**Context-aware Battery Management for Mobile Phones: A Feasibility Study**

**(CABMAN)**

Objective:

The main objective is to inform the user if his battery would actually sustain until he reaches the next charging location with all the current applications running and also taking into account the fact that he may have to make or attend some phone calls.

For instance, if the user has around 10% battery left and he is pretty close to his home and his heading right there, the mobile should not actually perturb the user by display warnings regarding the low battery level.

Sub-problems tackled:

Three main sub problems exist in order to actually achieve the objective.

1. Predicting the next charging location
2. Predicting the call time or the number of minutes every hour the user is going to spend for calls.
3. Predicting the battery level remaining until the user reaches the next charging location with the current applications running

Predicting the next charging location:

Every mobile is connected to a cell tower and this is indicated by the cell id.

Now the proposed solution is an application that keeps a track of all the cell ids that appear on the mobile device and marks the ones where charging is taking place. For almost every individual, there is a certain fixed schedule so once these cell ids are marked, the charging locations have been determined unless the user finds a new charging point which would be added to this list in that case.

Now the current cell movement pattern (a set of cell ids) is matched with the previously stored history of cell movement patterns and using that the next set of cell ids is predicted if a similar case exists. Now using these predicted cell ids, one can check which ones are marked and thus determine the charging point. Even the time until next charging point can be predicted since each cell id in the history of cell movement patterns is stored along with a timestamp. So the time until the next charging point can thus easily be detected. If the matching cases are multiple, an average value of time until next charging point is taken.

Cell ids keep changing even for a stationary mobile device but they change in an ordered fashion such that the same cell ids keep appearing repetitively and this is used to assert that the cell is stationary.

Predicting the call time:

Similar to charging this also uses past calling behavior to find the average number of minutes of call time that the user needs during each hour of the day, This is used to get an upper bound on the total call time required within a given time interval.

The call time in minutes for each hour in the day is determined by taking an average of the call time for the same hour that exists for the past call traces.

Predicting the battery level:

Plotting a base curve, which corresponds to how the battery is depleted over time when the device is completely idle, does this. Most batteries age with time and their performance generally degrades so this base curve has to be made periodically when the user is not using the mobile device to achieve greater accuracy.

Then another curve for the current set of applications running is generated showing the battery usage over time. This curve is extrapolated and then using these two curves a discharge speed up factor is calculated.

For instance if the base curve has battery c1 at time t1 and c2 at time t2 and the curve for applications running has the same battery capacities at time t3 and t4, then the discharge speed up factor would be:

(t2-t1)/(t4-t3) where t2 > t1, t4> t3, c2 >c1

Then the current battery capacity is multiplied by this discharge speed up factor to determine the duration for which it would actually last.

Platform:

CABMAN prototype was implemented for Linux using Java, Perl, shell- scripts and C++.

Putting it all together:

From the remaining battery time predicted using the above method, the battery consumption for calling based on the call minutes per hour predicted are deducted and then it is checked if the mobile battery would sustain until the next charging point is reached. If that is the case, then user is not given any warnings else the user is issued a warning to carry a charger along or so that he can kill applications or use them conservatively.

Error rates: The error rates using these algorithms were found out to be minimal which makes the CABMAN model pretty feasible.