices, so the estimation results are practical. Once the model is generated and loaded into the profiler, there is no need to measure power consumption directly.

Based on the above ideas, we pose 2 requirements for the profiler as follows; 1) Accuracy of energy estimation targeting overall system is sufficient. 2) Logging overhead is small.

2.1 Related Works

Kaneda [5] proposed a system-wide power model that accurately estimates the power consumption. The model covers the main components and the power consumed by each component is parameterized by common data which is easily obtained at the OS-level, such as CPU utilization. Thus the good feature of their model design is not only that it's extendable for system-wide estimation but also each component can be logged with small overhead. However, it doesn't consider the recent features of components such as frequency scaling of multi-core CPUs and mobile wireless interfaces such as 3G/LTE.

ARO (Application Resource Optimizer) is a profiler that does address such recent features [6]. It features a power model of the mobile wireless interface. In 3G/LTE networks, the wireless link between device and base station is managed by RRC (Radio Resource Control), which handles the bearer setup and release, and mobility management [8][9]. RRC State used as model parameters can be estimated using the RRC State machine that specifies the states from packet capture data. However, special privilege such as root is needed on common devices to permit packet capture for logging and such logging incurs high CPU time overhead.

The previous works don't satisfy our energy profiler requirements. We describe an energy profiler by following the model design of [3][5]. This approach makes it easy to cover the overall system, has low logging overheads, and allows the model to be extended to account for the features of recent hardware components like ARO

.

2.2 Energy Estimation using Power Model

We describe here our device power model for energy estimation. As follows, the device model is expressed as the sum of the power consumption of each hardware component.

$$P_{est} = c_{offset} + \sum_i^N c_i p_i \qquad (1)$$

Where P_{est} is the total power consumption of the device. N is the number of hardware components. p_i is a parameter that covers the power consumption of hardware component i. c_{offset} and c_i are coefficients to be determined through multiple regression analysis.

Figure 1 shows the process of power characterization to generate the power model of a specific device. As in the existing scheme [5], power characterization is done by multiple regression analysis using the above equation and training programs to collect training data used in the analysis. To collect the data, sets of power consumption values and parameter values, p_i, are measured under many test cases by the programs that run on the device. The coefficients of the equation, c_{offset} and c_i, are then obtained as a result of analyzing the training data.

One of our key points is that the parameters should be easy-to-obtained common values, such as CPU utilization, because it allows the process of power characterization to be as independent of device type as possible. We follow the basic design of our existing power model but extend it to cover the new features of multi-core CPUs and 3G/LTE.

2.2.1 Multi-core CPU

Recent smartphones use multi-core CPUs that can switch the number of active cores and scale frequency for energy-efficiency. Because our existing model assumes a single core CPU and only CPU utilization is used as a parameter, it has difficulty in accurately estimating the power consumption. We introduce a new model that can handle the complexity of multi-core operation.

Our new multi-core CPU power model is expressed as follows.

$$\begin{aligned} P_{cpu} &= P_{core1} + P_{core2} + \cdots + P_{coreN} \\ &= \sum_i^N c_i p_i = \sum_i^N c_i f_i u_i \end{aligned} \qquad (2)$$

Where P_{cpu} and P_{core1} to P_{coreN} are the overall power consumption of target CPU and the power consumption of each core in the CPU. We assume power consumption of each core can be separately modeled because recent frequency scaling technology allows cores to work independently.

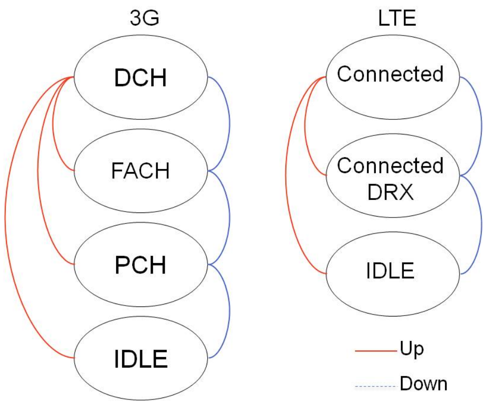


Figure 2: RRC State transition

Table 1: Power Model of Target Device

Component	Model Equation
CPU	$c_1 f_{core1} u_{core1} + c_2 f_{core2} u_{core2}$
Display	$c_3 p_{brightness}$
3G	$(c_4 + c_5 p_{sendBps} + c_6 p_{rcvBps}) p_{pch} + c_7 p_{fach} + c_8 p_{pch} + c_9 p_{idle}$
LTE	$(c_{10} + c_{11} p_{sendBps} + c_{12} p_{rcvBps}) p_{LTEconnected} + c_{13} p_{LTEcdrx} + c_{14} p_{LTEidle}$
Wi-Fi	$(c_{15} + c_{16} p_{sendBps} + c_{17} p_{rcvBps}) p_{flagWiFi}$
GPS	$c_{18} p_{gps}$
Disk	$c_{19} p_{readByte} + c_{20} p_{writeByte}$
GPU	$c_{21} p_{gpuLoad}$

Table 2: Values of Coefficients

i	Coefficient	i	Coefficient
0	1.462e-01	11	8.000e-08
1	1.375e-09	12	2.048e-08
2	1.162e-07	13	3.892e-02
3	2.340e-04	14	1.723e-02
4	1.563e-01	15	3.210e-02
5	3.265e-07	16	1.633e-07
6	1.107e-07	17	1.218e-07
7	1.179e-01	18	3.712e-02
8	1.384e-02	19	3.307e-10
9	1.402e-02	20	1.336e-09
10	2.054e-01	21	7.797e-10

c_i, p_i, f_i and u_i are the coefficient and parameter for core i, frequency value of core I, and CPU utilization rate of core I, respectively. As shown in (2), we define the core's work load, the product of f_i and u_i, as a new power consumption parameter for each core. This is because we think the parameter should allow consideration of the change of performance due to dynamic frequency scaling.

2.2.2 3G/LTE

In 3G/LTE networks, wireless link between device and base station is managed by RRC (Radio Resource Control), which handles the bearer setup and release, and mobility management [8][9]. The multiple link states, called RRC State, are defined in Fig.2, and the state is automatically switched depending on data transfer from/to device for the purpose of efficient use of wireless resources.

For example in the case of 3G, DCH is the state for the high throughput data transfer when using applications. FACH is the state for only low throughput. PCH is the stand-by state (entered when no data is transferred within a certain period). IDLE is the state of wireless link released. State transitions occur in various conditions as defined in RRC, such as data transfer defined in each state. For example, upper transition to DCH from lower states occurs when data transfer is requested. A lower transition to FACH, PCH or IDLE occurs after the inactivity timer, which runs while no data is being transferred, times-out. From the point of power consumption, it is known that more power is consumed in the upper state.

ARO accurately estimates the power consumption of the 3G/LTE interface using RRC State as a model parameter. In addition, ARO obtains the parameter value by RRC State estimation because the state cannot be directly accessed on general devices. Therefore, ARO loads the RRC State machine; it estimates the states by detecting data transfer from packet capture data (pcap), that is collected by the device. However, this can impose big CPU time overhead and needs super-user privileges. These limitations do not yield a widely provided, easy-to-use, profiler for general application developers.

In our work, we follow the basic idea of ARO's RRC state machine to extend our existing power model of wireless interface, and avoid the limitations noted. In detail, we replace packet capture logging with lightweight periodic logging of network statistics, which are easily accessed at the OS or API-level, such as /proc/net/dev. This yields much lower overheads because data volume is less than with than packet capture.

2.2.3 Result of Power Characterization

We show a result of power characterization for a real Android device that uses Qualcomm's MSM8960 [9] chipset. Model equations including parameters are shown in Table 1, and values of coefficients are shown in Table 2.

2.3 Application Energy Profiler

This chapter describes our application energy profiler equipped with the power model generated in the last chapter. Figure 4 shows the functional design of the profiler. The system consists of the Logger application, which runs on the device to collect the data needed for energy profiling, and the energy profiler, which analyzes energy consumption of the target application.

The logger application collects the log data needed for parameter generation at one second intervals while the target application is active. The energy profiler calculates the device's average power consumption in one second periods using the parameter values generated from the log data. Finally, the profiler shows the energy consumed by the application on a time chart, see Fig.3. The power breakdown for each hardware component is also shown in the chart. It enables the application developer to become aware of what factor in driving power consumption.

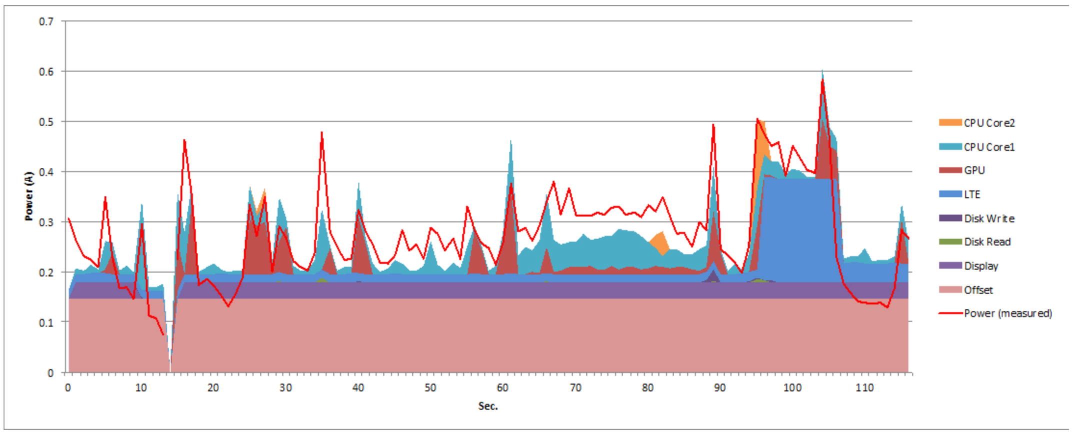


Figure 3: Estimated and measured power consumption for the mail application scenario.

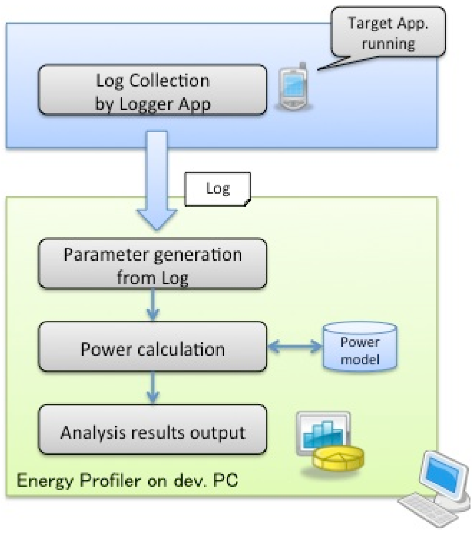

Target App. running
Log Collection by Logger App
Log
Parameter generation from Log
Power calculation
Power model
Analysis results output
Energy Profiler on dev. PC

Figure 4: Functional design of Energy Profiler

power consumption of OLED depends on the color and is able to be explained by RGB value of each pixel. As a result, dynamic change of color tone on the screen occurred and became the error factor.

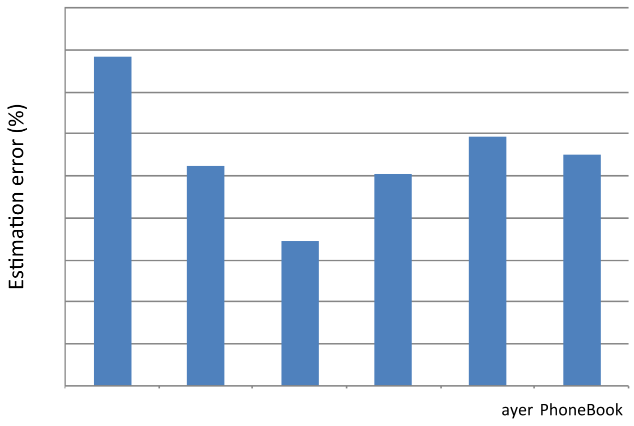


Figure 5: Estimation error in each scenario.

2.4 Evaluation

We evaluated our energy profiler according to the two requirements, 1) accurate power estimation and 2) lightweight online logging, respectively.

2.4.1 Estimation Accuracy

To evaluate the accuracy of the energy estimates we actually measured the consumption under the following commonly used application usage of Mail, Map, Calendar, Movie player and Phonebook.

Figure 3 compares measured and estimated power consumption for the mail scenario 1). It is confirmed that the estimation basically follows the actual consumption for the entire period.

Through the all scenario, the profiler estimates energy consumption with about 10% error in application-mix as shown in Fig.3. Thus, it is considered that the profiler can works with constant accuracy.

However, at most 0.1A errors are observed around at 20 second, the period of 40 to 90 second and around at 110 second in Fig.3. We suppose the feature of display causes these errors because the device used in this time is equipped with OLED display; however our power model originally assumes LCD display. According to previous work [7],

2.4.2 Overhead of logging

According to requirement 2), lightweight logging is important so as not to disturb natural use of the application in the tests. We pointed out the problem of ARO's logging overhead and presented a more lightweight logging approach for RRC State estimation to satisfy the requirement. Thus we compared the proposal to ARO in terms of CPU time overhead.

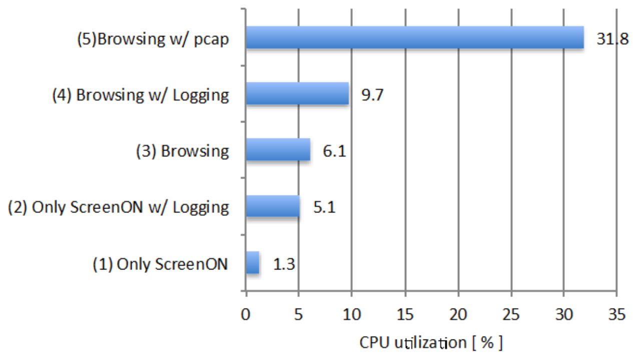


Figure 6: Comparison of CPU time overhead for logging

The logger application collects log data every second while the target application is running. First, we measured CPU utilization in the following test cases in order to understand the application's overhead.

(1) Leave the device as it is without logging

(2) Leave the device as it is with logging proposal

The results, shown in Fig.6, indicate that logging imposes 3.8 % of CPU time overhead (compare (1) to (2)).

Next, we test following test cases to compare the overhead of our logging with that of ARO.

(3) Automatic browsing without logging

(4) Automatic browsing with logging proposal

(5) Automatic browsing with ARO logging (pcap)

These test cases must involve data transfer because ARO's logging is packets capture and work on only condition that data transfer occurs. We use a test program that automatically download and display fixed Web pages on browser.

From the results of (3)-(5), the overhead of our logging proposal is 3.6% (compare (3) to (4)), whereas ARO's overhead is 25.7 %. As CPU utilization is one of the parameters of the CPU power model presented in section 2.2, the power consumption increases due to this overhead. Therefore, energy consumption estimated using packet capture could differ from the real consumption in normal usage of the application. Offsetting this effect is rather difficult because CPU load of packet capture depends on traffic pattern/volume of application usage. Furthermore, many applications for smartphones generally involve data transfer. Thus ARO logging may disturb natural application usage in many cases.

CPU overhead of our periodic logging proposal is small and always constant because it doesn't depend on the kind of application used. As stated above, our method achieves both accurate and lightweight estimation of 3G/LTE power consumption.

3. OPTIMIZING THE SYSTEM POWER-EFFICIENT IN BACKGROUND TASKS FOR EXISTING APPLICATIONS

We describe it is possible to improve the power efficiency in the execution of background tasks by shifting the execution timing. On the managing the execution timing, we need to consider the possibility of the synchronization of task invocations among device, since it may leads to network congestion if the communicating tasks are involved.

3.1 The BG task management mechanisms

Android OS has AlarmManager (AM) [12] that is a mechanism to invoke registered tasks at specified time and Broadcast Intent (BI) [13] that start tasks on the specified state change in the device during the circulation of event message. In this paper, we define wakeup task for tasks calling AM API having time spec with waking up directive (namely, RTC_WAKEUP, ELAPSED_REALTIME_WAKEUP). When the specified time has come, they wake up the device if it asleep. We also define non-wakeup task for tasks using non-wake up directive (namely, RTC, ELAPSED_REALTIME). The non-wakeup tasks are executed at exact time specified if the device already running, and at the most recent wakeup if the device asleep at the specified time.

The BG task invocation on state change happen when the devices' status has changed such as the screen turned on/off, change in the battery status and receipt on SMS, so on. For example, suppose a battery meter application that displays the remaining amount of electric energy in the battery. If the amount changed, the Android OS notify the event using a Broadcast Intent (BI) and the application updates the displayed info. In order to circulate the BI, the device is waked up. Therefore, the wake up for BI circulation will initiate the BG task invocation scheduled in AM. We define the BG task that is executed on BI, as system event (BI).

3.2 Avoiding the network congestion caused by the simultaneous invocation of BG tasks

We need to avoid the network congestion while enabling the simultaneous execution of background tasks for power efficiency.

3.2.1 Network Congestion by simultaneous BG task invocations

Since it is possible to specify the time of BG task invocation precisely in AM API, the network congestion is concerned caused by simultaneous BG tasks invocation at the specific time among many devices with the software created and operated carelessly [14][15]. The interworking/interference between wakeup task and non-wakeup task can synchronize devices globally at specific time. We define such unintended synchronization as "sync by interference" that occur when the device wakes-up with wakeup task, the device runs non-wakeup tasks. The issue is the invocation of non-wakeup tasks aligning to the specific time the wakeup task stated. In order to avoid the network congestion, we need to suppress the "synch by interference" among tasks that may initiates network communication when it is invocated. For preventing the network congestion by application activities, suppressing the "synch by interference" is important issue.

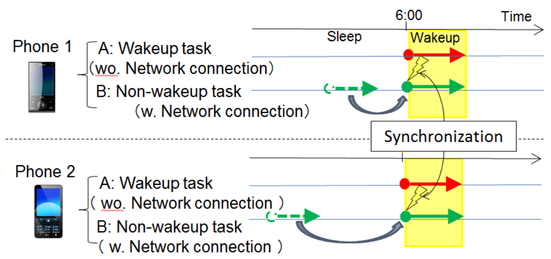


Figure 7: Non-wakeup task is invoked with wakeup task, which causes unintended global synchronization of task invocation among devices

3.3 Case: A Global Synchronization by SBI

Figure 1 shows the two devices, namely phone1 and phone2, each has been installed app A and app B. The app A wakes up device at 6am every day, and it does not start any communication but does some housekeeping task. The app B is a non-wakeup task and will start connection to other servers. As shown in Fig.1, even if the start timing for app B set to different time stamp such as 5:58 and 5:55, the invocation of app B is pending towards 6am, the app A will invokes app B in both cases by the device wakeup. So the app B will start communication at 6am for both devices at globally synchronized. It is easy for developers to expect the synchronization caused by communicating app that wakes up at specific time. The difficulty is to foresee the wakeup task without networking can cause congestion. Under such circumstances, the developer may specify every 00 minutes or 30 minutes of much "aligned" time for the invocation in the application. The developer of app B pays no attention for the possibility of congestion, since it does not specify exact time for invocation. Still, as shown here, there is possibility for global synchronization by the combination of different type of BG tasks.

3.4 Related Works

We could not find any single previous work to cover our issue. For the power optimization by simultaneous task execution, Calder [17], Kononen [18] showed the effectiveness. But we need consideration for network congestion. Yamamoto [19] and other related study [20] worked on power optimization and avoiding network congestion, still we need to preserve developer' s design for controlling the invocation timing. Although the task killer applications [16] are solving the power issues utilizing user intervention, they do not keep original behavior in applications.

3.5 Issues in BG task management mechanism and a proposal for a new control scheme

We propose a new BG task management mechanism that simultaneously invokes non-wakeup task with wakeup task in inexact type only. The mechanism preserves their semantics since the invocation timing in non-wakeup tasks have arbitrariness in BG and in wakeup task with inexact type is chosen by system inherently. Our proposal can reduce the energy consumption in BG task and avoid the network congestion.

3.5.1 Implication in API semantics provided by AlarmManager

In order to modify the invocation timing of BG tasks with less impact to intended behavior in the applications, we need to control them keeping the real time semantics designed in AM API. According to the API design, the real-time semantics can be classified into following three categories.

- Exact real time(API: wakeup task with 'exact' specifier)

The execution time needs to align with real time clock's specific time stamp.

- Exact interval(API: wakeup task with 'inexact' specifier)

The invocation need not be exactly specified time, but the interval between the invocations should be specified amount of time. AM has an SetInexactRepeating API for the purpose that keeps the interval, with the first invocation the system specified.

- Non real time (API: Non wakeup task)

There is no strong requirement in the invocation time and their interval in BG execution. Since it does not wake up the device, the invocation is opportunistic that any time after the designated time is acceptable, during the screen is off.

3.5.2 Avoiding the global synchronization between task invocations

The current implementation of AM invokes the non-wakeup tasks on device wakeup caused by the system events (BI) or wakeup tasks and they are executed altogether The "synch by interference" (SBI) happens when simultaneous execution of wakeup tasks with exact designation and non-wakeup task. Since system chooses specific timing for the wakeup task with inexact designate, it does not cause SBI issue. In order to prevent SBI, we propose a mechanism that exclude wakeup task with exact designate from simultaneous execution with non-wakeup tasks.

3.5.3 Power Reduction in simultaneous execution of Non-wakeup tasks

Using a model equation, we describe the energy consumption can be reduced by simultaneous execution of non-wakeup tasks and wakeup tasks better than non-wakeup tasks and system events (BI). As showed in Sec 2, non-wakeup tasks are executed on the device wakeup later than specified time stamp. So, non-wakeup task (N) is simultaneously executed with either wakeup task (W) or system events (BI).

$$\sum_{\{BI,W,N\}}^{i} t_i (P_{off} + P_c)$$

The power consumption with the simultaneous execution of all the three types (W, N, BI) is described as follows. Let P_{off} as the baseline power consumption of zero utilization in all resources. P_c as the averaged power consumption over unit time. $t_{\{N,W,BI\}}$ are the execution time for each type of BG tasks respectively non-wakeup (N), wakeup (W) and system events (BI). With the optimized simultaneous execution in our proposal, the power consumption modeled as follows. For timing manipulation, because we cannot change the timing of wakeup (W) and system event (BI), we will set the non-wakeup (N) aligned to either one of (W, BI) to be invoked. The power offset are shared during the simultaneous execution.

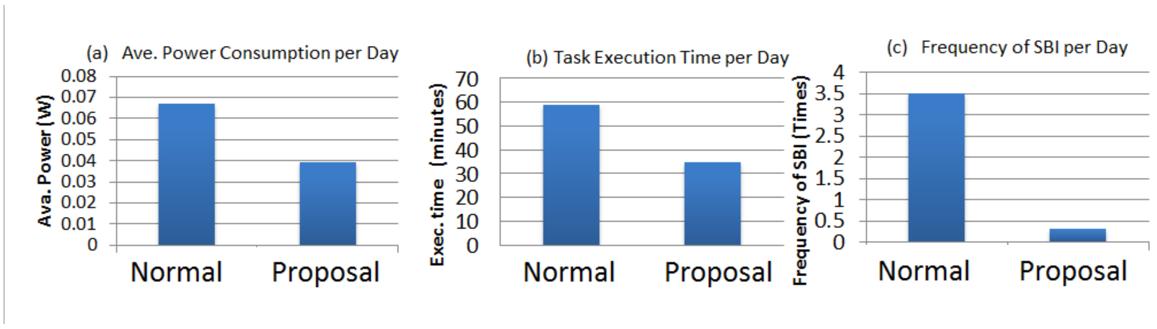


Figure 8: Evaluation of Our Proposed Method

Based on Sec 3.1, suppose the execution time is the longest among the tasks executed in parallel and the resource utilization will be the total of each, the power consumption is given in;

$$P_c \sum_{\{BI,W,N\}}^{i} t_i + P_{off} \min \begin{pmatrix} \max(t_{BI}, t_N) + t_W \\ \max(t_W, t_N) + t_{BI} \end{pmatrix}$$

Looking at Table 3 summarized the result in our user study, comparing the averaged execution time for non-wakeup task and system event are 7.8[s] and 0.5[s], respectively and differs 1:16 ratio. Considering with the averaged execution time in wakeup of 8.2[s], we see $t_W \cong t_N > t_{BI}$; therefore, the non-wakeup (N) task should be executed only with wakeup (W). With the consideration, we derive the optimized power consumption in model as;

$$P_c \sum_{\{BI,W,N\}}^{i} t_i + P_{off}(\max(t_W, t_N) + t_{BI})$$

The non-wakeup task should be executed with wakeup task in the same time, which improves the power consumption of the device as a whole.

Table 3: Average execution time of BG task

Execution Time	AM tasks		system event
	wakeup task	non-wakeup task	
Average (sec)	8.2	7.8	0.5
Std. Dev. (sec)	3.1	2.4	0.1

3.6 BG task control scheme

We propose a mechanism that invokes non-wakeup task only with Exact Interval wakeup. Since the execution timing will be chosen by system and randomized in Exact Interval wakeup, there is no risk for causing SBI. We have a chance to optimize power efficiency in BG task by simultaneous execution of non-wakeup tasks. As we discussed in 3.5.2, execution with exact real-time may cause SBI. For power efficiency, we exclude BI from execution opportunity as seen in 3.5.3. The remaining timing opportunity for execution is with Exact Interval wakeup in AM.

With this scheme, it is possible to apply the optimized BG task invocation to existing applications as is, without re-designing them to adopt to any new API. Our idea is preserving real-time semantics in existing AM API with minimum modification in task invocation policy.

3.7 Implementation and Evaluation

We have implemented our proposed scheme on AOSP Android r4.2.1. We have modified the task invocation policy in AlarmManager into following;

1. Classify the trigger of wakeup
2. Only if the wakeup is caused by SetInexactRepeating, invoke the pending non-wakeup task

The power consumption of the device with the implementation of our proposal and without are shown in Fig.8(a) as 0.039[W] and 0.067[W], respectively. It means 40% reduction while screen turned off (call waiting). The result explained as the reduction of execution time in BG shown in 8(b). In addition, the synchronization between non-wakeup task and exact designated wakeup task is 0.3 [times/hour] in our scheme, while 3.5 [times/hour] in normal device. So, we could eliminate 90% of SI issue during the screen is turned off.

Our proposed method ensures the real time constraint designed in application and avoids re-design them for improvements. So there is least impact for user and developer perspective while achieving energy reduction. Also we could effectively eliminated the issue of “synch by interference” and network congestion that may be caused by concentration of task execution among devices. Since the amount of traffic from devices during the screen turned off is determined the behavior of BG tasks, our proposal is successful in prevention of network congestion as well.

4. SUMMARY AND CONCLUSION

In this paper we describe an approach to extending the battery lifetime with these two issues and our proposals. With the combination of solutions, the extension can be made both for applications to be designed as well as existing applications on the market.

We implemented a tool for developer to visualize how their code consumes the energy, with low overhead for data collection in runtime. Our implementation is a model-based energy profiler for Android applications, taking account of the features of multi-core CPUs and 3G/LTE; Experiments showed that it estimates energy consumption with about 10% error in the application-mix examined, and that logging incurs a CPU time overhead of only 3.8%, which superior to other profilers. We improved the power efficiency in the background task management in Android OS and it showed up to 40% power reduction for call waiting time. We also pointed out the issue of “synch by interference” and our scheme successfully suppresses the issue as well.

REFERENCES

[1] Google.Inc : Android http://www.android.com/about/. (accessed 2013-04-21).

[2] H. Furusho , K. Hisazumi , T. Kamiyama , H. Inamura , T. Nakanishi, and A. Fukuda, ``An Energy Profiler for Android Applications Used in the Real World,'' Proc. of the 10th International Conference on Mobile Systems, Applications and Services (MobiSys'12), pp.517-518 (2012).

[3] T. Kamiyama, H. Inamura, and K. Ohta, ``Evaluation of Model based Energy Profiler for Android Applications,'' Proc. of DICOMO2013 Symosium, pp286-292 (2013).

[4] T. Kamiyama, and M. Katagiri, ``Visualization of mobile terminal power consumption based on OS-level analysis,'' NTT DOCOMO Technical Journal, Vol.11 No.3, pp.72-76 (2009).

[5] Y. Kaneda, T. Okuhira, T. Ishihara, K. Hisazumi, T. Kamiyama, and M. Katagiri, ``A Run-Time Power Analysis Method using OS-Observable Parameters for Mobile Terminals,'' Proc. of International Conference on Embedded Systems and Intelligent Technology, pp.39-44 (2010).

[6] F. Qian, Z. Wang, A. Gerber, Z. Morley Mao, S. Sen, and O. Spatscheck, ``Profiling Resource Usage for Mobile Applications: A Cross-layer Approach,'' Proc. of MobiSys 2011, pp.321-334 (2011).

[7] M. Dong, and L.Zhong, ``Chameleon: A Color-Adaptive Web Browser for Mobile OLED Displays,'' Proc. of MobiSys 2011, pp.85-98 (2011).

[8] 3GPP TS 25.331 Radio Resource Control (RRC); Protocol specification, 3GPP Website, http://www.3gpp.org/ftp/Specs/html-info/25331.htm.

[9] Qualcomm Inc., http://www.qualcomm.com/snapdragon.

[10] 3G Specifications, 3GPP Website, http://www.3gpp.org/3GPP-specifications.

[11] 3GPP TS 36.331 Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification, 3GPP Website, http://www.3gpp.org/ftp/Specs/html-info/36331.htm.

[12] Android API: Alarm Manager (online) available from http://developer.android.com/intl/ja/reference/android/app/AlarmManager.html (accessed 2013-04-21).

[13] Android API: Broadcast Receiver http://developer.android.com/reference/android/content/Context.html#sendBroadcast(android.content.Intent (accessed 2013-04-21).

[14] T. Konishi, T. Kamiyama, S. Kawasaki, H. Inamura, ``Reducing the use of Energy and Wireless Resources on Android Devices during Screen Inactive Periods,'' DPS Workshop IPSJ, pp. 249-256 (2012).

[15] S. Kawasaki, T. Kamiyama, S. Ohkubo, H. Inamura, ``A congestion avoidance method for event delivery mechanism,'' IPSJ SIG Technical Report Vol 2012-CDS-6, pp.1-8 (2012).

[16] Google Play: Advanced task killer. (online) available from https://play.google.com/store/apps/details?id=com.rechild.advancedtaskkiller&hl=ja (accessed 2013-04-21).

[17] M. Calder, and M. Marina, ``Batch Scheduling of Recurrent Applications for Energy Savings on Mobile Phones,'' Proc. of 7th Annual IEEE Communications Society Conference on Sensor Mesh and Ad Hoc Communications and Networks (SECON2010), pp. 1–3 (2010).

[18] V. Kononen, and P. Paakkonen, ``Optimizing Power Consumption of Always-On Applications Based on Timer Alignment,'' Proc ˆ of COMSNETS2011, pp.1-8 (2011).

[19] T. Yamamoto, S. Saruwatari, M. Minami, H. Morikawa, ``Piggyback Transport Protocol: Energy-efficient Upload Engine for Participatory Sensing,'' Journal of IPSJ, Vol.53, No.1, pp.274-285 (2012).

[20] E.J. Vergara, and S. Nadjm-Tehrani, ``Energy-aware Cross-layer Burst Buffering for Wireless Communication,'' Proc. of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet. e-Energy '12, pp.1-10 (2012).

(Received December 29, 2013)
(Revised March 26, 2014)

Hiroshi Inamura joined NTT DOCOMO, Inc. in 1998. His research interests include system research on mobile device and distributed system. Before DOCOMO, he was a research engineer in NTT labs since 1990. From 1994 to 1995, he was a visited researcher in the Department of Computer Science, Carnegie Mellon University. He received B.E., M.E. and D.E. degree in Keio University, Japan. He is a member of IPSJ, IEICE, ACM and IEEE.

Teppei Konishi His main research interests include energy-efficient task scheduling for Android OS. He received B.E. and M.E. degree in Osaka University, Japan. He is currently a research engineer at Research Laboratories, NTT DoCoMo Inc. He is a member of IPSJ.

Takeshi Kamiyama joined NTT DOCOMO, Inc. in 2006. His research interests include system research, especially energy-efficient design, on mobile device and distributed system. Before DOCOMO, he received MS degree in University of Tokyo, Japan in 2006. Also, he was co-founder and CEO in e-jis, Inc. from 2003 to 2006. He is a member of IPSJ.

Ken Ohta received the BE, ME, and DE degrees from Shizuoka University, Japan in 1994, 1996, and 1998, respectively. In 1999, he joined NTT Mobile Communications Network Inc. (NTT DOCOMO). His research interests include mobile computing, distributed systems, and system security. He is a member of the Information Processing Society of Japan and of the Institute of Electronics, Information and Communication Engineers.

Application of a Lump-sum Update Method to Distributed Database

Tsukasa Kudo[†], Yui Takeda[‡], Masahiko Ishino*, Kenji Saotome**, and Nobuhiro Kataoka***

[†]Faculty of Comprehensive Informatics, Shizuoka Institute of Science and Technology, Japan
[‡]Mitsubishi Electric Information Systems Corporation, Japan
* Department of Management Information Science, Fukui University of Technology, Japan
** Hosei Business School of Innovation Management, Japan
*** Interprise Laboratory, Japan
kudo@cs.sist.ac.jp

Abstract - At the present time, with spread of the internet business, many business systems have become to be built as distributed systems. Accordingly, the database is also dispersed to plural business systems as the distributed database. By the way, in the actual business systems, a lump-sum update of a great deal of data has often to be performed concurrently with the online transactions. However, in this case, there is a problem of efficiency in the conventional update methods, which update databases by the chain of many divided transactions. For this problem, we have shown the efficient lump-sum update method for centralized business systems, which use the transaction time. However, as shown by the distributed transactions, some transaction features for the distributed database are different from the centralized database. Therefore, in this paper, first we show the problems on applying this method to the distributed databases. Secondly, we propose their measures. Moreover, through the evaluations using a prototype, we confirmed that our measures are valid and this method can be applied to the distributed databases.

Keywords: database, distributed database, distributed transaction, business system, nonstop service.

1 INTRODUCTION

With spread of the internet business and decentralized technology, many business systems have become to be built as distributed systems at the present time. That is, each system performs its business by mutual cooperation with other systems [7], [11]. For example, in corporations, business systems are built at each branch office, and the business of this office is performed using its system. On the other hand, each business system is operated as a part of the distributed system, since each system accesses the data of the other systems when it is necessary. In this way, the distributed business system as the whole corporation is composed. And, it is possible to reduce the communication cost and perform the suitable system operation for the each office. On the one hand, as for the online services on a wide range of Internet such as net shops, non-stop services have become common because of the convenience of customers, the globalization and so on. So, it is difficult to stop the online service for the particular business.

However, in such a business system, a great deal of data often has to be updated in a lump-sum. So, formerly, it was executed as a night batch to avoid the time zone of the online service. However, because of the spread of nonstop service, it has to be executed concurrently with the online service at present. Therefore, some methods were put to practical use to execute it concurrently with the user entry of online service (hereinafter "online entry"). Here, these conventional methods divide the updating of a great deal of data into short time transactions, and then execute and commit them one after another. Therefore, though its impact on the online entry is small, there are problems: the intermediate results of the updating can be queried by the other transactions, that is, the isolation cannot be maintained; its processing efficiency declines because of the increase of its commit number.

For these problems, we had proposed an updating method that utilizes the records about the transaction time[2]. Moreover, we had shown the following evaluation result as for the updating of a great deal of data in the centralized systems: it executes the update more efficiently than the conventional method with maintaining the ACID property[3]. Since the transaction time is a kind of time of the temporal database, hereinafter we call it "temporal update" method. However, in order to apply this method to a distributed database, it is necessary to measure against not only the distributed transaction but also the various problems of the distributed environment.

Here, the temporal update has the characteristic of its process as follows: the completion time of updating is set beforehand; as for the commit of this method, its execution control has to be performed for the serialization between the online entries. However, there are problems: the dispersion in the update time is often increased by the efficiency of the network and cooperative business system environments; the synchronization between plural servers often causes the decline of efficiency. Therefore, in this paper, we propose a method for applying the temporal update method to the distributed databases. Moreover, we show the following experimental results of the prototype: the problems can be solved by the proposal method; since the implementation of the destination server of data transfer is easy, this method is valid for a data distribution system having many destination servers.

The remainder of this paper is organized as follows. Section 2 shows the related works about the update transaction; an overview of the temporal update method; the problem about applying it to the distributed databases. In Section 3, we propose the method for this problem, and show its implementation in Section 4. In Section 5, we show the experimental results of the prototype, and show our considerations in Section 6.

2 PROBLEM OF DATABASE UPDATE IN A LUMP-SUM

2.1 Related works

As for the update of the database, it is necessary that the transaction maintains the ACID properties: atomicity, consistency, isolation, and durability[1]. Here, since a lot of users use the online entry concurrently, a lot of corresponding transactions access the database concurrently. Therefore, database updates are serialized by locking the data to be updated by each transaction, and we can obtain a result as if transactions were executed sequentially. On the other hand, in the actual business systems, it is necessary to update a great deal of data in a lump-sum. For example, in the banking systems, the ATM is provided as the online service, and a lot of users perform online entry at the same time. On the other hand, a great deal of account transfer that is entrusted from the credit card company and so on is executed as the lump-sum update.

Since the update time of such a lump-sum is often so long, there is the problem that it can't lock the whole target data to serialize between the online entries. It makes the online entries wait for a long while. So, some methods were put to practical use to execute it concurrently with the online entries. In the mini-batch, a great deal of data is divided into small units, and they are updated and committed individually to shorten each update time. That is, the lump-sum update is performed by a set of short transactions. Also, in sagas, they compose a sequence of transactions and are performed one after another. It has a configuration to recover by executing the compensating transaction corresponding to each transaction in the case of fault [1], [10]. But, in these methods, the update and commit are repeated alternately many times. And, it makes the problems: the intermediate results of the updating can be queried by the other transactions; the efficiency declines because of the increase of commit number.

Here, in the distributed databases, the transaction needs not only the update and commit feature as for the individual database but also the feature for simultaneous update of multiple databases. So, the distributed transaction feature was put to practical use, in which the concurrency control across multiple databases is performed by two-phase commit and so on [6]. In this way, as for the distributed database, the different transaction feature from the centralized database has to be introduced.

By the way, the temporal database was proposed from the viewpoint of the record management about the time [8]. The transaction time is one of the times of this database: the time that a fact is valid in the database, which is expressed by the half-open interval $[T_a, T_d)$. Here, T_a shows the addition time that the data of the fact was added to the database; T_d shows the deletion time that it was deleted from the database. Even in the case of data deletion, data is deleted only logically by setting the deletion time, so the data record is left. Incidentally, if the data wasn't deleted, the value of attribute T_d is expressed by now [9]. It shows the current time and changes with passage of time. The relation R_t of the table having the transaction time is expressed below.

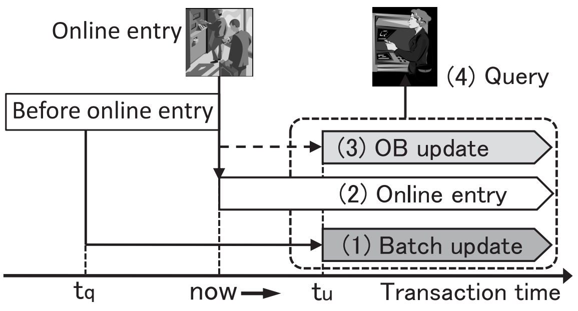


Figure 1: Change of data by temporal update method.

$$R_t(K, T_a, T_d, A) \tag{1}$$

Here, T_a and T_d are the above-mentioned transaction time. Let $q[T_a]$ be the value of attribute T_a of tuple q. Then the data set of the snapshot of R_t at the designated time t is expressed by the following $Q(t)$.

$$Q(t) = \{q|q \in R_t; q[T_a] \leq t \wedge t < q[T_d]\} \tag{2}$$

Here, K is the primary key attribute of the snapshot; A is the other attribute. The transaction time is not exposed to the users. Since the update of the online entry is performed at the current time $t = now$, the data of Equation (2) at any past time can be queried without the conflict with the online entry.

For a data update method, we can use the optimistic concurrency control [4] by utilizing the transaction time. The lump-sum updates are often performed by the following procedure: reading the target data; generating the update data from it; updating the database. In this method, the transaction confirms the transaction time of the target data again at the update timing. And, if it was not changed from the read timing, it shows the data was not updated by another transaction. However, the transaction needs to lock the data between this confirmation and the commit.

For another update method which utilizing a time, there is the timestamp-ordering concurrency control. It uses the time stamps (start time) of transactions $\{T_1, T_2, ..., T_n\}$, which compose the ordered set. The time stamp is stored in data when a transaction accesses the data, and the order of transactions that access to each data is maintained by it [4]. However, when one transaction is updating a data, the other transactions that update the same data have to wait until the commit.

Thus, these methods intend for short time transactions. For example, since the lump-sum update takes a long while, the other transactions are also waited for a long while. That is, it disturbs the non-stop services. Moreover, we can't find the lump-sum update method for a great deal of data with maintaining the ACID property.

2.2 Temporal Update Method

As for the centralized database, we proposed the temporal update method that utilizes the transaction time to update a great deal of data in a lump-sum with maintaining the ACID property of the transaction [2]. Figure 1 shows the data change with the transaction time in the case of the database

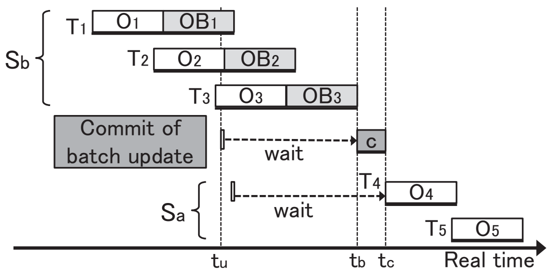


Figure 2: Serialization of batch update and online entry.

update by this method. In this method, the lump-sum update is executed by both the batch update (1) and OB update (the online batch update) (3). Here, the batch update (1) corresponds to the usual lump-sum update. It updates the data at the past time t_q, and stores the update results at the future time t_u that was "beforehand" established. Incidentally, though the online entry (2) is executed even during the batch update (1), it updates the data at the current time now. So, the competition between (1) and (2) can be avoided. On the other hand, since the batch update also has to update the data changed by the online entry, the OB update performs the process corresponding to it individually.

As a result, at time tu, three kinds of results are stored: the batch update, the online entry and the OB update. Therefore, we query only the high-priority data by the following order of priority, and we can get the query result as if the batch update and online entry were performed in a series.

(A) First, for each value of the primary key K of Equation (1), we select the data that was updated at the latest time. Here, we use t_q to the updated time as for only the batch update.

(B) Second, if there are plural data having the same key value, we select the data by the following priority of update process: the OB update, the online entry, the batch update.

In the case of Fig. 1, since both results of the OB update and online entry were updated at the latest time, the OB update result is queried based on above-mentioned (B). Also, if there is no online entry as for the case of Fig. 1, there are the online entry data entered before t_q ("Before online entry" in the Figure) , and the result of the batch update. So, the batch update result is queried based on above-mentioned (A). That is, in this method, all the updated data are stored, and only the valid data is queried. As a result, we can perform the lump-sum update and the online entry concurrently, without their competition.

However, at the end of a batch update, the control for the serialization between it and the online entry is necessary. That is, the online entry, which is begun before time t_u, is accompanied by the OB update to reflect the batch update. On the other hand, another one begun after t_u has to be performed using the result of the batch update. We show this in Fig. 2. Therefore, in this method, the commit of the batch update is executed after the online entries (S_b) that are begun before time t_u; the other online entries (S_a), which are begun after t_u, are waited until this commit.

2.3 Problem about Application to Distributed Database

As for the temporal update method, we had been assuming the centralized database and the controlled operation. That is, as mentioned in Section 2.2, we assumed that the batch update time t_u can be established beforehand; the individual online entry completes in a short time. In the actual systems, the former feature is often used in the lump-sum update executing at the designated time: the bank transfer at the designated time; the change of organization data at the designated date; and so on. However, as for the case where we apply this method to the distributed database, it has to be executed in the various environments. So, the following problems occur.

First, the dispersion of the batch update time is very larger than the centralized database. Though the batch update updates plural databases at the same time, the individual environment in this distributed system is varied: the traffic on the network; the load and performance of each server. So, the prediction of the time t_u is extremely difficult. Therefore, in the case that the prediction time is earlier than the actual elapsed time, the batch update doesn't complete by t_u. So, it is aborted, and it has to be re-run. On the contrary, if it is later, the unnecessary OB update must be continued after the batch update completion. As a result, there is a problem that the efficiency as the whole system declines.

Second, the problem that the online entry wait of one server spreads to the other servers occurs. As for the distributed database, the related online entries executing in all the servers have to wait the completion of the commit of the batch update until t_c as shown in Fig. 2. However, since the systems were built individually at each branch office, there may be the online entries having a long time transaction. Therefore, there is a problem about the delay of the commit of the batch update, which is caused by some online entry. And, it delays the online entries in the all related servers.

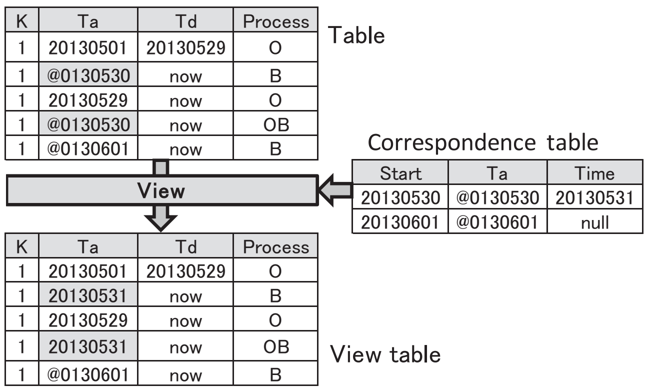
3 PROPOSAL METHOD FOR DISTRIBUTED DATABASE

To solve the problem for applying the temporal update to the distributed database, we propose the following two methods.

3.1 Setting Method of Dynamic Batch Update Completion Time

For the problem about the elapsed time of the batch update, we propose the method to perform its commit immediately after the batch update. Here, as shown in "Table" of Fig. 3, since the predicted completion time is set to every updated data by the batch update and OB update, it takes time to update these time again. Therefore, in this method, we use a view table to change these times when these data are queried.

Figure 3 shows the overview of this method. Here, time is shown as date: year, month and day. The predicted completion time of the batch update is set as the temporary time, and its first digit is replaced by "@" as an example. In the case



Table

K	Ta	Td	Process
1	20130501	20130529	O
1	@0130530	now	B
1	20130529	now	O
1	@0130530	now	OB
1	@0130601	now	B

Correspondence table

Start	Ta	Time
20130530	@0130530	20130531
20130601	@0130601	null

View table

K	Ta	Td	Process
1	20130501	20130529	O
1	20130531	now	B
1	20130529	now	O
1	20130531	now	OB
1	@0130601	now	B

Figure 3: Change of addition time of business data.

of Fig. 3, the addition time T_a, which is equal to the completion time, is set to "@0130530" and "@0130601". Also, its "Process" column shows the classification of the update process of the data: O (the online entry), B (the batch update) and OB (the OB update). In this case, the data entered by the online entry on 5/1 of 2013 was updated by the batch update on 5/29, which above-mentioned completion time (T_a) is "@0130530". And, the OB update was performed along with this. In addition, the batch update with the completion time "@0130601" was performed after this.

When the batch update completed, its completion time is set ("Time" of "Correspondence table"), which corresponds to the temporary time T_a. Incidentally, it is set to $null$ until this completion. Only if the the completion time is set to "Time", T_a of data of "View table" is replaced with it. In the case of Fig. 3, "@0130530" of this is replaced with "20130531"; "@0130601" isn't replaced.That is, even if there are many target data, all T_a of them can be replaced by updating only one data. Therefore, this update process can be executed by a short time transaction. Incidentally, since the value of "@" is larger than any numerical value, the data having the temporary time (with "@") isn't queried by Equation (2) with designating time.

3.2 Serialization Method without Lock Feature

To solve the problem shown in Fig. 2, we propose the serialization method between the online entry and batch update. In (A) and (B) of the Fig. 4, we show the case that the online entry is executed across the batch update completion time t_u. Incidentally, (C) shows the case that there is no online entry, as the reference. Here, the black hatching shows a transaction of the online entry; the broken line shows the data that should not be queried though it is exists.

First, (A) of Fig. 4 shows the success case of the online entry. Transaction (A) continues across the batch update completion time t_u. If the batch update result $(a-1)$ is queried between time t_u and the completion time of this transaction, it becomes a phantom read. Therefore, the mechanism, which makes this data not to be queried, is necessary to maintain the consistency. Incidentally, after the transaction completed, data (a) is deleted logically by setting the deletion time; the

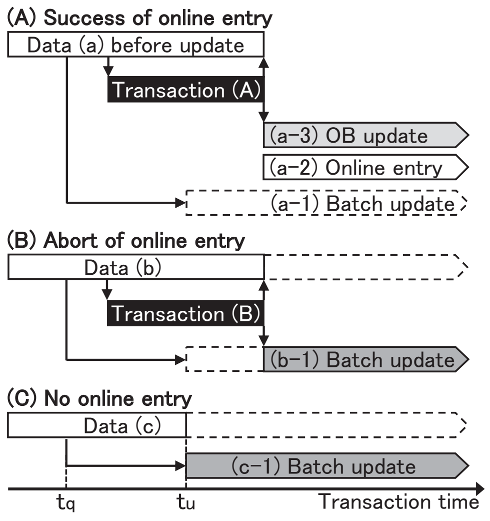


Figure 4: Query result as for online entry spanning t_u.

online enrty data $(a-2)$ is inserted; the OB update data $(a-3)$ is inserted if data (a) was the target of the batch update. So, $(a-1)$ is not queried, though either $(a-2)$ or $(a-3)$ is queried.

Next, (B) of Fig. 4 shows the abort case of the online entry. Transaction (B) continues across the batch update completion time t_u similar to (A), and its result is undecided at t_u: success or abort. Therefore, as for data $(b-1)$, the mechanism similar to (A) is necessary. In the case of the abort, data (b) remains without change, since the rollback of the transaction is performed. And, since the batch update was executed, its result $(b-1)$ has to be queried after the completion of the transaction. That is, for the serialization, the temporal update has to be composed to obtain the following query result.

- **During the online entry:** the batch update result is not queried, but the data just before the start of the online entry transaction is queried.
- **After the online entry completion (success):** either the online entry result or the OB update result is queried. Incidentally, in only the case of this data being target of the batch update, the latter occurs.
- **After the online entry completion (abort):** the batch update result is queried.

Therefore, in the case that the online entry transaction continues across the batch update completion time t_u, the consistency of the query result is maintained by the mechanism: the batch update result is not queried until the completion of the transaction. Therefore, we propose a method using a table that manages the key of the data being updated by online entry transactions. And the data, which key is registered in this table, is excluded from the query result of the batch update. This table has the following relation.

$$R_e(t\_name, t\_key) \tag{3}$$

Here, t_name shows the name of the target table; t_key shows the value of the primary key of the online entry data.

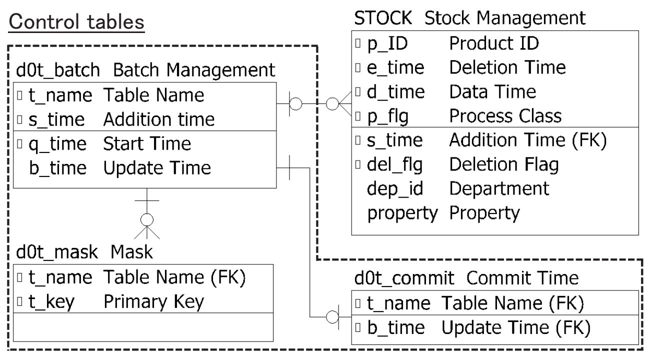


Figure 5: ER diagram of control tables.

And, (t_name, t_key) composes the primary key of this table. In this way, it is possible to perform the serialization by the table R_e without the lock feature between the onlien entry and batch update.

4 IMPLEMENTATION

First, Fig. 5 shows the ER diagram of the Control tables, which are used for the implementation of the proposal method. These tables save the data to control the temporal update and are placed in each target database of the temporal update. Batch management table $d0t\_batch$ manages the time of the temporal update, and it saves the following data: Table name t_name; Addition time s_time, Start time q_time; Update time b_time. Here, b_time is used to set the completion time of the batch update at the timing of its completion dynamically as shown in Fig. 3. That is, if the estimation of the completion time of the batch update is difficult beforehand, a temporary time and null are set to each s_time and b_time at the timing of its beginning. Then, the completion time t_u is set to b_time at the timing of its completion to query the updated data having the addition time t_u. Also, Mask table $d0t\_mask$ is an implementation of the relation of Equation (3). Commit time table $d0t\_commit$ stores the latest completion time t_u in b_time for each target table which name is set to t_name. And the result of the batch update and OB update which s_time is before b_time is queried.

Stock Management table $stock$ is an example of the business data table, and we use it for the experiment in Section 5. Here, as for the stock management, various kinds of distributed databases are used. For example, in the retail companies, each branch has its database system and manages its stock. On the other hand, the database serves as a part of the distributed database in the case of the stock movement among branches or the delivery from the distribution sector. Similarly, in the manufacturing industry, the supply chains of the parts are built among the related companies.

As for the attributes of $stock$, the following are correspond to the attributes (K, T_a, T_d, A) of the relation R_t shown in Equation (1): Product ID p_ID; Addition Time s_time; Deletion Time e_time; Department dep_id and Property $property$. In addition, we add the following properties for the temporal update. Data Time d_time shows the update order of the data as shown in Section 2.2; Process Class p_flg show the classification of the update process of the data shown as "Process" column in Fig. 3; Deletion Flag del_flg shows the data was deleted if it is set. Here, del_flg is used to exclude all data having the same primary key K from the query result. For example, in the case that the OB update (3) is deletion in Fig. 1, the other data (1) and (2) also not have to be queried. For this purpose, we exclude the unnecessary queried data (3) after the query, and as a result, the other data is also not queried.

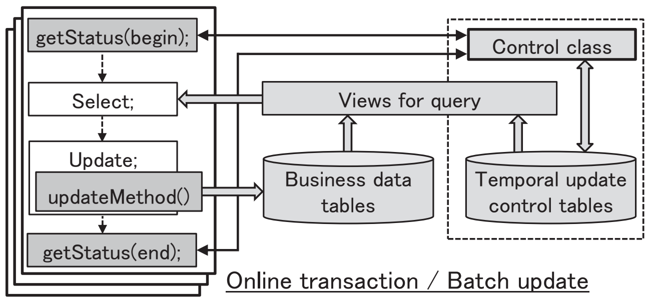


Figure 6: Composition of temporal update.

Second, the concurrency control between the online entries and batch update is necessary as shown in Fig. 4. For this purpose, we implemented the Control class by the Java to perform this control as shown in Fig. 6. It exposes a method $getStatus$ that is the interface of business programs, and the programs call this method at the timing of the start and completion each of the online entry transaction and batch update. Then, the updating of the control tables and the control about the serialization are performed by this method.

In the case of the batch update, $getStatus(begin)$ is called at the timing of start to set the control data of the batch update to $d0t\_batch$ of all related databases. $getStatus(end)$ is called at the timing of its completion similarly, and it sets the completion time to b_time of $d0t\_batch$ and updates $d0t\_commit$. Thus, the batch update result can be queried. In the case of the online entry, $getStatus(begin)$ is called at the timing of its transaction start similarly, and confirms whether or not the batch update updates the target table. If it is not being updated, only the usual online entry transaction is performed. On the contrary, if it is being updated, the transaction's data is set to $d0t\_mask$ by this method. And, the transaction performs the OB update after the online entry, and calls $getStatus(end)$ to delete the data of $d0t\_mask$ at the timing of its completion.

Third, The business programs query the table through its view table, if it is the target of the temporal update. The view table is created by the DDL (Data Definition Language) shown in Fig. 7 and has the following feature.

(a) It changes the addition time T_a of the view table as shown in Fig. 3.

(b) It controls query results of the batch update and OB update by using the time changed in (a): only the data, which Addition Time s_time is older than Update Time b_time of $d0t\_commit$ (including b_time), is queried. Incidentally, every online entry result is queried.

(c) It excludes the data from the query result of (b) by using

```
CREATE VIEW stock_v AS
SELECT p_id, COALESCE(b_time, a.s_time, b.s_time),
   e_time, d_time, p_flg, del_flg, dep_id, property FROM stock a
LEFT OUTER JOIN d0t_batch b USING (s_time)
WHERE p_flg= '1'
   OR (p_flg = '0' AND p_id NOT IN
      (SELECT t_key FROM d0t_mask
         WHERE t_name = 'stock' AND q_time = a.d_time)
      OR p_flg = '2')
   AND COALESCE (b_time, a.s_time, b.s_time) <=
      (SELECT b_time FROM d0t_commit
         WHERE t_name = 'stock');
```

Figure 7: Example of DDL to create view table.

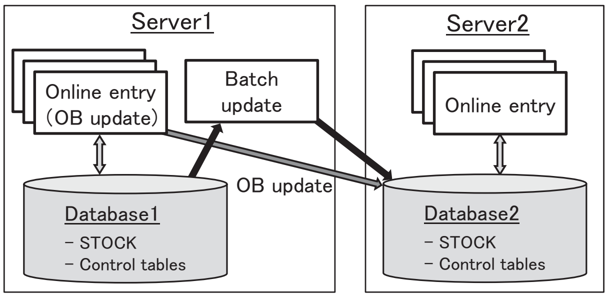


Figure 8: Experimental system composition.

$d0t\_mask$, which is being updated by the online entry transaction.

Using these view table, there are the following effects: since it doesn't allow to query the intermediate state of the temporal update shown in Fig. 4, the ACID properties of the database can be maintained; the developer can build the program efficiently by it, since it conceals the above-mentioned procedure.

On the other hand, these view tables aren't always updatable. So, though it is necessary that the table is updated directly by the business programs, transaction time attributes of the table are not exposed to users: e_time, d_time and s_time of $STOCK$. Therefore the method $updateMethod$ is provided, and the business programs update tables using this method. In addition, the above-mentioned features about databases were implemented by MySQL: InnoDB for the transaction feature; XA transaction for the distributed transaction [5].

5 EXPERIMENTS AND EVALUATIONS

5.1 Overview of Experiments

To evaluate the proposal method, we built the prototype of a stock management system, and performed experiments by it. We show the composition of the experimental system in Fig. 8. This system is a distributed system consisting of two servers, and each server has its own database. Its business table is the stock management table shown in Fig. 5. And, we assume this system is a non-stop service system. That is, its data can be deleted by the online entry transactions at any time when the corresponding goods are sold.

Data	Time (Sec)								Remarks
	2	4	6	8	10	12	14	16	
	←Batch update→								
Server1 (Local)									
R010	●	●	●						Sold
R011	●	●	●	●	●				Sale canceled, Move out
R020	●	●	●	●	●	●			Sold
R021	●	●	●	●	●	●			Sale canceled, Move out
R100	●	●	●	●	●				Move out
Server2 (Remote)									
R010									
R011						●	●	●	Move in
R020									
R021							●	●	Move in
R100						●	●	●	Move in

Figure 9: Query result of proposed temporal update.

The stock movements from $Database1$ to $Database2$ are performed in a lump-sum when necessary. This is built by using the temporal update method, and its batch update results, which is shown by (1) in Fig. 1, are inserted to $stock$ of each database. Among them, as for the data of $Database1$, Deletion Flag del_flg is set not to be queried after the completion of the temporal update. On the one hand, the data is inserted into the table of $Database2$, and they can be queried after the completion time. In addition, in the case that the data of $Database1$ is deleted by the online entry, its movement to $Database2$ has to be also canceled by the OB update. Concretely, as for $Database2$, the data with setting del_flg is inserted by the OB update, so the batch update results are also excluded from the query result.

In the experiment, we performed the online entry in both of the server, and performed the above-mentioned temporal update on its way. In Fig. 9, we show the query result of the typical data along the passage of time. Among the data, $R010$ was sold during the data movement by the temporal update; the sale of $R011$ was canceled, that is, the transaction was canceled by the rollback. As for $R020$ and $R021$, the sale of them continued across the completion time of the data movement, and $R020$ was sold; the sale of $R021$ was canceled. $R100$ was only moved. Also, the black circle shows that the data was queried at the time through the view table in Fig. 7; the blank shows not to be queried;

5.2 Evaluations of Validity of Proposal Method

First, in order to complete the temporal update immediately after the batch update, we set the temporary time "@0" (we show only its second, the same in the following) at the timing of the batch update start. Then, we set its completion time "11" to Update Time b_time of both of $d0t\_batch$ and $d0t\_commit$ at the timing of the batch update completion. As shown by $R011$ and $R100$, we could complete the temporal update immediately after the batch update and query their update result.

Next, for the serialization between the online entry and batch update without the lock feature, we inserted $d0t\_mask$ the corresponding data to the online entry at the timing of its start; we deleted the data at its completion. As a result, during online entry, the batch update results are not queried, but the data before the online entry is queried as shown by $R020$ and $R021$. That is, as for the sold data $R020$, the data before its selling is queried until the completion of the online entry; it can't be queried after the completion. Also, it didn't be moved to $Database2$. As for $R021$ in the case of sale being canceled, since it was moved at the timing of the completion of the online entry transaction, it can't be queried in $Database1$ after the completion; it can be queried in $Database2$. As described above, the temporal update results are queried after the online entry transaction, that is, the serialization between them could be controlled.

5.3 Evaluations of Implementation in Distributed Environment

As for the data movement by the temporal update, its business programs in the destination server could be composed by only local transactions. That is, as shown in Fig. 8, as for Stock Management Table $stock$ in $Database2$, since the existing data in another database is only inserted by both of the batch update and OB update, there is no competition with the online entry. Also, since Control tables are used only inside of the view table and hidden from the above-mentioned programs, these tables could be implemented without influences on the existing programs of $Server2$ except the implementation about the view tables.

On the other hand, as for the business program in the server where the temporal update is performed, the implementations about it were necessary. As for the updating of Control tables, by developing the control class for the temporal update, the business programs could be configured to call its methods at the timing of start and end of the online entry transactions as shown in Fig. 6. So, these tables are hidden from the business programs. However, since the OB update has to be executed as a part of the online entry transaction, it had to be integrated into the corresponding online entry program. By the way, as for the implementation of this method, since the data of $d0t\_mask$ has to be inserted before the online entry transaction start, the transaction of this method had to be executed separating from it. Similarly, in the case of abort, since the rollback of the online entry transaction is executed, the data of $d0t\_mask$ had to be deleted by another transaction. In addition, in the case of success, the deletion could be executed in the same transaction.

As for the batch update, the serialization control between the temporal update and the online entry transactions had to be executed at the start and end of the transaction. On the other hand, since this control wasn't necessary except these timing, the business tables of each database could be updated by the local transactions one after another. In particular, in the case of the number of the updating data was small, its commit was not necessary in the middle of the updating. So, the updating features could be implemented by only using SQL statement without the cursor operations, and we could implement it more efficiently than the mini-batch.

6 CONSIDERATIONS

We found that the temporal update can be completed immediately after the completion of the batch update by this method. As for the temporal update, since the OB update has to be executed as a part of the online entry transaction, its execution time is considered to become long particularly in the distributed environment. Therefore, from the viewpoint of the efficiency, to reduce the execution time of the temporal update is effective.

Incidentally, we consider that the method to reserve the completion time of the temporal update is also useful in both of the centralized and distributed systems. It can be used in the case to change great deal of data in a lump-sum at the designated time and so on. For example, we discuss the case that the share of product of each branch is changed at the prearranged date in Fig. 8. In this case, we execute the temporal update with reserving its completion time at 0 a.m. of the designated date and the stock of each branch office can be updated at once with reflecting the sale earlier.

Also, we found that the serialization between the temporal update and online entry can be composed without the lock feature. It has been pointed out that the long time transactions and the lock feature cause the fault in the distributed systems. So, we consider that the lump-sum update can be composed more secure by this method.

Moreover, in the case of the data movement from one server to another server, we found that we need to implement only the control tables and the view tables as for the latter; the existing business programs are not affected except the implementation about the view table. In particular, even the lock feature for update isn't necessary, because the lump-sum update can be executed by only the insertion of data. That is, this method is valid in the case of data transfer from the specified administration server to the other servers widely.

7 CONCLUSIONS

In IWIN2012, We showed the temporal update method in a centralized database is effective in the viewpoint of maintaining the consistency and updating data efficiently. However, to apply this method to the distributed database, there are problems: it is difficult to estimate its completion time; an online entry wait is spread to the other servers at the completion timing of this method, because the serialization between the batch update and the online entry transactions has to be performed. For these problems, we propose the method in this paper for the following purpose: the beforehand estimation of its completion time becomes unnecessary; the serialization can be executed without the online entry waits. And, we confirmed by experiments that these feature was valid and could be implemented in the distributed database. Moreover, we find through these experiments this method is also valid in the case of data transfer from the specified administration server to the other servers in a wide area.

The future challenge is the evaluation of the operational efficiency and performance in the viewpoint of the business system in order to adopt it to the actual distributed systems.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number 24500132.

REFERENCES

[1] J. Gray, and A. Reuter, "Transaction Processing: Concept and Techniques," San Francisco: Morgan Kaufmann (1992).

[2] T. Kudo,, et al., "A batch Update Method of Database for Mass Data during Online Entry," Proc. of 16th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES2012), pp. 1807–1816 (2012).

[3] T. Kudo, et al., "Evaluation of Lump-sum Update Methods for Nonstop Service System," Proc. of Intenational Workshop on Informatics (IWIN2012), pp. 3–10 (2012).

[4] P.M. Lewis, A. Bernstein, and M. Kifer, "Databases and Transaction Processing: An Application-Oriented Approach," Addison-Wesley (2001).

[5] ORACLE, XA Transactions, http://dev.mysql.com/doc/refman/5.1/en/xa.html.

[6] M.T. Özsu, and P. Valduriez, "Principles of Distributed Database Systems," Springer (2011).

[7] U. Shanker, ,M. Misra, A.K. Sarje, "Distributed real time database systems: background and literature review," Distributed and Parallel Databases, Vol. 23, Issue 2, pp. 127–149 (2008).

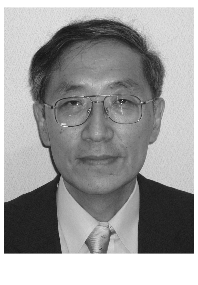
[8] R. Snodgrass, and I. Ahn, "Temporal Databases," IEEE Computer, Vol. 19, No. 9, pp. 35–42 (1986).

[9] B. Stantic, J. Thornton, and A. Sattar, "A Novel Approach to Model NOW in Temporal Databases," Proc. of 10th International Symposium on Temporal Representation and Reasoning and Fourth International Conference on Temporal Logic, pp. 174–180 (2003).

[10] T. Wang, J. Vonk, B. Kratz, and P. Grefen, "A survey on the history of transaction management: from flat to grid transactions," Distributed and Parallel Databases, Vol. 23, Issue 3, pp. 235–270 (2008).

[11] J. Yang, I. Lee, O. Jeong, S. Song, C. Lee, and S. Lee, "An architecture for supporting batch query and online service in Very Large Database systems," Proc. of IEEE International Conference on e-Business Engineering (ICEBE '06), pp. 549 – 553 (2006).

(Received December 2, 2013)
(Revised March 4, 2014)



Tsukasa Kudo received the M. Eng. from Hokkaido University in 1980 and the Dr. Eng. in industrial science and engineering from Shizuoka University, Japan in 2008. In 1980, he joined Mitsubishi Electric Corp. He was a researcher of parallel computer architecture, an engineer of application packaged software and business information systems. Since 2010, he is a professor of Shizuoka Institute of Science and Technology. Now, his research interests include database application and software engineering. He is a member of IEIEC, Information Processing Society of Japan and The Society of Project Management.

Yui Takeda received the B.E. from Keio University, Japan in 1987. In 1987, she joined Mitsubishi Electric Corp. She was an engineer of artificial intelligence and application software. Since 2001, she joined Mitsubishi Electric Information Systems Corp. Now, she manages intellectual property rights.

Masahiko Ishino received the master's degree in science and technology from Keio University in 1979 and received the Ph.D. degree in industrial science and engineering from graduate school of Science and technology of Shizuoka University, Japan in 2007. In 1979, he joined Mitsubishi Electric Corp. Since 2009, he is a professor of Fukui University of Technology. Now, His research interests include Management Information Systems, Ubiquitous Systems, Application Systems of Datamining, and Information Security Systems. He is a member of Information Processing Society of Japan, Japan Industrial Management Association and Japan Society for Management Information.

Kenji Saotome received the B.E. from Osaka University, Japan in 1979, and the Dr. Eng in Information Engineering from Shizuoka University, Japan in 2008. From 1979 to 2007, he was with Mitsubishi Electric Corp., Japan. Since 2004, he has been a professor of Hosei business school of innovation management. His current research areas include LDAP directory applications and single sign-on system. He is a member of the Information Processing Society of Japan.

Nobuhiro Kataoka received the master's degree in electronics from Osaka University, Japan in 1968 and the Ph.D. in information science from Tohoku University, Japan in 2000. From 1968 to 2000, he was with Mitsubishi Electric Corp. From 2000 to 2008, he was a professor of Tokai University in Japan. He is currently the president of Interprise Laboratory. His research interests include business model and modeling of information systems. He is a fellow of IEIEC.

Proposed Integration Framework for Viewpoint-based Enterprise Architecture

Akira Tanaka[†], and Osamu Takahashi [‡]

[†]view5 LLC, Yokohama, Japan
[‡] Department of Media Architecture, Future University Hakodate, Japan
atanaka@view5.co.jp, osamu@fun.ac.jp

Abstract – Enterprise Architecture needs to cover various aspects of the target enterprise, starting from business strategy down to communication protocols. However, business environment changes daily, and new technologies are constantly introduced. There is a need to define technology integration framework for system architecture such as Enterprise Architecture. This paper presents a proposed mechanism for integrating technology elements, which did not exist when the architecture was introduced, such as mobile computing, cloud computing, and social network, into Viewpoint-based Enterprise Architecture. Specifically, the mechanism is targeting for RM-ODP viewpoint language based Enterprise Architecture, which includes modification approach to meta-model and UML Profile. Findings and discussion covers integration approaches, relationship with model driven software development, and openness or interoperability of Enterprise Architecture.

Keywords: Enterprise Architecture, RM-ODP, Mobile Device, Cloud Computing, Social Networks, UML

1 INTRODUCTION

1.1 Enterprise Architecture

Enterprise Architecture is a widely recognized term for designing enterprise's software and system architecture. Examples include Zachman Framework [1][2], TOGAF [3], and Federal Enterprise Architecture (FEA) [4]. In standards domain, RM-ODP [5][6][7] is an ISO/IEC/ITU-T standard for specifying Open Distributed Processing. It provides foundational concepts and standard viewpoints including languages and structuring rules for specifying enterprise systems. The standard viewpoints in RM-ODP are Enterprise, Information, Computational, Engineering, and Technology. Other frameworks use different classification: Perspectives in Zachman Framework, Architecture Domains in TOGAF, and sub-architecture domains in Federal Enterprise Architecture. Because of its neutrality, openness, availability of tools and documents, we chose RM-ODP to represent Enterprise Architecture for the research described in this paper.

One of the issues with Enterprise Architecture Framework is they are designed to be stable, which is good but at the same time that may imply hard to modify architectural elements or hard to introduce new architectural elements. Therefore, there is a need to study how Enterprise Architecture can be extended in consistent way. The proposed mechnism involves an approach to consistently modify Enterprise Architecture's or RM-ODP's meta-model and corresponding UML Profile.

1.2 Modeling Software Architecture

Widely used modeling language for specifying software systems is UML or Unified Modeling Language [12]. It provides a means for defining structural and behavioral aspects of the target system, using its modeling elements such as Class, Component, Activity, State Machine, and Use Case. Although it is a general purpose modeling language, with its profiling mechanism, users can define customized model elements for specific domains such as embedded systems, real-time systems, and Enterprise Architectures. Example such profiles covering Enterprise Architecture domain include UML Profile for DoDAF and MODAF [20] and Use of UML for ODP system specifications [8][9]. It usually starts with defining meta-model for the Enterprise Architecture concepts and relationships among them, followed by defining its UML profile for use with UML tools. This approach is also applicable to our case, which is to introduce meta-model elements for recently adopted technologies, define UML Profiles for it, and integrate them into the base Enterprise Architecture Framework.

2 RECENT TECHNOLOGIES' IMPACT ON ENTERPRISE ARCHITECTURE

Growing popularity of mobile devices such as smart phones and tablets, cloud computing, and social networks are examples of the changes we have witnessed recently. Although those are just examples, those technologies were not really considered in Enterprise Architecture ten years ago, but we need to integrate them today. We will first examine kinds of impact those technologies bring to Enterprise Architecture.

2.1 Mobile Devices

Mobile devices at early days were mainly portable laptop PCs and communication means were very limited.

Reason for success may include Moore's law, people's welcome of mobile devices (actually computer systems), and acceptance of living in cyber world (emails, social networks etc.).

Impact on Enterprise Architecture includes addition and processing of new data elements like moving object's location, position, direction and time-stamp. Additional work of adjusting UI for mobile devices and increased security concerns e.g. by stolen devices or less secure network is also part of the impact. Mobility may be applied to people and things, such as software elements and hardware elements, specified in Enterprise Architecture. The impact would be to all the viewpoints.

2.2 Cloud Computing

Application Service Provider or ASP might be conceptually close to today's SaaS (Software as a Service). PaaS (Platform as a Service) equivalent did not exist, and rental server services or hosting services of the time could be considered as services similar to IaaS (Infrastructure as a Service) with limited capability.

Reason for success may be that cloud computing usually provides better ROI of IT resources than internal investment in hardware, software, system administration, training, and system development.

Impact on Enterprise Architecture includes increased use of external application services (SaaS), and external platforms to run the applications (IaaS and PaaS). There are various types of clouds: public cloud, private or internal cloud, hybrid cloud, and personal cloud. NIST's definition of cloud computing [10] is widely accepted. However, the use of external resources requires strong governance over security and regulatory issues. The impact would be to enterprise viewpoint for external applications and to computational/engineering/technology viewpoints for integration with this technology.

2.3 Social Network

Except for forums provided by e.g. online service providers, there was nothing comparable with today's social networks in popularity and scale.

Reason for social networks' success may be that owners of enterprise systems have looked at success cases in consumer oriented social networks and realized the importance of people aspects, involving e.g. managers, developers and operators, who may not be communicating with each other within the enterprise system. It is, therefore, logical to consider including this capability into Enterprise Architecture based systems and expect "social effects."

Impact on Enterprise Architecture includes adding and processing new data elements (social profile and people oriented network information), which leads to a new class of applications that enables posting, reading, reacting to messages to construct his/her social networks and analyzing social network [11], in the context of enterprise systems. The impact would be to all the viewpoints.

Those three elements, mobility, cloud computing, and social network, may have the following use relationships or dependencies (Fig. 1).

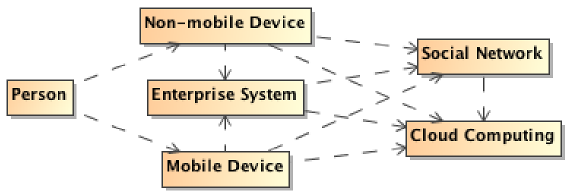


Figure 1: Relationship/Dependencies between mobile devices, cloud computing, and social networks

Mobility enhances access capability to a part of enterprise system running on cloud platform, cloud computing's scalability and reliability supports dynamic traffic changes of social networking, and mobile access to social network will accelerate the use of social network.

3 ANALYSIS AND PROPOSED MODIFICATIONS ON RM-ODP

We analyze these three technology areas and consider meta-models or UML Class diagrams representing MOF [13] model, consisting of their core concepts and its relationships among them. Those meta-models with RM-ODP's meta-models can be used to define integrated meta-models, which can be used for UML Profile development. Note that some concepts from three areas may already be defined in RM-ODP (or Enterprise Architecture), and we will use existing concepts with possible modifications when needed. Other elements are assumed to be independent and to be integrated into RM-ODP meta-model as domain-specific extensions.

3.1 Mobile Devices

In case of mobile devices, mobility elements were not really present in the most Enterprise Architectures. Some did have location concept, but it was not introduced to represent moving object's location. It was more for static location e.g. person's address or department's address such as "Stockholm, Sweden." In order to add mobility to an object, it needs to have a dedicated attribute for mobility or a link to mobility. Mobility attribute or mobility will consist of time-stamp and location. If necessary, velocity of moving object can be computed using this record ($v=\Delta location/\Delta t$).

Modifications to RM-ODP are the followings.

Enterprise/Computational/Engineering/Technology Languages: Optional link to Mobility added to Viewpoint Objects

Information Language: No change [since information models do not change depending on time and location]

As proposed changes to RM-ODP concepts, Mobility can be defined as a combination of LocationInTime and LocationInSpace (Fig. 2, fragment of the modified meta-model), both are defined terms in RM-ODP. Geographic Information standard could be used for LocationInSpaceType.

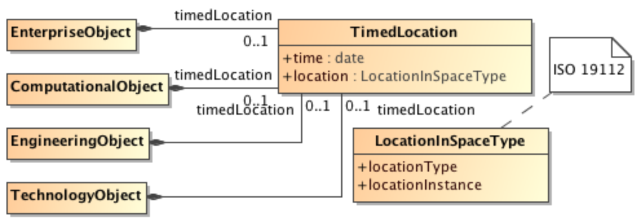


Figure 2: TimedLocation

3.2 Cloud Computing

In case of cloud computing, especially in business domain, the major concerns are actually on business execution, and underlying technologies such as computing platform are usually considered as engineering issues. However, use of cloud computing based external services can become an impact. When they are incorporated into business processes, they will work as action/activity or step implementations within certain business processes. Those external cloud applications usually provide services using web interfaces. Therefore, to incorporate external services, support of web interfaces or Service-Oriented Architecture (SOA) [14] will be needed in the architecture. In addition, integration with existing systems (legacy systems) could also be done in the same fashion. Interaction with external services, though, will require its policy to cover agreement for using external services and/or regulatory restrictions.

Modifications to RM-ODP are the followings.

Enterprise/Computational/Engineering/Technology Languages: Optional link to CloudService was added to Objects, and a new datatype “CloudServiceType” was introduced.

Information Language: No change [since information models do not change depending on where and how they are managed]

As proposed changes to RM-ODP concepts (Fig. 3, fragment of the modified meta-model), CloudNature can be defined as a combination of CloudSupport (Boolean) with CloudType ((Public, Private, Hybrid) & (SaaS, PaaS, IaaS)). Also in Enterprise Viewpoint, a Step may be labeled with CloudNature, showing that an Object with Cloud Support is supporting the Step.

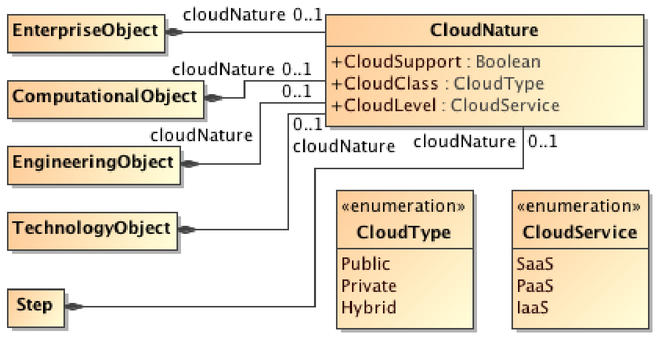


Figure 3: CloudNature

3.3 Social Network

When a person starts working for an enterprise, he/she will be given a title or role in an organization, which is connected to what the person is obligated to do, allowed to do, and prohibited to do, or job description. This model needs to be modified to incorporate sociality, which is described in the person’s social profile (Fig. 4, fragment of the modified meta-model). A new data types such as person’s interests, experience, and participating social communities with roles within need to be there. Person is usually modeled as Party, which can have a relationship with other Parties. RM-ODP’s Community concept is a good fit to represent social community. The resulting architecture will include parties, services, processes, etc. within the enterprise just like normal business processes to make best use of people’s capability.

Modifications to RM-ODP are the followings.

Enterprise Language: Optional link to SocialProfile with SocialRelationship were added to Party, and definition of SocialProfile was added.

Information/Computational/Engineering/Technology Language: No change

We use Party as defined in RM-ODP and introduced SocialRelationship, Social Profile, and Social Community, which is a subclass of Community. We can also use suitable Viewpoint Language elements.

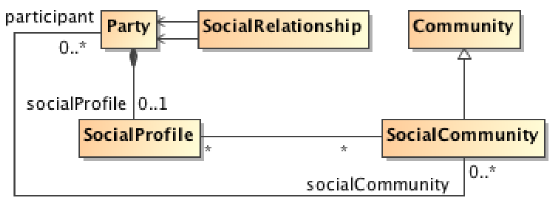


Figure 4: SocialProfile

4 UML PROFILES

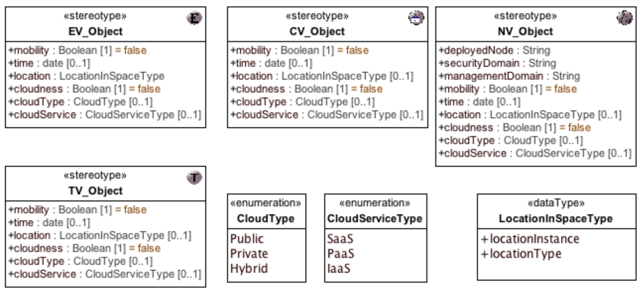
The changes to conceptual model need to be reflected in the UML Profile. There is an ISO/IEC/ITU-T standard called “Use of UML for ODP system specifications” or UML4ODP for short, with which ODP models can be created with UML tools.

There are two ways for modifying UML Profile. First is to modify the existing stereotypes, and the second is to define new stereotypes such as MobileObject and CloudObject by customizing existing stereotypes. The latter will lead to a creation of many subclass stereotypes such as NV_MobileBinder and NV_CloudBinder. In order to avoid the confusing too-many stereotypes, we chose modifying existing stereotypes approach.

The following shows the mapping of modified meta-model to UML Profile.

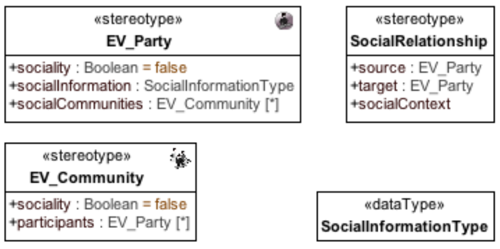
4.1 Mobile Device and Cloud Extension as UML Profile

Each viewpoint object’s stereotype (EV_Object, CV_Object, NV_Object, and TV_Object) is enhanced to include properties covering mobility and cloud-ness (Fig. 5). A property “mobility” is a Boolean with default value false, meaning if it is true the object is mobile object. Only in that case, time and location properties are set. In the same manner, a property "cloud-ness" is a Boolean with default value false, meaning if it is true the object is cloud-supported object. In this case, cloud type and cloud service type properties are set.


«stereotype»
EV_Object
+mobility : Boolean [1] = false
+time : date [0..1]
+location : LocationInSpaceType
+cloudness : Boolean [1] = false
+cloudType : CloudType [0..1]
+cloudService : CloudServiceType [0..1]
«stereotype»
CV_Object
+mobility : Boolean [1] = false
+time : date [0..1]
+location : LocationInSpaceType [0..1]
+cloudness : Boolean [1] = false
+cloudType : CloudType [0..1]
+cloudService : CloudServiceType [0..1]
«stereotype»
NV_Object
+deployedNode : String
+securityDomain : String
+managementDomain : String
+mobility : Boolean [1] = false
+time : date [0..1]
+location : LocationInSpaceType [0..1]
+cloudness : Boolean [1] = false
+cloudType : CloudType [0..1]
+cloudService : CloudServiceType [0..1]
«stereotype»
TV_Object
+mobility : Boolean [1] = false
+time : date [0..1]
+location : LocationInSpaceType [0..1]
+cloudness : Boolean [1] = false
+cloudType : CloudType [0..1]
+cloudService : CloudServiceType [0..1]
«enumeration»
CloudType
Public
Private
Hybrid
«enumeration»
CloudServiceType
SaaS
PaaS
IaaS
«dataType»
LocationInSpaceType
+locationInstance
+locationType
Figure 5: Stereotypes for Mobility and Cloud

4.2 Social Network Extension as UML Profile

EV_Party was enhanced to have sociality, social information, and social communities properties. EV_Community was enhanced to have sociality and participants properties A new stereotype SocialRelationship, which extends UML Association, was also introduced (Fig. 6). The details of SocialInformationType are not defined here.


«stereotype»
EV_Party
+sociality : Boolean = false
+socialInformation : SocialInformationType
+socialCommunities : EV_Community [*]
«stereotype»
SocialRelationship
+source : EV_Party
+target : EV_Party
+socialContext
«stereotype»
EV_Community
+sociality : Boolean = false
+participants : EV_Party [*]
«dataType»
SocialInformationType
Figure 6: Stereotypes for Social Network

5 IMPACT ON ENTERPRISE ARCHITECTURE ELEMENTS

We have applied those stereotypes together to see what kinds of impact Enterprise Architectures receive.

5.1 Mobile Device

We observe the following impact.

1) Enterprise Viewpoint: The most part of this viewpoint model does not get impact of mobile object, since this viewpoint model mainly talks about why and what. However, a Role, an abstraction of behavior performed by Object, may get influence by mobile object. Especially, policy value will need update to include the cases where some Roles are performed by mobile object. For instance, if a mobile object performs a part of the process or interaction, e.g. new security policy X may apply.

2) Information Viewpoint: No impact

3) Computational Viewpoint: Although this viewpoint does not care about distribution, mobility is functionality, and a mobile object can still be introduced. That will give surrounding objects some impact. For instance, an object providing geographical map based on a mobile object's TimedLocation information may be introduced to support mobile objects.

4) Engineering Viewpoint: A case of a mobile object moving from Node A to Node B becomes a possibility. Also, a mobile object may need multiple channels for communication, since available channel may be different from place to place and from time to time.

5) Technology Viewpoint: Technology objects representing software, hardware, and network will be categorized into mobile and non-mobile object.

5.2 Cloud Computing

We observe the following impact.

1) Enterprise Viewpoint: The same observations as above (5.1 1)) apply. Policy value will need update to include the cases where some Roles are performed by cloud object. For instance, if a part of the process (or a step) or interaction is performed by a cloud object, or an artifact used is fulfilled by a cloud object, e.g. new security policy Y may apply.

2) Information Viewpoint: No impact

3) Computational Viewpoint: Computational Viewpoint model specifies distributed transparency attributes as a whole (see UML4ODP). Depending on the cloud provider or service, a part of distribution transparency may be provided by the cloud, which means with cloud object computational model may become composite of ODP specified distribution transparency part and cloud provided distribution transparency part.

4) Engineering Viewpoint: In case of SaaS, Engineering Viewpoint model including cloud objects and channels to communicate with them is all we need to define. Other elements such as Node for SaaS may be created as virtual element. In case of PaaS, a cloud object providing specific application functionality and the platform are the main elements to be modeled. Other elements such as Node for PaaS may be created as virtual element as well. In case of IaaS, it is possible to model most of the engineering viewpoint except for Nucleus etc.

5) Technology Viewpoint: Technology objects representing software, hardware, and network will be categorized into cloud and non-cloud object.

5.3 Social Network

We observe the following impact.

1) Enterprise Viewpoint: SocialParty, SocialRelationship and SocialCommunity are additions to the viewpoint model, and those need to be defined. The behaviors defined in the model will need updates to reflect the new elements. A process to construct social profile, setup/execute social activity, and to achieve some social objective with the help of social relationships may be added.

2) Other Viewpoints: No impact except for normal viewpoint modeling of supporting viewpoint objects for communicating with social networks.

6 APPLYING PROFILES TO MAJOR ELEMENTS OF ENTERPRISE ARCHITECTURE

We have applied all the UML Profile elements described before into existing UML4ODP Profile definition. With this revised UML4ODP, we can create new kinds of models or diagrams as a step towards Flexible Enterprise Architecture.

6.1 Enterprise Viewpoint Model

There are various types of model or diagram in Enterprise Viewpoint when UML4ODP is used. The following covers only major diagrams.

Objective diagram: A diagram showing Objective decomposition

CommunityContract diagram: A package diagram showing Community and Objective, a package of EnterpriseObjectTypes, a package of Roles, a package of Policies, and a set of Processes.

EnterpriseSpec diagram: A package diagram showing included CommunityContract packages and associated FieldOfApplication.

EnterpriseObjectTypes diagram: A package showing included EnterpriseObjectTypes including ODPSystem, and the relationships among them

RolesInCommunity diagram: A Community and a list of Roles involved

RolesObjects diagram: A diagram showing a set of Roles, a set of Objects, and FulfilsRole relationships between them

Interaction diagram: One or more Interactions with associated Roles, referenced Artifacts, and Enterprise Objects fulfilling the artifact roles

Process diagram: An activity diagram showing Roles and their bahaviors (Steps, Artefact, etc.)

Policy diagram: A set of Policy Envelope, Policy Value, relevant controlling Process, and affected behaviors such as Interactions

A package of Enterprise Object Types can include Enterprise Object Types with new properties (Fig. 7 and 8).

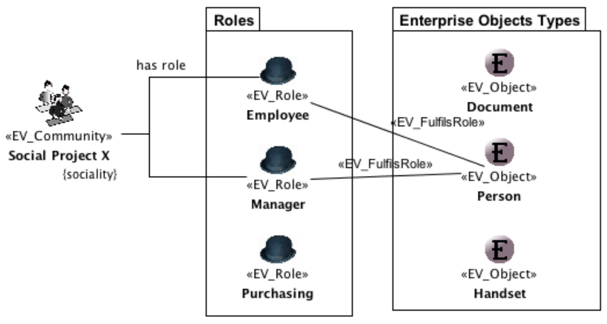


Figure 7: Example Relationships among Enterprise Object Types, Roles, and their participating Social Project

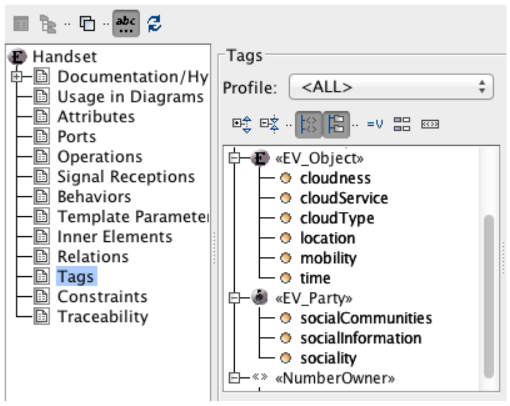


Figure 8: Enterprise Objects with properties and Roles

Enterprise Objects with new properties (see Fig. 8 for Handset) can also appear in Interaction Model (Fig. 9).

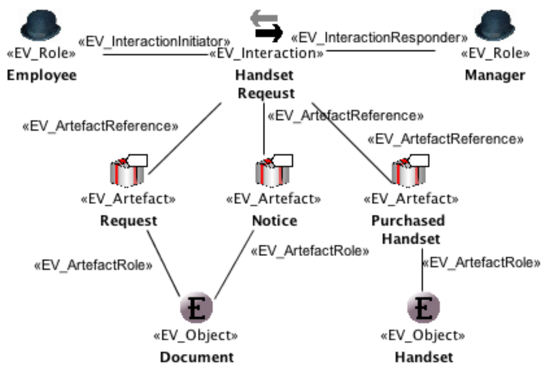


Figure 9: Sample Interaction Diagram

Defined Enterprise Objects can also appear in process diagram (Fig. 10).

Although those are RM-ODP specific diagrams, similar models can be found in other Enterprise Architecture Frameworks.

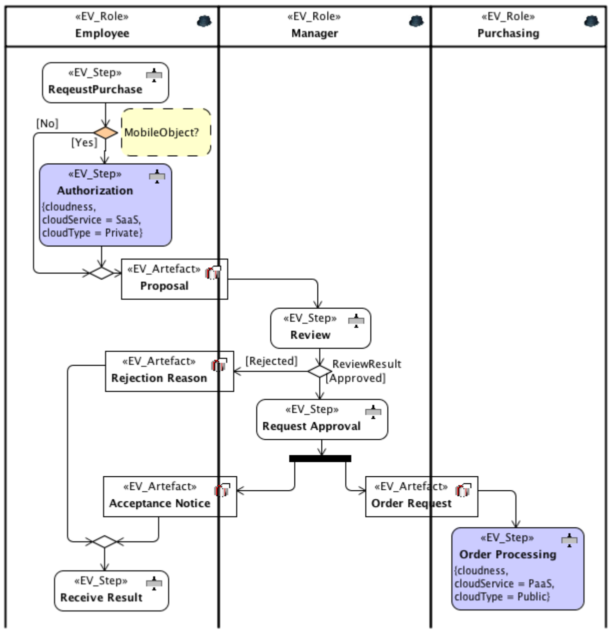


Figure 10: MobileObject in Sample Business Process

6.2 Information Viewpoint Model

We have defined no additional stereotypes for this viewpoint. We can, however, still define additional data types or Information Objects in Invariant Schema diagram using standard UML4ODP (Fig. 11).

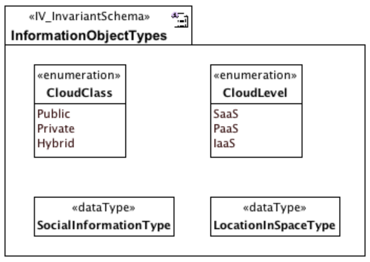


Figure 11: New Invariant Schema Elements

6.3 Computational Viewpoint Model

There are various types of model or diagram in Computational Viewpoint when UML4ODP is used. The followings are two of the examples.

Architecture diagram: A diagram showing logical grouping of architectural packages such as application objects package containing business functions package and ODP function package. Components definition can be a part of this diagram.

Interface/Signature diagram: A diagram showing a set of interface definition and signature definitions, with datatype definitions used.

A Computational Object can have TimedLocation and/or CloudNature as properties and be used in the architecture diagram (Fig. 12).

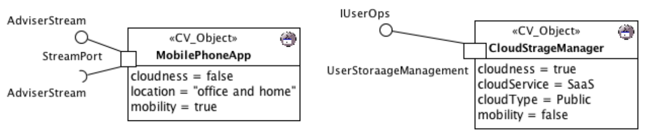


Figure 12: Sample Mobile and Cloud Components

6.4 Engineering Viewpoint Model

Types of diagrams in Engineering viewpoint are similar to those of Computational viewpoint, and the differences are in their distribution-awareness, or their mission to support distributed transparencies, and in their internal structure of Nodes and Channels. When we use e.g. SaaS, it would be the objects on remote node to access and use, and in general there is no need, or no way to describe internals of the target SaaS system. However for the purpose of this modeling, we used the same stereotypes with properties (Fig. 13) for describing SaaS object.

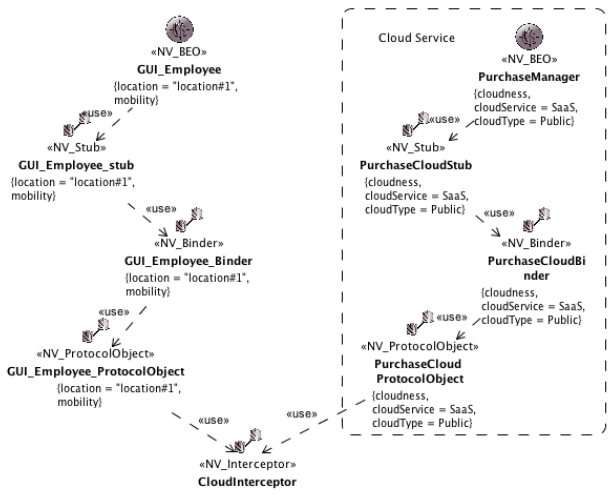

«NV_BEO»
GUI_Employee
{location = "location#1", mobility}
«use»
«NV_Stub»
GUI_Employee_stub
{location = "location#1", mobility}
«use»
«NV_Binder»
GUI_Employee_Binder
{location = "location#1", mobility}
«use»
«NV_ProtocolObject»
GUI_Employee_ProtocolObject
{location = "location#1", mobility}
«use»
Cloud Service
«NV_BEO»
PurchaseManager
{cloudness, cloudService = SaaS, cloudType = Public}
«use»
«NV_Stub»
PurchaseCloudStub
{cloudness, cloudService = SaaS, cloudType = Public}
«use»
«NV_Binder»
PurchaseCloudBinder
{cloudness, cloudService = SaaS, cloudType = Public}
«use»
«NV_ProtocolObject»
PurchaseCloud ProtocolObject
{cloudness, cloudService = SaaS, cloudType = Public}
«use»
«NV_Interceptor»
CloudInterceptor

Figure 13: Sample Channel from Mobile Device to Cloud Services

6.5 Technology Viewpoint Model

Technology Object with mobility or cloud-ness can be used to specify elements of hardware, software, or network. However, in case of cloud computing, and if it is a private cloud, there is not much difference with ordinary in house servers case (Fig. 14). However, if it is a public cloud, there is a limit for defining technological architectures, since internal of cloud services is not visible, and cannot be specified, from outside.

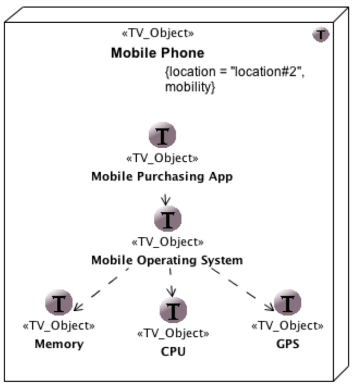


Figure 14: Sample Technology Object for Mobile Device

6.6 Viewpoint Correspondence Model

RM-ODP provides concept of Correspondence, with which we can specify a model element in one viewpoint is related to another model element or model elements in other viewpoint. This is implemented in UML4ODP, and without modifications, we can use this capability in our example as well (Fig. 15).

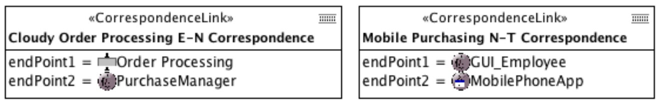

«CorrespondenceLink»
Cloudy Order Processing E-N Correspondence
endPoint1 = Order Processing
endPoint2 = PurchaseManager
«CorrespondenceLink»
Mobile Purchasing N-T Correspondence
endPoint1 = GUI_Employee
endPoint2 = MobilePhoneApp

Figure 15: Sample Correspondences

7 FINDINGS IN BRINGING FLEXIBILITY INTO ENTERPRISE ARCHITECTURE

7.1 Summary of Proposed Extension Mechanism

The mechanism we have employed can be summarized as follows. First, we need to have meta-models for Enterprise Architecture and new technologies in the context of Enterprise Architecture. Note it is necessary to understand the domain enough to define meta-models. Second, the meta-model for Enterprise Architecture, as a receiving package, package-merges with one or more of meta-models, as merged packages, for new technologies. When there is a conflict, a process to resolve the conflict described below should be followed. Third, based on those meta-models, we can define UML Profile definitions. UML Profile definitions can also use package merge by treating UML Profile for Enterprise Architecture as a receiving package and UML Profile for new technologies as merged packages.

7.2 Conflict resolutions process

When two concepts from two meta-models are considered similar, there may be several types of relationships between the two. The following diagram (Fig. 16) shows several possibilities.

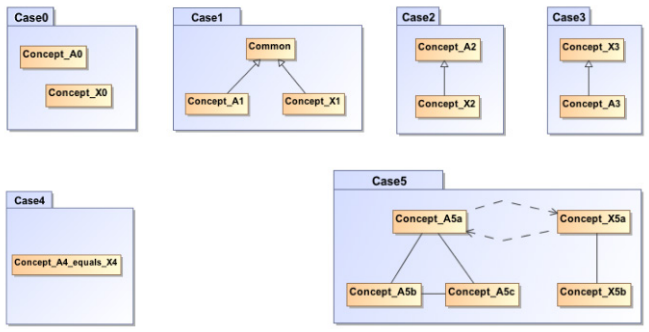


Figure 16: Possible relationships between two concepts

They could be different and independent (Case0), may share the common super concept (Case1), one of them may be the super class of the other (Case2 and Case3), they may be the same concept (Case4), or each of them has its structure and one of each element may be the same concept (Case5).

The proposed process is the following.

Step 1: Compare the definitions to understand the meaning of the concept in each context, and highlight the difference.

Step 2: Compare the surrounding concepts of the two to find differences between the two concepts.

Step 3: Compare the constraints (if existed) to highlight the similarity and differences.

Step 4: Identify the type of relationships between the two.

Step 5: Decompose two concepts into finer grained concepts and relationships, when they are composite concept, find similar concepts between the two sets, and apply Step1 to Step4.

Step 6: Apply direct integration or UML package merge to integrate two meta-models.

Once meta-model is integrated, we can proceed to define UML Profile for it.

7.3 Relatively Independent Cases

If target new technologies were relatively independent from existing Enterprise Architecture elements, there would be at least two enhancement strategies for integration. The first strategy is to add new technology elements at meta-model level (package merge of meta-models), and enhance existing elements at UML Profile level (package merge of UML Profile). The second strategy is to add new technology elements at meta-model level (package merge of meta-models), and at UML Profile level (package import of UML Profile). With the first strategy, users will need to understand the updates for each stereotypes, but will be able to use the same set of stereotypes. This can be used in the case that quick implementation is required. With the second strategy, users will need to learn new stereotypes. This can be used to get better modularity of UML Profile, but too many stereotypes may become an issue.

7.4 Conflicting Cases

If the target new technology meta-model has some conflict with existing Enterprise Architecture elements, the conflicting portions need to be resolved by following the resolution process explained in 7.2. One such example is SOA concepts against RM-ODP concepts. Key concepts are quite similar but they use different terminologies with the different scope. Additional resolution strategy for the conflict is 1) to choose primary one over the other and gradually migrate the second choice to the first one, or 2) to not change anything but treat these as independent and just add correspondence in formal way (e.g. using OCL) or even in informal way. If, for instance, we need to bring SoaML [15] into UML4ODP, the conflict may be around Interaction (RM-ODP) and Collaboration (SoaML). In this case, choose the one that is more suitable for defining behavior in integrated SOA/RM-ODP world. Another example is BMM (Business Motivation Model) [16] against RM-ODP, and the conflict is between Objective (RM-ODP) and Vision/Goal/Objective (BMM). In this case it would be less difficult to resolve, since BMM provides wider and richer goal/motivation model. It depends on how deep objective model you need.

8 DISCUSSIONS

8.1 Flexible Extension Mechanism

The question we started with was “is it possible to design flexible or extendable Enterprise Architecture, preparing for the time new technology emerges?” We developed an extension mechanism described in 7.1 “Summary of Proposed Extension Mechanism” and in 7.2 “Conflict resolution process." In the final step of merging UML Profiles, we need to add/modify stereotypes, properties, and constraints of UML4ODP. When doing this, use of modeling tool is helpful, especially when you already have meta-model data to be revised and profile data to be revised. Some modeling tools have validation capability. Even without tooling, it is still possible to take the same steps. However, if you did it by hand, it would be hard to avoid errors and inconsistencies.

8.2 Enterprise Architecture and Model based Software Development

Enterprise Architecture usually means high-level description of the entire enterprise system. That may not change, but if Enterprise Architecture is presented as e.g. a UML model, we can at least make consistent modifications to the model with the help of UML tools. When the idea of Model Driven Architecture (MDA) [17] was introduced, there was no MDA tooling available. Today, there are some commercial and open-source products for model transformations, such as eclipse modeling project. Once source model is prepared, it can be used as an input to model transformation tool chains. Enterprise Architecture in UML is a reasonable starting point for this process. Also, if it is UML model, whole or a part of the model may become candidate for reuse. The real challenge would be to achieve step-by-step model-to-model transformation chains. There are several UML tools that support RM-ODP, Zachman Framework, and TOGAF (e.g. MagicDraw). With those tools and with enhancement support mechanism in place, the tool chain can consume UML models, representing enhanced Enterprise Architecture. In this case, development of model transformation logic is needed and once it is done it would become a candidate for reuse, since it is built against standard UML models. Another possibility is use of Domain-Specific Language or DSL [18]. Some people prefer DSL to UML for its simplicity. If a set of DSLs are designed to describe Enterprise Architecture, then created models (XMI form) can be used as an input for the tool chains. In this case, however, development of model transformation logic is required for each DSL, and because they are not standard based, reuse may become an issue.

8.3 Interoperability among Enterprise Architecture Frameworks

As of today, there is no interoperability among different Enterprise Architecture Frameworks, which produced silos of Enterprise Architectures. The required actions are to make their meta-model open and encourage development of transformations between them, or to make one of the meta-models as standard and each framework provider to develop and provide transformation to/from the standard. The issue in doing this is the difference in scope or coverage of the concepts for Enterprise Architecture. The first thing to do, therefore, is to agree on common core set of concepts for Enterprise Architecture framework. At least among the three frameworks mentioned in this paper, RM-ODP based one is the most neutral and open, and this could be used as a base for the discussion on the common core.

9 RELATED WORK

There are a number of research papers and books on Enterprise Architecture and the modeling of it. After Zachman's first paper [2] in 1987, "Generalised Enterprise Reference Architecture and Methodology" [21] was developed by IFAC/IFIP Task Force on Architectures for Enterprise Integration in 1999, "Recommended Practice for Architecture Description of Software-Intensive Systems" or IEEE 1471-2000 [22] was standardized by IEEE in 2000, "A Method to Define an Enterprise Architecture using the Zachman Framework" [23] in 2004 proposed a practical method for utilizing Zachman Framework, and "A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodeling Platform" [24] book discussed about meta-model for Enterprise Architecture in 2005. This stream of researches continues to reflect recent topics at e.g. EDOC Conference. In the area of UML modeling tool for Enterprise Architecture, there are activities within OMG to develop/enhance UML Profile for Enterprise Architectures such as UPDM. "A Tool for the Model-Based Specification of Open Distributed Systems" [25] in 2013 describes UML tool for ODP modeling, based on ISO's UML4ODP standard.

10 CONCLUSION

In order to integrate new technologies into Enterprise Architecture, we will need to take the following steps: analyze the domain that new technology is applied and follow the steps described in 7.1 "Summary of Proposed Extension Mechanism" and 7.2 "Conflict resolutions process." The use of "package merge" allows merging only necessary set of packages into Enterprise Architecture package. The steps handling UML Profile may be replaced with other steps, e.g. generating DSLs using eclipse-modeling project, for non-UML model case.

It is likely that new technology provides new capability to things or persons. In this case, new meta-model element related to the capability should be related to Object or Party (Person) so that the capability is explicitly visible to UML Profile designer.

If the Enterprise Architecture model is used in model driven environment as an input file, it should be interpretable by model transformation engines. This means input file should better be in the form of UML or XMI [19].

Regarding openness and interoperability of Enterprise Architectures, we will need a chance to discuss and find common core concepts. Until this is done, RM-ODP based Enterprise Architecture would be the most open one, since it is an ISO/IEC/ITU-T standard, and standard document, meta-model data, and UML Profile data are available on the Internet.

ACKNOWLEDGEMENTS

The authors thank to contributors in Enterprise Architecture domain, Modeling Language domain including

MOF, UML, UML Profile for DoDAF and MODAF and a variety of essential modeling standards by Object Management Group, and various Eclipse Modeling Projects by Eclipse Foundation including EMF, GMF, and Xtext. The authors also thank many people from various countries involved in ISO/IEC JTC1/SC21/WG7, SC7/WG19, and ITU-T SG19's continuing efforts to standardize and maintain RM-ODP family of standards including Use of UML for ODP system specifications. The authors finally thank many participants to IWIN 2103 who provided us with valuable feedbacks.

REFERENCES

[1] J.A. Zachman,``John Zachman's Concise Definition of The Zachman Framework™,'' http://www.zachman.com/about-the-zachman-framework (2013).

[2] J.A. Zachman, ``A Framework for Information Systems Architecture,'' IBM Systems Journal, Vol.26 No.3 (1987).

[3] The Open Group, ``TOGAF Version 9.1,'' http://pubs.opengroup.org/architecture/togaf9-doc/arch/.

[4] FEA Reference Models, ``Consolidated Reference Model Version 2.3,'' http://www.whitehouse.gov/sites/default/files/omb/assets/fea_docs/FEA_CRM_v23_Final_Oct_2007_Revised.pdf.

[5] ISO/IEC 10746-2:2009, Information technology -- Open distributed processing -- Reference model: Foundations.

[6] ISO/IEC 10746-3:2009, Information technology -- Open distributed processing -- Reference model: Architecture.

[7] ISO/IEC 15414:2006, Information technology -- Open distributed processing -- Reference model -- Enterprise language.

[8] ISO/IEC 19793:2008, Information technology -- Open Distributed Processing -- Use of UML for ODP system specifications.

[9] D. Hashimoto, A. Tanaka, and M. Yokoyama, ``Case study on RM-ODP and Enterprise Architecture,'' Proc. of 11th International IEEE EDOC Conference Workshop (EDOCW'07), pp.216-223 (2007).

[10] National Institute of Standards and Technology, ``The NIST Definition of Cloud Computing,'' Special Publication 800-145 (2011).

[11] D. Dreyfus, and B. Iyer, ``Enterprise Architecture: A Social Network Perspective,'' Proc. of 39th Hawaii International Conference on System Sciences, p.178a (2006).

[12] OMG, Unified Modeling Language™ (UML®), http://www.omg.org/spec/UML/

[13] OMG, Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, http://www.omg.org/spec/QVT/1.1/PDF/ (2011).

[14] OASIS, Reference Model for Service Oriented Architecture 1.0, http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf.

[15] OMG, Service oriented architecture Modeling Language (SoaML®).

[16] OMG, Business Motivation Model (BMM), http://www.omg.org/spec/BMM/.

[17] OMG, Model Driven Architecture ® (MDA ®), http://www.omg.org/mda/.

[18] M. Fowler, ``Domain-Specific Languages,'' Addison-Wesley (2011).

[19] OMG, XML Metadata Interchange (XMI®), http://www.omg.org/spec/XMI/.

[20] OMG, Unified Profile for DoDAF and MODAF (UPDM), http://www.omg.org/spec/UPDM/.

[21] IFAC/IFIP Task Force on Architectures for Enterprise Integration, ``Generalised Enterprise Reference Architecture and Methodology'' (1999).

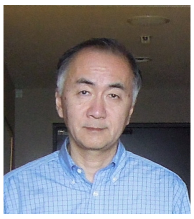
[22] IEEE 1471-2000, ``Recommended Practice for Architecture Description of Software-Intensive Systems''.

[23] C.M. Pereira, and P. Sousa, ``A method to define an Enterprise Architecture using the Zachman Framework,'' Proc. of ACM symposium on Applied computing (SAC '04), pp. 1366-1371 (2004).

[24] C. Braun, and R. Winter, ``A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodelling Platform,'' Desel & Frank (2005).

[25] J.R. Romero, J.I. Jaén, and A. Vallecillo, ``A Tool for the Model-Based Specification of Open Distributed Systems,'' The Computer Journal, doi: 10.1093/comjnl/bxs021 (2012).

(Received January 7, 2014)



Akira Tanaka received M.E. from University of Tokyo, Japan in 1978. From 1978 to 2008, he was with Hitachi Ltd. His current research area includes model-based software development. He is a member of ACM, IEEE Computer Society and IPSJ.

Osamu Takahashi received master's degree from Hokkaido University in 1975. He is currently a professor at the Department of System Information Science at Future University Hakodate. His research interest includes ad-hoc network, network security, and mobile computing. He is a member of IEEE, IEICE, and IPSJ.

Development of Teaching Materials for Computer Programming using a Robot Remotely Controlled by a PC through Wireless Communication

Toshihiro Shikama[†]

[†]Fukui University of Technology
shikama@fukui-ut.ac.jp

Abstract - We developed teaching materials for students to increase their interest in computer programming. We employed a robot specified by ET Robocon (Embedded Technology Software Design Robot Contest). Although the robot in ET Robocon is controlled by a program running in the robot itself, a program written by a student runs on a separate PC and also controls the robot through wireless communication via Bluetooth. As for the programming language that students learn, we selected Python because of its simplicity and similarity with object-oriented programming. A student can start programming simple sequential control of the robot and extend it to programming that realizes line tracing.

Keywords: Embedded Systems, ET Robocon, Python, Teaching Materials, Line Tracing

1 INTRODUCTION

This paper reports the development of teaching materials (sometimes shortened to "materials" in this paper) for computer programming. When a student is learning computer programming, the initial stage is important. The student generally takes a long time to become familiar with abstract programming concepts such as data types, structures, and classes of object-oriented programming. These concepts are separate from physical instances and difficult to learn. If we educate students under the false assumption that they will easily understand these abstract concepts, the students may abandon learning because of a loss of interest.

We participated in the ET Robocon (Embedded Technology Software Design Robot Contest) so that students could learn embedded systems [1]. In this contest, students analyze and design a computer program using UML (Unified Modeling Language) to control the robot, which contains strictly defined hardware with no modifications allowed. We observed that students who participated in this contest tended to become enthusiastic about computer programming. From this experience, we expect that more students will show an interest in computer programming if we incorporate the robot into the programming education. Based on this motivation, we developed teaching materials for computer programming using the robot defined by ET Robocon.[1]

2 OUTLINE OF ET ROBOCON AND OBJECTIVES OF THIS WORK

2.1 ET Robocon

The objective of ET Robocon is to improve the capability of software technology for embedded systems. This contest uses control software targeting a two-wheel self-balancing robot using LEGO® MINDSTORMS NXT [2]. Figure 1 shows the appearance of the robot and its components. The robot consists of an ultrasonic sensor, a gyro sensor, a light sensor, and three motors for the right wheel, left wheel, and tail. It is equipped with a 32-bit ARM7 microprocessor. Students develop control programs that enable the robot to autonomously trace a line around a specified course. Figure 2 shows a photo of the ET Robocon 2011 course.

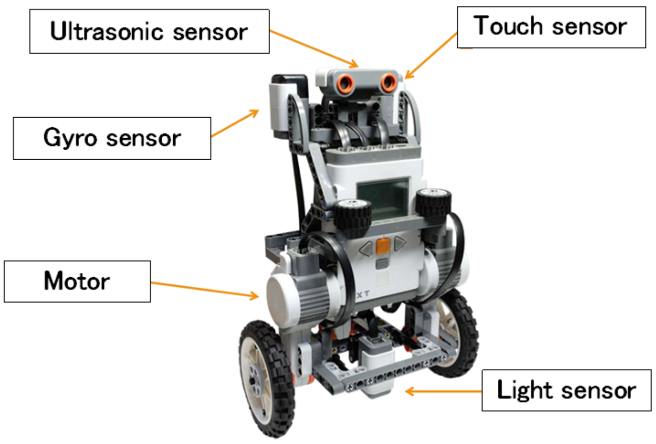


Figure 1: The robot and its components

Figure 2: ET Robocon 2011 course

[1] The work reported in the paper was supported by the Special Research Grant-in-Aid of Fukui University of Technology.

The robot runs along the black lines drawn over white areas, which are surrounded by green "ground". Students are required to develop a control program that makes the robot run along the black line at high speed. All teams in the contest use the same robot, which has a limited number of sensors (an ultrasonic sensor, a gyro sensor, and a light sensor). The ET Robocon consists of two parts: modeling and a time trial. The modeling part is a competition of the UML modeling skill used in developing the program, and the time trial part is a run-time competition of the robot. The total score is determined from the results of the two parts.

2.2 Objectives of this work

The intention of this work is to develop teaching materials for computer programming and to promote the enjoyment of computer programming for beginner students. We are aiming at the following goals:

1) The control of the robot is realized by a simple program (i.e., a small number of program steps).
2) A student can start programming without understanding abstract programming concepts.
3) The basics of programming skills, such as conditional branches, loops, functions, etc., can be studied through the developed materials.
4) The materials can be applied to the education of object-oriented programming and multi-thread programming.

3 THE BASIC ARCHITECTURE

3.1 Outline of the materials

In ET Robocon, a control program is written in the C or C++ language and the program is compiled to produce a binary file which is loaded into the robot through a USB interface. After the program is invoked, the robot is autonomously controlled by the program. Although this scheme can enable accurate and efficient control of the robot, debugging is limited since the robot has only a small display to show internal information and status. Another problem is the time and work required for students; each time a program is modified, the students must compile, link, and download the binary file through the USB interface. After considering these drawbacks, we apply the following scheme for a program running on a PC to control the robot.

1) A fixed control program is pre-loaded into the robot. This program executes basic commands from the PC. Students do not modify the program in the robot.
2) The basic commands are sent from the PC through wireless communication via Bluetooth.
3) Control of the robot is achieved by the program running in the PC. This program describes combinations of basic commands.
4) As the programming language, Python [3] is selected for the program in the PC.

3.2 Adoption of Python

As mentioned above, we adopted Python as the programming language for students to learn. Python is an object-oriented scripting language and has the following features:

- Since a program can be executed without compiling, a student can modify his program easily and test it quickly.
- Python is well defined and easy for beginners to learn.
- Python is used globally.
- A student can learn object-oriented programming easily by Python.
- Python is available at no cost and is supported by multiple platforms, including Windows and Linux.
- Because indents are mandatory in Python, a program can generally be read easily. In addition, differences in the programming style between students are small.

Adoption of Python has the following drawbacks:

- Python has a compatibility problem between versions 2 and 3.
- Performance of the program is slow because it is a scripting language.
- Indentation is employed for identifying program blocks; this programming style is different from other languages, such as C and C++.

For compatibility between Python versions, we use version 2. Our focus is on the educational aspect, and so we do not seek to realize high running speed of the robot. Concerning the indentation used in Python, we think that this is not a serious problem for students, who will study the C or C++ programming language after learning Python.

3.3 Configuration of the materials

Figure 3 shows the total configuration of the developed materials. The robot and the PC are connected through wireless communication via Bluetooth. The program running in the PC controls the robot remotely.

The Python program running in the PC sends a command through wireless communication; the robot moves forward or makes a left or a right turn by following the program commands. The running speed of the robot is also controlled by the program.

A fixed program running on the ARM 7 microprocessor inside the robot controls the movement of the robot; students do not modify this program, which is developed in the C++ language and runs on the Real-Time Operating System nxtOSEK [4]. This program performs the following functions:

- Controls the posture of the robot
- Receives commands through Bluetooth and interprets them
- Executes commands from the PC
- Sends log data to the PC every 40 ms

The Python module, which was developed for this setup, calculates data to get information about the robot, including the X- and Y-coordinates of the robot, the angle of the robot, and the total running distance from the start point.

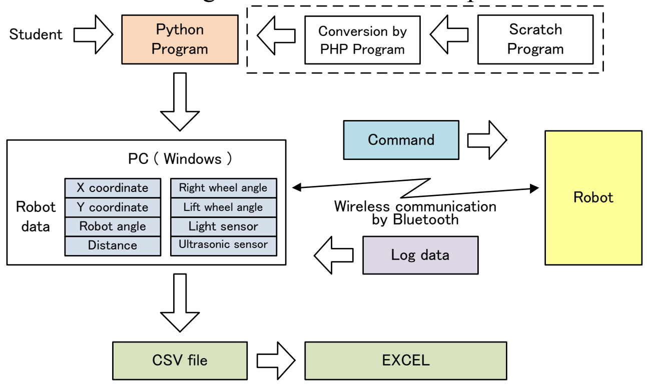


Figure 3: Configuration of the developed materials

When students write programs, they can use variables concerning these data by importing the Python module. The module also provides a log file, including all log data, in the CSV format. Using EXCEL, students can analyze the log file to obtain, for example, a trace of the robot.

Table 1 summarizes the basic functions and commands that students can use in their Python programs. For simplification, the specification commands are limited to one character, whereas extended commands consisting of multiple characters are also provided for future use.

The variable “bt” is the object to control the robot. At the head of a program, the object is generated from the defined class “nxt_bluetooth” as follows:

```
from nxt_bluetooth import nxt_bt
bt = nxt_bt("00:16:53:0c:82:39", 0).
```

The first line imports the class “nxt_bt” from the module “nxt_bluetooth”. This mandatory class was developed for the materials used. The second line generates the object “bt”, where the parameter "00:16:53:0c:82:39" is an example of the MAC address of the robot. Bluetooth employs a 48-bit MAC address, which is the same as that of LANs. As the “nxt_bt” class hides the Bluetooth communications and updates of the robot variables from students, the students can develop their programs without knowing the internal details.

Table 1: Basic functions and commands

Function	Command	Meaning
bt.send (character)	“f”	Move forward
	“r”	Make a right turn
	“l”	Make a left turn
	“b”	Move backward
	“0”–“9”	Set speed
bt.swait (seconds)	-	Wait specified seconds

Table 2 summarizes the variables of the robot status and sensors. The values of these variables are updated every 40 ms, and so students can use these variables to control the robot. For example, students can quantitatively control the robot, such as moving it forward 500 mm or making a 90-degree left turn.

Table 2: Variables of robot status and sensors

Variable	Meaning
bt.x	X-coordinate of the robot
bt.y	Y-coordinate of the robot
bt.angle	Angle of the robot
bt.distance	Total running distance from the start point
bt.diff_light	Value of the light sensor

4 INTERNAL REALIZATION SCHEME OF THE MODULE

Students import the module “nxt_bluetooth” at the head of their program. The class “nxt_bt”, which has been developed in the materials, is included in this module. This class has the functions of communication via Bluetooth, processing log data, and synchronization between the robot and the program running in the PC. As the details are hidden, students have to be aware of only the MAC address of the robot.

When we consider the implementation of the Python class, it is natural to use two separate threads for sending and receiving functions. However, to simplify the implementation, we realize the functions by a single thread, since the robot sends log data every 40 ms. We can eliminate the complexity of the multi-threading and extend the materials to realize the control of multiple robots simply by multi-threading.

Inside the nxt_bt class, sending and receiving functions via Bluetooth are realized by importing the Bluetooth module. This module is provided by python-bluez [5], which is a wrapper function that enables Python to use BlueZ [6]. The robot employs the virtual serial port communication by the serial port profile (SPP) of Bluetooth. BlueZ supports this profile. Although python-bluez is for Linux, PyBluez is also available for the Windows environment. This means that the materials can be used on both platforms, if Python is installed.

Each communication via Bluetooth is initiated by generating a socket with the required parameters and connecting it as follows, where “bt_addr” holds the character string of the MAC address.

```
self.etrobo_address = bt_addr
self.port = 1
self.sock = BluetoothSocket( RFCOMM )
try:
   self.sock.connect((self.etrobo_address, self.port))
except IOError:
   print "Robot is not invoked."
   sys.exit()
print "connected address = ", self.etrobo_address
```

After the socket has been connected, the program enters a wait state, if it calls the receive() function. As we mentioned before, since a single thread performs both sending and receiving, the program has to call the receive() function to enter the receive wait state after its process has completed. This is actually done by calling the wait() or swait() functions. The wait() function specifies a number of 40 ms units as the wait duration, while the swait() function specifies the wait time in seconds.

```
def wait(self, n):
  self.i = 0
  while self.i < n:
    self.receive()
    self.i = self.i + 1

def swait(self, time):
  self.n = time // 0.04
  self.wait(self.n)
```

In the wait() function, the program waits for the receiving data by the self.receive() function. Since the robot sends log data periodically, completion of the receive occurs within 40 ms. The initial part of the receive() function executes the following code.

```
def receive(self):
  self.starttime = time.time()
  self.data = self.sock.recv(34)
  if len(self.data) != 34:
    print "receive byte length =", len(self.data)
    sys.exit()
  self.udata = unpack('<2BI2bH3i4hi', self.data)
```

The second line records the receive time of the log data, and then the third line extracts the received data. As the length of the log data is fixed in units of 34 bytes, the log data is divided into pre-defined formats and stored, if the data length is normal (unpack process). The unpacked data is used to calculate X- and Y-coordinates and the angle of the robot. These calculated values are stored in the Python variables, which students can use in their programs.

As mentioned above, one of the features of the materials described in this paper is that the module and programs, including the one used inside the robot, are completely open (i.e., white box). We are able to customize the robot itself and the Python module for future requests from students as well as teachers.

5 EXAMPLES OF PROGRAMS USING THE DEVELOPED MATERIALS

To explain the use of the developed materials, it is appropriate to show some program samples. We will show examples of a simple sequence control, usage of loops, usage of functions, and a simple line trace in the following.

5.1 Example 1

The program shown in Fig.4 is a basic program that controls the robot sequentially. After the program is invoked, the robot moves forward for 2 seconds, turns right for 2 seconds, moves forward for 2 seconds, turns left for 2 seconds, and then stops. Each time a command is sent, the next command is issued after the time specified by the "swait()" function. Since the program is written in Python, the program file has the extension "py". If the name of the program is "sample1.py", the program is invoked by typing the following command in a terminal window.

```
python sample1.py
```

```
from nxt_bluetooth import nxt_bt
bt = nxt_bt("00:16:53:0c:48:1e", 0)

print "START"
bt.send( "3" )
bt.send( "f" )
bt.swait(2)
bt.send( "r" )
bt.swait(2)
bt.send( "f" )
bt.swait(2)
bt.send( "l" )
bt.swait(2)
bt.send( "0" )
print "END"
```

Figure 4: Example 1—sequential control

5.2 Example 2

The program shown in Fig.5 uses a "while" loop to check the value of a variable repeatedly. The execution leaves the loop if the variable takes a specific value. Here, the robot moves forward 500 mm (50 cm), then it makes a 180-degree left turn. After this it moves 50 cm forward again and then stops. By using the variable "bt.distance" that indicates the total distance from the start point and the "while" loop, the program can control the moving distance quantitatively. When the robot makes a turn, the angle of the robot can also be controlled in the same manner.

```
from nxt_bluetooth import nxt_bt
bt = nxt_bt("00:16:53:0c:48:1e", 0)

print "START"
bt.send( "f" )
bt.send( "3" )
while bt.distance < 500:
  bt.swait(0.04)
bt.send( "l" )
while bt.angle < 180:
  bt.swait(0.04)
bt.send( "f" )
target_dist = bt.distance + 500
while bt.distance < target_dist:
  bt.swait(0.04)
bt.send( "0" )
print "END"
```

Figure 5: Example 2—while loops

5.3 Example 3

The program shown in Fig.6 defines functions concerning an advance and a left turn. Each function takes a parameter:

a distance or an angle. This program makes the robot move forward 300 mm, make a 180-degree left turn, and then move forward 300 mm. The program repeats these actions four times by using the “for” loop.

```
from nxt_bluetooth import nxt_bt
bt = nxt_bt("00:16:53:0c:48:1e", 0)

def forward(distance):
    t_distance = bt.distance + distance
    bt.send( "f" )
    while bt.distance < t_distance:
        bt.swait(0.04)

def left_turn(angle):
    t_angle = bt.angle + angle
    bt.send( "l" )
    while bt.angle < t_angle:
        bt.swait(0.04)

print "START"
bt.send( "3" )
bt.swait(0.04)
for var in range(0, 4):
    forward(300)
    left_turn(180)

bt.send( "0" )
print "END"
```

Figure 6: Example 3—“for” loop

5.4 Value of the light sensor

Figure 7 shows changes in the light sensor value as the robot moves over the course illustrated in Fig.8. In this case, the robot crosses the black line on the course several times to measure the characteristics of its light sensor. The sharp dips observed in Fig.7 occur when the robot crosses the black line. While the robot moves over the white part of the course, the sensor value is approximately 900. When it crosses the black line, the sensor value decreases below 400. Students confirm the characteristics of the light sensor by themselves. Based on these results, black and while colors can be identified by using some threshold value, for example, 700.

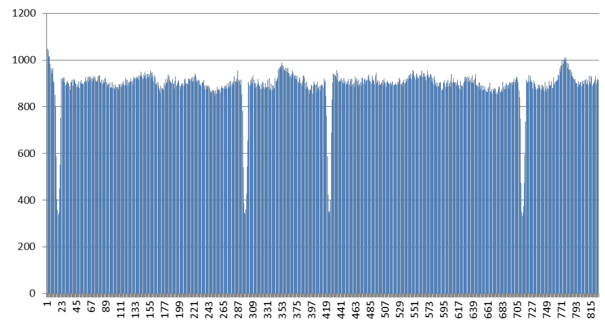


Figure 7: Change of the light sensor value, where horizontal axis is time

If the value of the light sensor is larger than 700, it seems that the robot is running over the white part; otherwise, the robot is running on the black line. Students can know that the robot movement is tracing the black line by using this threshold value.

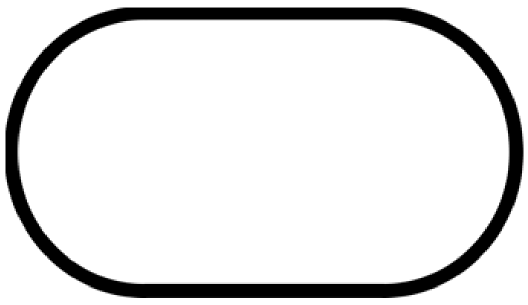


Figure 8: Course for line tracing

5.5 Example 4

Making use of the characteristic of the light sensor and the threshold value, the student can realize line tracing by the robot.

Figure 9 shows the simple program that realizes the line tracing. The variable “target” holds the threshold value. In the infinite “while”, the variable “diff_light” holds the value of the light sensor. If the value of the light sensor is less than the threshold value, the robot makes a right turn; otherwise it makes a left turn. The program repeats this process endlessly every 40 ms. This simple program is able to make the robot move along the black line. Students can learn conditional branching through this example.

```
from nxt_bluetooth import nxt_bt
bt = nxt_bt("00:16:53:0c:48:1e", 0)

print "START"
target = 700
bt.send( "2" )
bt.send( "f" )
while True:
    print bt.diff_light
    if bt.diff_light < target:
        bt.send( "r" )
    else:
        bt.send( "l" )
    bt.wait(0.04)

bt.send( "0" )
```

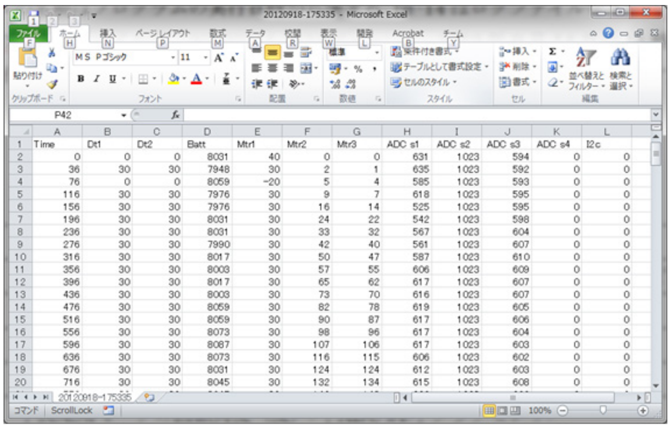
Figure 9: Example 4—Line tracing by simple control

5.6 The log file

Each time a program is executed, a log file is produced. This file includes a record concerning the details of the robot every 40 ms. Table 3 summarizes the items included in each record of the file. The log data is recorded in CVS format. Figure 10 shows an example of the log file opened in EXCEL. Students can analyze the log file and obtain a trace of the robot by using some mathematical calculations. Figure 11 shows an example of a trace of the robot obtained from calculations. The trace is almost the same as the course depicted in Fig.8. We can observe zigzag lines in the trace, which is the effect of the simple "ON and OFF" control by the program listed in Fig.9.

Table 3: Items recorded in the log file

Item	Meaning
Time	Elapsed time (ms)
Dt1	PWM value for right motor
Dt2	PWM value for left motor
Batt	Voltage of battery
Mtr1	Rotate angle of tail motor
Mtr2	Rotate angle of right wheel motor
Mtr3	Rotate angle of left wheel motor
ADC s1	Gyro sensor value
ADC s2	Ultrasonic sensor value
ADC s3	Light sensor value
ADC s4	Touch sensor value
I2c	Distance measured by the ultrasonic sensor



20120918-175335 - Microsoft Excel

	A	B	C	D	E	F	G	H	I	J	K	L
1	Time	Dt1	Dt2	Batt	Mtr1	Mtr2	Mtr3	ADC s1	ADC s2	ADC s3	ADC s4	I2c
2	0	0	0	8031	40	0	0	631	1023	594	0	0
3	36	30	30	7948	30	2	1	635	1023	592	0	0
4	76	0	0	8059	-20	5	4	585	1023	593	0	0
5	116	30	30	7976	30	9	7	618	1023	595	0	0
6	156	30	30	7976	30	16	14	525	1023	595	0	0
7	196	30	30	8031	30	24	22	542	1023	598	0	0
8	236	30	30	8031	30	33	32	567	1023	604	0	0
9	276	30	30	7990	30	42	40	561	1023	607	0	0
10	316	30	30	8017	30	50	47	587	1023	610	0	0
11	356	30	30	8003	30	57	55	606	1023	609	0	0
12	396	30	30	8017	30	65	62	617	1023	607	0	0
13	436	30	30	8003	30	73	70	616	1023	607	0	0
14	476	30	30	8059	30	82	78	619	1023	605	0	0
15	516	30	30	8059	30	90	87	617	1023	606	0	0
16	556	30	30	8073	30	98	96	617	1023	604	0	0
17	596	30	30	8087	30	107	106	617	1023	603	0	0
18	636	30	30	8073	30	116	115	606	1023	602	0	0
19	676	30	30	8031	30	124	124	612	1023	603	0	0
20	716	30	30	8045	30	132	134	615	1023	608	0	0

Figure 10: Example of a log file opened in EXCEL

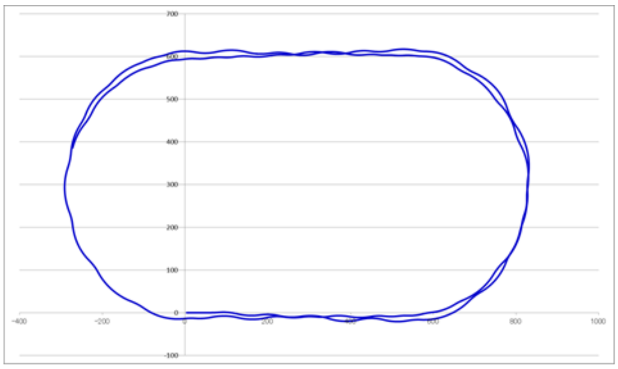


Figure 11: Example of a trace of the robot

6 EXTENSION OF THE PROGRAM FOR CONTROLLING MULTIPLE ROBOTS

The examples explained above concern basic programs that control a single robot. The materials can be applied to the advanced case where multiple robots are controlled by threads. Figure 12 shows the configuration of this case, where two robots are controlled by a single program. A program developed by a student generates two threads for two robots. Each thread executes the same program and controls one of the two robots. Figure 13 shows the sample program for this.

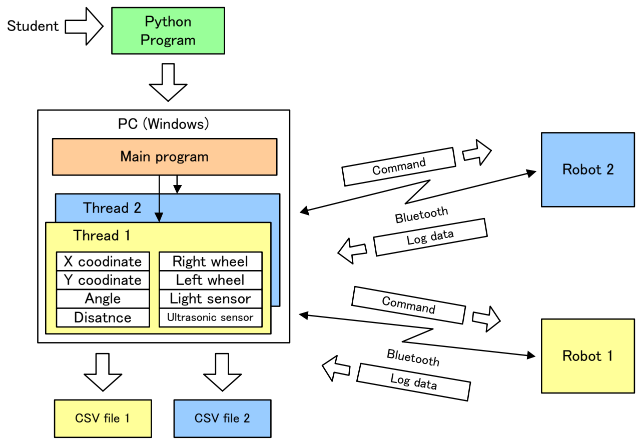


Figure 12: Configuration of the developed materials for controlling two robots

```
import threading    # thread model
import time
from nxt_bluetooth import nxt_bt

class test(threading.Thread):
    def __init__(self, s):
        threading.Thread.__init__(self)
        self.setDaemon(True)
        self.bt = nxt_bt(s, 0)

    def run(self):
        self.bt.start()
        self.bt.send( "3" )
        self.bt.send( "f" )
        self.bt.swait(5)
        self.bt.send( "r" )
        self.bt.swait(5)
        self.bt.send( "0" )

if __name__ == "__main__":
    t1 = test(""00:16:53:0c:48:1e")
    t2 = test("00:16:53:0c:82:39")
    print "Hit enter key, if you are ready."
    raw_input()
    t1.start()
    t2.start()
    time.sleep(20)
```

Figure 13: Program controlling two robots

The part of the program surrounded by the dashed line is the definition of the class that defines the movement of the robots. Two threads are generated from the same class in the main part; the movement of the two robots is the same in this case. Advanced students can learn thread mechanisms through this example.

7 APPLICATION OF THE MATERIALS TO AN ACTUAL CLASS

We applied part of the materials to sessions of an actual experimental class in the first semester of this year. We conducted 9 sessions. The total number of students who participated in the sessions was 45. Approximately half of the students had no experience in programming. The duration of each session was 3 hours and the number of students for one session was at most 7. We explained the material for the first 40 minutes. Two robots were employed to execute the programs. Then students were asked to write four simple programs, including one for line tracing. Although some students had difficulty in understanding Python, 87% of the students indicated a positive impression of the session and expressed their satisfaction when the robot moved correctly. Although students with no programming experience had strong concerns about Python programming, after the session, most of them stated that it was easier than they had thought it would be.

Table 4 shows the summary of their impressions of the materials. Students of Group A had not taken a class in the C programming language as university students, whereas Group B students had, although some students in Group A had learned the C programming language in high school. Students also pointed out aspects of the materials that could be improved.

Table 4: Summary of impressions by students

Impression	Excellent	Good	No comment
Group A	28	1	4
Group B	17	0	1

We also demonstrated the materials and explained a simple program to high school students. A large number of these students found the materials highly interesting.

Through the experience of the actual class, we could identify advantages of the developed materials, summarized as follows:

- The materials could attract more attention from students who had no programming experience.
- As students could edit and execute a Python script directly, program errors were modified quickly. Most of the students could complete the given exercises within the prescribed class hour.
- Students were strongly impressed when the robot performed as they intended.
- Students were also surprised when they got a trace of the robot from the log file.

However, we found drawbacks of the materials from the experience:

- Much effort was needed for preparing and guiding a session.
- Support by teaching assistants was needed for every three or four students to help when they encounter programming problems.
- The difficulty level of programs that students have to develop should be reconsidered. Natural steps from simple to advanced are required.
- We have to improve the assignments before the session to shorten the time needed to explain the materials.
- The number of robots is too small in the case of 7 students.
- Because the quality of components in the robot varied, the robot could not go straight accurately after it received the forward command. Compensation for the error caused this variation in the components is needed.

8 RELATED WORK

Application of robots to education has been prevailing. Target of students are widely spread from elementary and high school students to university students [8][9]. There is also a report on a class using a robot for teacher training [10]. In many cases robots are used for the education of embedded systems including programming [11]-[13]. Robots are also used for object-oriented programming education [14]. Robot contests targeting education have also been widely held [15][16].

In all of the above related work, programming is done for a micro-controller located in a robot itself, while, in our materials of this paper, programming is done for a PC that remotely controls a robot through wireless communication. This remote control architecture realizes easy debugging of a program and quick modification of program errors.

There is NXT Python which provides a Python driver and interface for LEGO Mindstorms NXT, where a NXT robot is remotely controlled by Python programming [17]. Although the control architecture is similar to our materials, there is no report concerning application of NXT Python to programming education. In this NXT Python, the default control program pre-installed in NXT is used and cannot be customized, while, in our materials, both the control program in NXT itself and the Python module in the PC are all developed by ourselves. This means the all functions in our materials are in a while box state easy to modify to cope with the future demand.

Another feature of our materials as compared with the related work is that we intended to cover introductory programming as well as object-oriented programming seamlessly. There is no existing work aiming at this aspect.

Most of the related work employs LEGO Mindstorm® for the target robot. Several products target the education of programming by using the robot of LEGO Mindstorms NXT [1]. The typical product is NI LabVIEW for LEGO MINDSTORMS software [18], which makes it possible for students to develop programs by combining predefined blocks graphically. The difference between our materials and this product is that our materials control the robot remotely by the scripting language Python. Students can learn programming through widely used high-level programming language. As a result, they become accustomed to the conventional programming paradigm.

9 CONCLUDING REMARKS

In this paper we reported the development of teaching materials for computer programming. Our objective is to give beginner students the satisfaction of creating programs that control a robot. Students can implement line tracing by a simple program consisting of a small number of program steps.

One of the features of the teaching materials is that the module and programs are in a white box state. We are able to flexibly customize the robot itself and the Python module for future requests. Since the characteristics of the students vary depending on the number of students, their interests, their scholastic ability, and characteristics of the university or college, the capability of customizing the materials is considered to be important. We will improve the teaching materials based on the experiences of this semester and extend them for teaching the basics of object-oriented programming.

We are planning to extend the materials by integrating them with the programming language SCRATCH [7], which is intended for students in elementary or junior high school. The boxes surrounded by the dashed lines in Fig.3 show this extension. We will report on this development in the future.

REFERENCES

[1] ET Robot Contest, available from http://www.etrobo.jp/2013/, accessed 2014-04-09.

[2] LEGO Mindstorms education, available from http://www.afrel.co.jp/mindstorms/nxt/, accessed 2014-04-09.

[3] Python, available from http://www.python.org/, accessed 201404-09.

[4] nxtOSEK/JSP, available from http://lejos-osek.sourceforge.net/index.htm, accessed 2014-04-09.

[5] pybluez, available from http://code.google.com/p/pybluez/, accessed 2014-04-09.

[6] BlueZ, available from http://www.bluez.org/, accessed 2014-04-09.

[7] SCRATCH, available from http://scratch.mit.edu/, accessed 2014-04-09.

[8] S. Kato, and H. Tominaga, ``Applied Programming Exercises for Problem Solving Learning with LEGO Robot Control - Teaching Materials and Exercise Problems for Basic Control Practice by ROBOTC -,'' IPSJ SIG Technical Report, 2011-CE-108, No.3, pp.1-10 (2011).

[9] T. Takahashi, and H. Tominaga, ``Game Projects and Simulation Materials of LEGO Robot Control in Introductory Programming Exercises for High School Students,'' Proc. of EC2013, pp.301-304 (2013).

[10] T. Kamada, ``A Report on Measurement and Control Class Using a Robot for Teacher Training Course Students,'' IPSJ SIG Technical Report, 2009-CE-99, No.11, pp.1-6 (2009).

[11] E. Hayakawa, T. Takahashi, and K. Aoshima, ``Experiment on Embedded System Education using Robot in Computer Science Course,'' IPSJ SIG Technical Report, 2009-CE-98, No.15, pp.127-134 (2009).

[12] H. Nishigaya, H. Aoki, S. Inoue, K. Eguchi, and S. Kurebayashi, ``Lessons of Learning Measurement and Control with an Autonomous 3 Motor Control Robot,'' IPSJ SIG Technical Report, 2009-CE-98, No.15, pp.113-120 (2009).

[13] Y. Nishino, and E. Hayakawa, ``A Note on Robot Based Embedded System Study Environment in Technical High School,'' IPSJ Journal, Vol.51, No.12, pp.2261-2272 (2010).

[14] N. Chubachi, and K. Ito, ``A Trial of Experimental Task using LEGO® in Object Oriented Programming Education,'' IPSJ SIG Technical Report, 2014-CE-124, No.8, pp.1-6 (2014).

[15] H. Yamashita, ``A Robot Contest for Children and Comprehensive Science Education,'' IPSJ Magazine, Vol.48, No.5, pp.502-511 (2007).

[16] K. Hisazumi et al., ``A Distributed Project Based Learning Curriculum to Design Embedded Systems using Contest Challenge,'' IPSJ SIG Technical Report, 2014-EMB-32, No.34, pp.1-6 (2014).

[17] NXT_Python, available from http://home.comcast.net/~dplau/nxt_python/, accessed 2014-04-09.

[18] NI LabVIEW for LEGO® MINDSTORMS®, available from http://www.ni.com/academic/mindstorms/, accessed 2014-04-09.

(Received December 9, 2013)
(Revised April 10, 2014)

Toshihiro Shikama received his B.E and M.E. degrees from Tokyo Institute of Technology in 1974 and 1976, respectively. From 1984 to 1985, he stayed at University of Waterloo. He received his Ph.D. degree from Shizuoka University in 2006. He joined Mitsubishi Electric Corp. in 1976 and had engaged in developing a computer network using a satellite channel, high speed ring type LANs, time division multiplexers, ATM equipment, a high speed IP switch, and network security systems. Since April of 2007, he has been a professor of Department of Electrical, Electronic and Computer Engineering, Fukui University of Technology. He is a member of IPSJ, IEICE, and IEEE Communications Society.

Submission Guidance

About IJIS

International Journal of Informatics Society (ISSN 1883-4566) is published in one volume of three issues a year. One should be a member of Informatics Society for the submission of the article at least. A submission article is reviewed at least two reviewer. The online version of the journal is available at the following site: http://www.infsoc.org.

Aims and Scope of Informatics Society

The evolution of informatics heralds a new information society. It provides more convenience to our life. Informatics and technologies have been integrated by various fields. For example, mathematics, linguistics, logics, engineering, and new fields will join it. Especially, we are continuing to maintain an awareness of informatics and communication convergence. Informatics Society is the organization that tries to develop informatics and technologies with this convergence. International Journal of Informatics Society (IJIS) is the journal of Informatics Society.

Areas of interest include, but are not limited to:

Computer supported cooperative work and groupware
Intelligent transport system
Distributed Computing
Multi-media communication
Information systems
Mobile computing
Ubiquitous computing

Instruction to Authors

For detailed instructions please refer to the Authors Corner on our Web site, http://www.infsoc.org/.

Submission of manuscripts: There is no limitation of page count as full papers, each of which will be subject to a full review process. An electronic, PDF-based submission of papers is mandatory. Download and use the LaTeX2e or Microsoft Word sample IJIS formats.

http://www.infsoc.org/IJIS-Format.pdf

LaTeX2e

LaTeX2e files (ZIP) http://www.infsoc.org/template_IJIS.zip

Microsoft Word™

Sample document http://www.infsoc.org/sample_IJIS.doc

Please send the PDF file of your paper to secretariat@infsoc.org with the following information:

Title, Author: Name (Affiliation), Name (Affiliation), Corresponding Author. Address, Tel, Fax, E-mail:

Copyright

For all copying, reprint, or republication permission, write to: Copyrights and Permissions Department, Informatics Society, secretariat@infsoc.org.

Publisher

Address: Informatics Laboratory, 3-41 Tsujimachi, Kitaku, Nagoya 462-0032, Japan
E-mail: secretariat@infsoc.org

CONTENTS