Time-Awareness System for Managing Activities

Samuel King Opoku[1], Samuel Otoo[2]

[1,2]Computer Science Department, Kumasi Technical University, Ghana

[1]samuelk.opoku@kpoly.edu.gh, samuel.k.opoku@gmail.com;
[2]slayyer9@gmail.com

Abstract— *Humans require reminder systems to prompt them into performing certain tasks. There are many reminder systems designed to help humans. These applications fail to recognize users' preference and interest. They, therefore, hardly handle multiple tasks at the same time. This study implements a time-awareness system that takes into account prioritization of multiple tasks. The application accepts the various activities of the user including their priority levels. When two or more applications crush, users' preference and interest are used to select and display the desired task. The application is implemented using Java Platform, Micro Edition*

Keywords—*Java ME, J2ME, Mobile Devices, Reminder System, Time-awareness System, Managing Activities*

1. Introduction

The use of devices to prompt humans in order to perform certain task has been predominant in our modern society. Mobile devices come with alarm clocks, to-do list, calendar and task to help prompt users to perform activities. Custom applications have been developed to serve the same purpose. The introduction of context-aware computing enriched the development of reminder systems based on contextual information. Applications rely on time solely as alarm clocks. Others combined time and location such as Schilit's work [1], [2]. Other applications combined different forms of context and other technologies to deliver the right prompt at the appropriate time and at the appropriate location [3]-[5]. All the above applications fail to recognize user's preference and interest. They also fail to handle different tasks that have the same priority. The work carried out in [6] ensures that user's preference and interest are taken into account. Unfortunately, the work was non-mobile and can only be used with desktop applications. This study implements time-awareness management system that prompts users regarding the task to be performed. When there are multiply activities to be executed at the same time, the activity with the highest priority based on user's preference and interest is displayed. The application is implemented using Java Platform Micro Edition (Java ME).

Java ME is a Java platform programming language for small and portable devices like cell phones, smart cards, personal organizers, palmtops and others. Java ME was built on three main concepts technology, namely, Configuration, Profile and Optional packages. These technologies determine the features of the API that can be used and how the application is packaged. The Profile is a set of selected Java library class collections from one or more virtual Application Programming Interface (API). The Java ME configurations have one or more associated profile and some profile may be dependent on other profile [7]. The configuration is a subset of java core API and java language functionality selected to provide a minimal java platform for a set of vertical market or specific domain. The configuration can be categorized into two and this includes: Connected Limited Device Configuration and the Connected Device Configuration. **Connected Limited Device Configuration (CLDC):** CLDC is aimed at targeting low ended devices like cell phones and PDA's. These devices usually have something in common. They have limited memory not exceeding 512KB, 16 bit or 32 bit processor and limited power supply. The type of memory usually used by CLDC is non-volatile and is used to store run time libraries and KVM. Volatile memory is used to located run time memory. The VM is provided by the CLDC and the core libraries used within the industry define profile is also set by the CLDC. **Connected Device Configuration (CDC)** defines the needs of those devices that fall between the CLDC specifications and the desktop specifications. CDC is built upon CLDC and can run CLDC applications. CDC application devices usually have 2MB of memory and 32 bit processor. Both configurations consist of JVM and core class libraries, because these devices have limited capabilities. It therefore makes it impossible to support or run all Java class libraries such as those with floating point numbers.

Some of the common profiles are PDA profile, foundation profile and mobile Information Device Profile. The profile is built on top of the configuration and they are specific to the size of the device on which the application will operate [8]. **PDA Profile** is similar to Mobile Information Device Application (MIDP) that, aimed at targeting PDA's with better screens and large memory than cellular phones. This profile which is still under construction is meant to give an enhanced functionality for interface class library and Java-based API for accessing some important futures of host operating system. When the profile is completed the probability of taken over MIDP is very high. **Foundation Profile** extends the CDC and is meant to be used as the ground foundation for most other CDC profile. **Mobile Information Device Profile (MIDP)** is built upon CLDC and its major aim is to target limited display and storage facilities of limited devices such as cellular phones and pagers. The optional package is a "set of API that does not define a complete application environment". It is used in conjunction with either a configuration or a profile to support devices capabilities that are not defined as part of the profile or needs to be shared by different profile.

2. System Architecture and Design

The user starts the system the first time by making a system registration which gives the user access to use the system. The user sets his or her preference and also creates activity in the to-do-list. The user can set due date and time; the number of times an activity should reoccur, set priority and then make some modifications after the activity has been created. Each time activity is created it is stored in the data store. The user can view an activity for the day or all the activities created in the notification. After an activity is created the user can organized the activities into folders and set priorities and change due dates. The search part allows the user to search for a specific activity which is yet to be executed. If the activity is found, it can be modified or deleted from the system. If no activity is found the system gives the user an option to create a new activity or exit the search. The system therefore keeps track of these activities, monitors them every minute and prompts the user when the time is due. The user can use the help and support part of the system if he or she is finding difficulty in using the system or needs a personal assistance. The use case diagram of the system is shown in the figure below

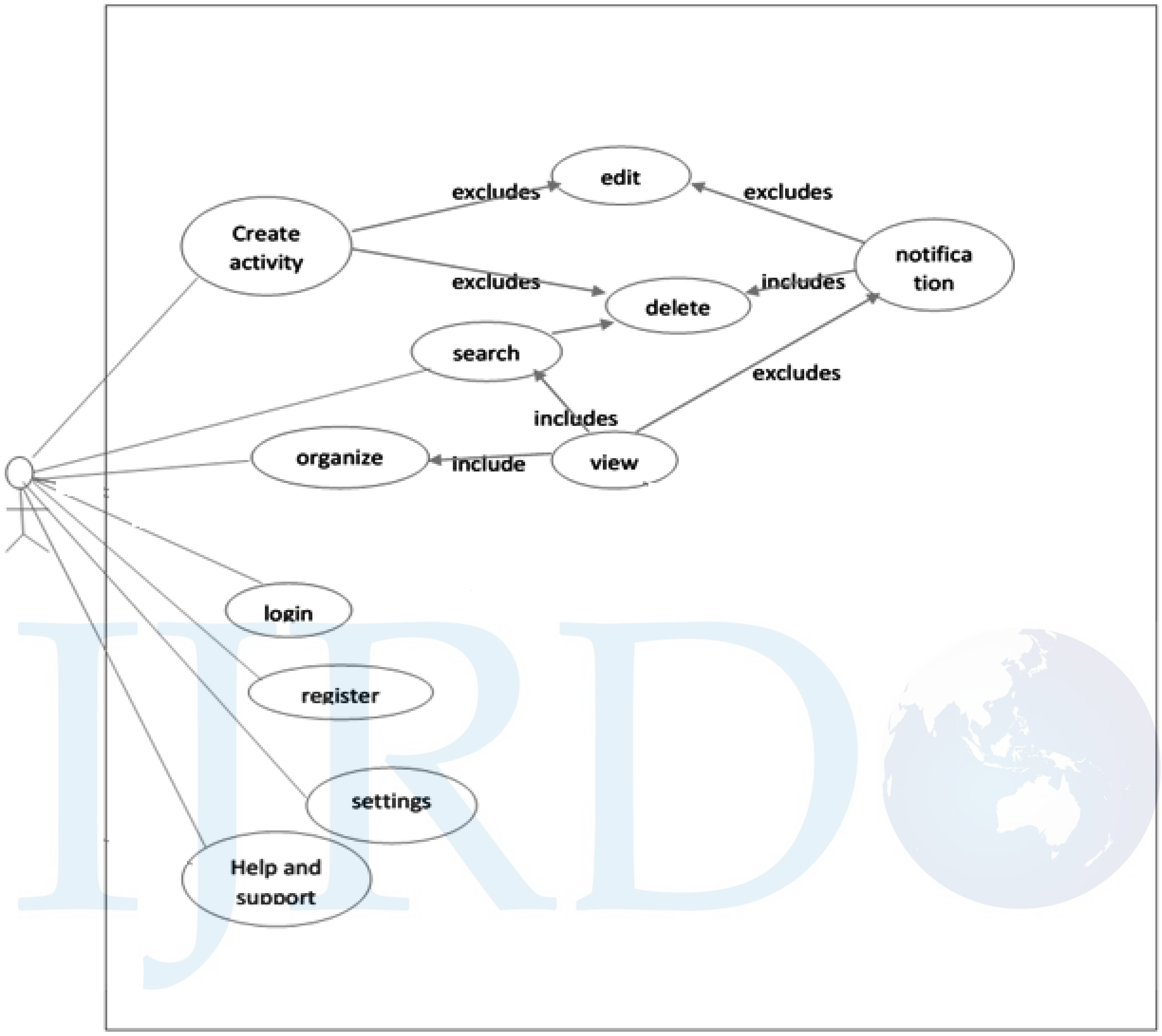


Fig 1: Use Case Diagram of System Interaction

From Fig 1, the **Register** is one of the major components of the system. Without registration, users cannot have access to use the system. The **User Help and System Support** gives user some useful guidelines on how to use the system. It also provides a link to the manufacturer of the system for personal assistance in case the information provided is not adequate. Another component of the system is **To-do-list.** It is the most important part of the system. All the operations of the system is centred in this area. In this section the user creates the activities that he or she wants to perform and then set time durations and add other complimentary notes. **Settings**, gives the user some privileges to customize the system. The user can change the appearance of the system or reset it to the original default settings. This section also allows users to set their preferences. **Notification** aspect of the system performs only one major function that is to notify or prompt the user when an activity is due. The **Search** part of the system allows the user to search, modify and delete an activity. It also gives the user the option to create a new activity if an activity is not found. **Organization** part of the system allows the user to organize the created activities into categories of his or her choice. The user can view all the activities created in the organization area. Once the application is installed, a folder is created in the mobile device internal memory. For every activity created will be stored in a file inside the folder which serves as a data store. The activities are stored in a form of table separated by tabs. Each created activity is stored in a single row, the tabs determines the columns or fields of each activity. Each time an activity is modified it will override the previous activity created. The use of text file data store conserves storage space of the mobile device as these files do not consume storage space. The various sections of the system share data. Fig 2 below depicts the data flow diagram of the system.

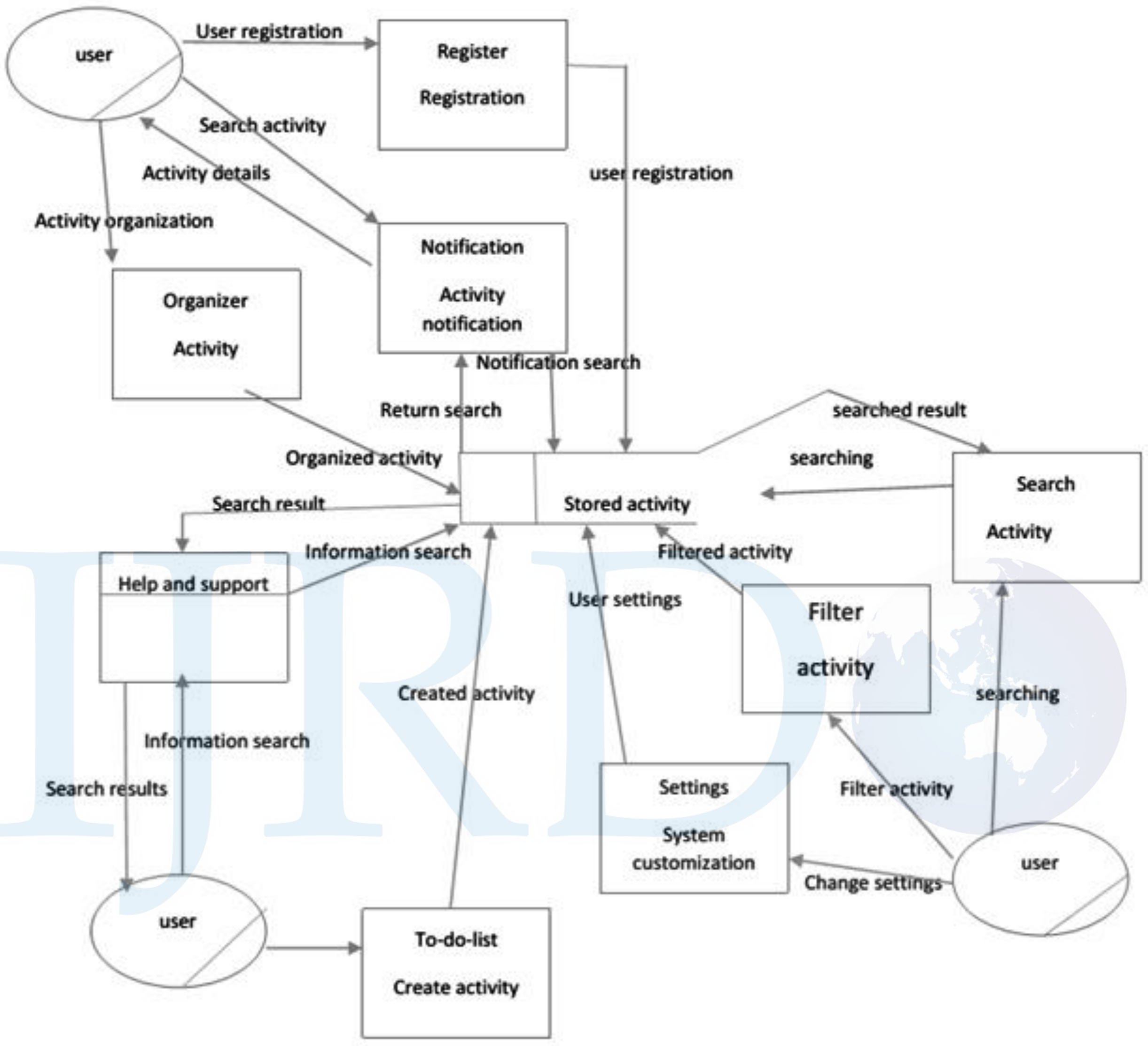


Fig 2: Data Flow Diagram of the System

From Fig 2 above, all the processes connect to the user and the data store by fetching and serving data. Fig 3 below illustrates the logic behind operations of the system. Flowchart is used in this instance as depicted below

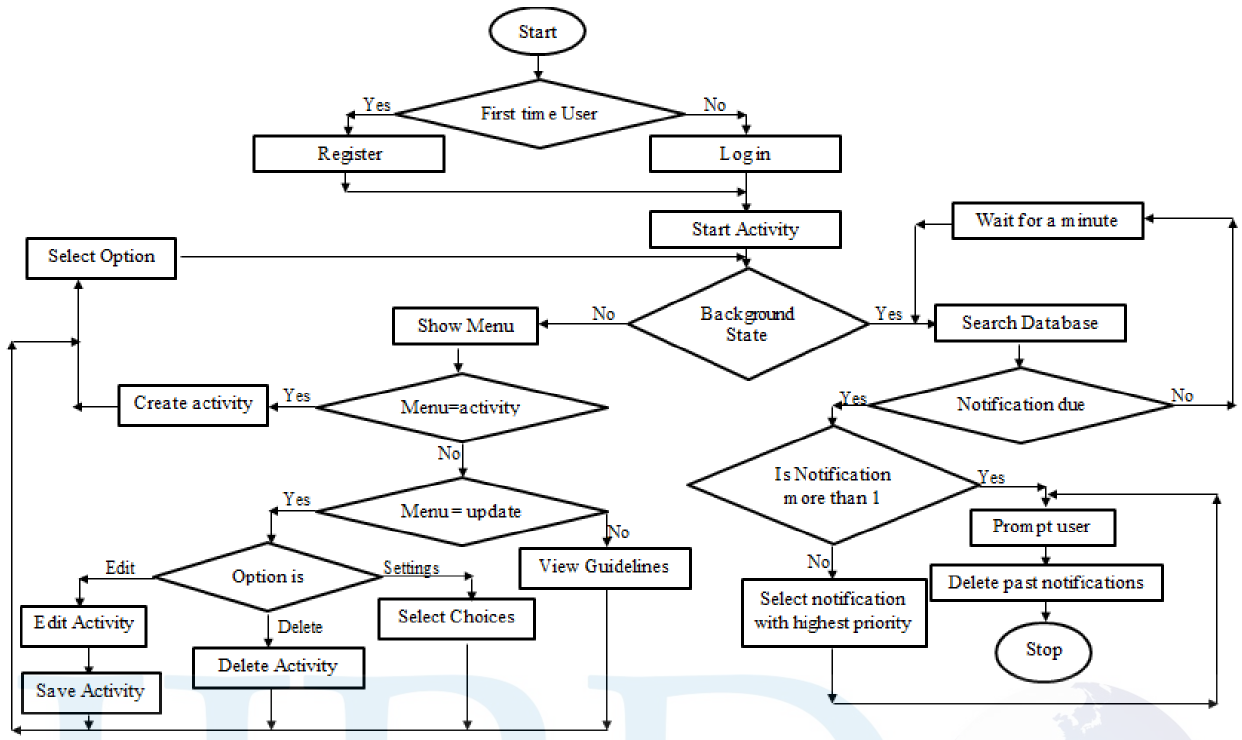


Fig 3: Flowchart of the System

Two or more activities are said to be clashed if they all have the same due date and time. From Fig 3, if there is a clash of activities the system prompt the user based on set priorities. When the clashing activities are of the same priority, the settings of the application take precedence. If per the settings, religious activities supersede secular activities then religious activity is displayed otherwise secular activity is displayed. If the clashing activities belong to the same category, the activities are combined and displayed.

3. System Implementation and Testing

The implemented system is developed using CLDC and MIDP technologies of Java ME. Thus the system can be deployed on mobile phones. The testing system is called STULECT which comprises of different classes that are defined to perform a specific function. The classes are linked together to form the system. The functions performed by these classes are:

- **Stulect Class** is the MIDlet class of the system which starts the execution of the entire application. Without this class the application cannot run. All the other subclasses, except MyMethod class, are linked to this class and these subclasses define the various aspect of the application.
- **ActivityForm Class** defines the activity form interface. The purpose of this class is to allow the user to create activities. The user will be prompted when the time is due and the user has the chance of setting priority for each activity created.
- **ActivityUpdate Class** is similar to the ActivityForm class. The only difference is that user has the chance to edit or delete existing activity after it has been searched.
- **Settings Class** defines the interface for users to select their preferred choice.
- **MenuList Class** defines a menu list interface of the system. Each list item opens a different interface of the application.
- **MyMethod Class** consists of only methods that are used by other classes. The function defined in this class allows delete command or button and the add command or button to function as expected.
- **Guidelines Class** contains user guideline information on how to use the system and a link to an online support.

- **UserAccountForm Class** defines the system lock interface. This class allows the user to create an account to protect his or her privacy. With this class the user can lock and unlock the system after the account is created.

The various interfaces illustrate the implementation of some of the classes of the system. Fig 4 illustrates the various items on the menu list shown below.

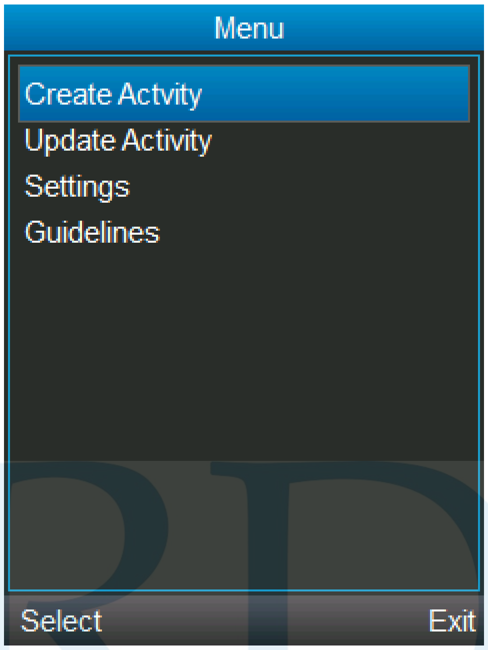


Fig 4: Menu Interface of the System

The menu interface is obtained after a successful login by the user. Fig 5 shows the activity form. Users provide the following information which allows the system to keep track of users' activities and display the necessary one when the time is due.

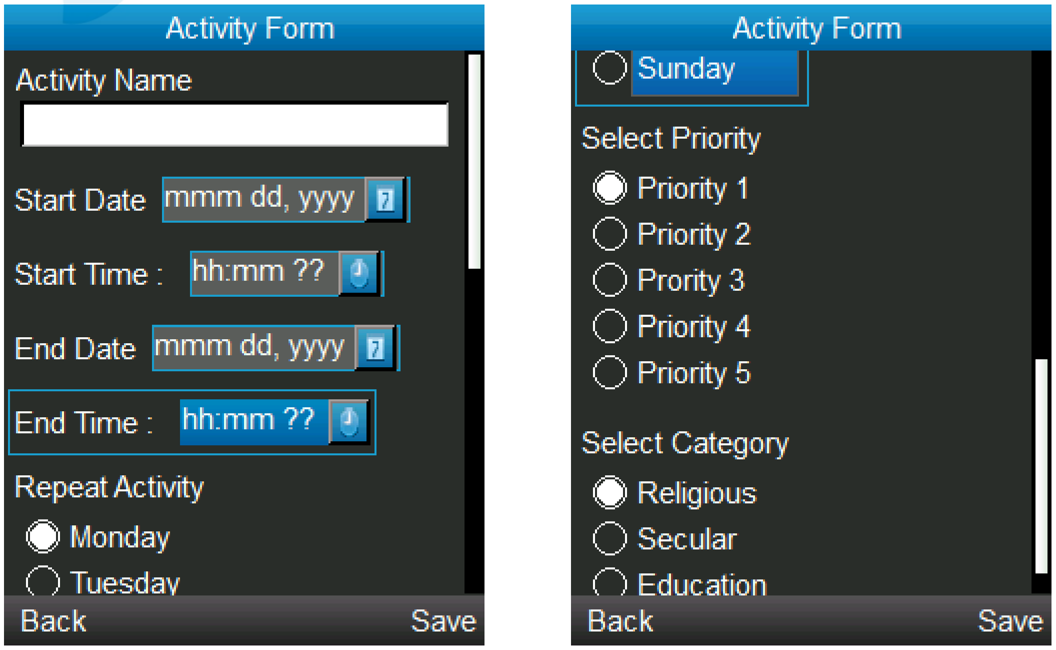
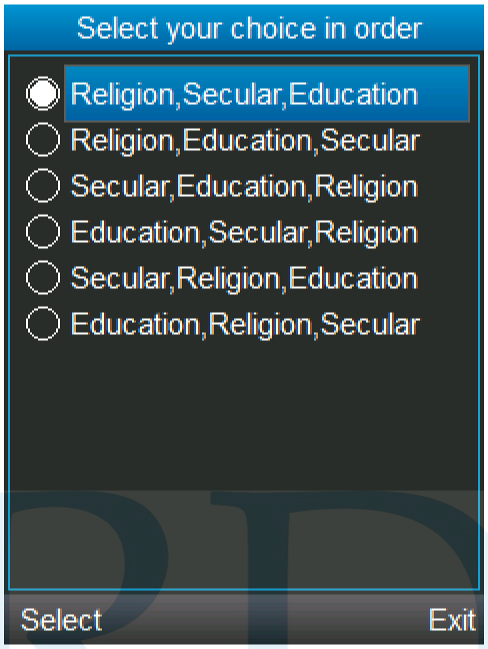


Fig 5: Activity Entry Interface of the System

The activity name, date, start time and end time constitute the composite keys of an activity to be saved into the data store. These keys put together, ensures that all activities are uniquely saved. Fig 6, depicts the various preferences the user can select. There are three preference items arranged in order of precedence starting with the one with the highest priority as they will be ranked by the user. The three preference items are religion, secular and education.


Select your choice in order

[✓ Religion,Secular,Education]
Religion,Education,Secular
Secular,Education,Religion
Education,Secular,Religion
Secular,Religion,Education
Education,Religion,Secular

Fig 6: Preference Setting of the System

Fig 7 illustrates how a stored activity will be displayed when the time is due.

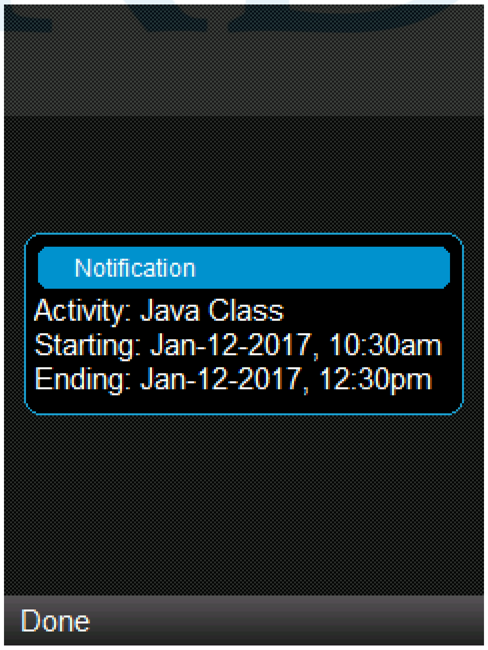

Notification
Activity: Java Class
Starting: Jan-12-2017, 10:30am
Ending: Jan-12-2017, 12:30pm

Fig 7: Sample Prompting Interface

Prompts consist of the start time, end time, day and activity description. This enables the user to remember precisely the task that needs to be carried out.

4. Conclusion

A mobile application is designed to prompt the user into taking activities with highest priority based on the dynamics of users' interest and preference. The application can be deployed on Symbian OS devices, since they widely support Java ME. The system is simple to use. Users provide the information required for system functioning. The system runs background and only displays alert when notification is due. This does not affect the activities of the system. The system also conserves memory by deleting old activities whose time has been overdue. Future work should focus on using machine learning approach to determine the probabilistic behaviour of the prompting activities. Also, there are different activities within each category with different priority levels. The effect of the dynamics of the various activities under a category against the activities of different categories should be considered for further studies.

References

[1] Opoku S. K., Awisie E. T., 2015, "*A mobile phone based mechanism for designing an announcement system*", presented at 1st Faculty of Applied Science Academic Paper Presentation, Kumasi. 14th January

[2] Schilit B. N., 1995, "*System architecture for context-aware mobile computing*", PhD Dissertation Columba University, New York

[3] Opoku S. K., Awisie E. T., 2015, "*An event notification system for authorization and available organization members*", International Journal of Advances in Computer Science and Technology (IJACST), 4(12): 171-175

[4] Dey A. K., Abowd G. D., 2000, "*A context-aware system for supporting reminders*". Proceedings of the 2nd International Symposium on Handheld and Ubiquitous Computing (HUC), pp 172-186

[5] Ludford P., Frankowski D., Reily K., Terveen L. G, 2006, "*Because I carry my cell phone anyway: Improving location-based reminder systems*". In proceedings of ACM conference on Human Factors in Computing Systems (CHI) pp.889-898

[6] Opoku S. K., Appiah S., 2016, "*Automating students' activities in Higher Institutions*", International Journal of Computer Applications Technology and Research, 5(11): 693-697

[7] Beji S., Kadhi E. N., 2008, "*An Overview of Mobile Applications Architecture and the Associated Technologies*", Fourth International Conference on Wireless and Mobile Communications, ICWMC '08, IEEE, p. 77 – 83

[8] Yuan M. J., 2006, "*Entreprise J2ME, Developing Mobile Java Applications*", Upper Saddle River: Prentice Hall PTR, pp. 20-25