An Empirical Approach for Determining Context of Mobile Systems

Mai Abusair

Antinisca Di Marco

Paola Inverardi

University of L'Aquila, Italy

Outline

- Research idea and contribution
- Definition of context, context variables, context-awareness
- Empirical Approach for deriving contextual situations.
- Introducing Context Sensing Application CSA
- Modeling context variables and contextual situations
- Applicable examples
- Conclusions and Future Work

Research General Idea

- Context awareness increasingly becomes an essential attribute for software systems.
- Mobile applications may benefit from context awareness since they incur to context changes during their execution.
- Mobile applications can adapt their structure and behavior as a way to preserve the service quality they offer under the different contexts.

Research Contributions

- Introducing an approach for capturing the context variability of mobile applications, starting from monitoring, through a sensing app, context variables values.
- Learning, from the monitored data, context variables models and deducing from them the contextual situations.
- Deducing, other knowledge including, the user behavior that identifies specific user profile.

Context Definitions

- "**Context** is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves."

- **Context variable** is any type of contextual information affecting the system behavior.

- **Contextual Situation** can be defined by a group of context variables and their values, under which a system will eventually run.

- "A system is **context-aware** if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task. "

Context Variables Categories

Operational categorization identifies context variables according to how they are attained, modeled, and treated [1]

Conceptual categorization is the one that categorizes context variables according to their meaning.[1]

Sophistication categorization is a refinement for operational categorization to categorize the context variables based on the level of sophistication to achieve them [2].

Operational Categorization of Context

Conceptual Categorization of Context

	Sensed	Static	Profiled	Derived	
User	User Blood pressure	Gender (male, Female), user abilities	Preferred language, age	Time user spent on the application per day	Biometrics, Conversational voice
Computing Platform and device infrastructure	Type of network connection, battery level, Network Speed	Device type (Samsung, LG..), CPU of device, RAM of device	Automatic mobile network operator switching , sensing location information every 15 seconds	CPU usage for application	Gesture-based control
Environment	Weather temperature, time, Location	Wide angle of detection camera, CPU of server, RAM of server	Changing height of weather sensor location	Crowd detection , Server workload	Augmented reality
	Basic level			**Intelligent level**	**advanced intelligent level**

Sophistication categorization of context

[1] A. H. Van Bunningen, L. Feng, and P. M. Apers, "Context for ubiquitous data management," in Ubiquitous Data Management, 2005. UDM 2005. International Workshop on. IEEE, 2005, pp. 17–24.

[2] B. J. A. Ask, "The future of mobile ebusiness is context." forrester, 2012.

Context Variables

$C \in \{val_1, val_2, \ldots val_z\}$ for discrete values

$C \in \{ [val_a, val_b],[val_c, val_d], .. \}$ for range of values

Example:

$B \in \{SufficientBattery, NonSufficientBattery\}$

$CN \in \{AvailableConnectivity, NonAvailableConnectivity\}$

Contextual Situations

$S = \{ C1(val_{C1}), C2(val_{C2}), \ldots, Cx(val_{Cx}) \}$

Example:

S1={B(SufficientBattery),CN(AvailableConnectivity)}

S2={B(SufficientBattery),CN(NonAvailableConnectivity)}

S3={B(NonSufficientBattery),CN(AvailableConnectivity)}

S4={B(NonSufficientBattery),CN(NonAvailableConnectivity)}

Empirical Approach to Context Modeling

- To understand context variability, we introduce an empirical approach, based on an Android mobile application that monitors the mobile phone context variables.

- By analyzing the monitored data, we are able to model the context variables and the contextual situations as UML Statecharts.

Empirical Approach to Context Modeling (Cont.)

- Context modeling can be used for analyzing QoS of mobile systems, determining the different contextual situations under which one needs to study the system behaviors and deciding at which contextual situation to adapt.

1

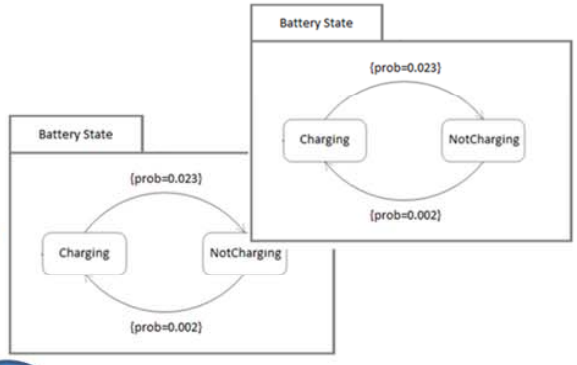
Run CSA application

2

Share monitored data

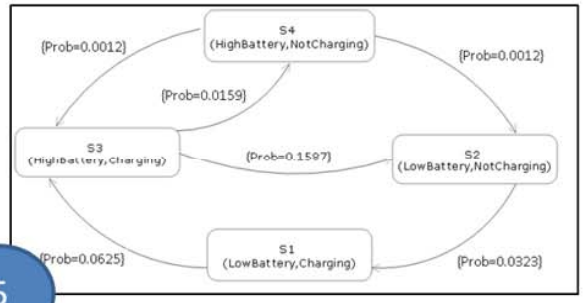
3

Analysis of Context Data



4

Context Variable Modeling



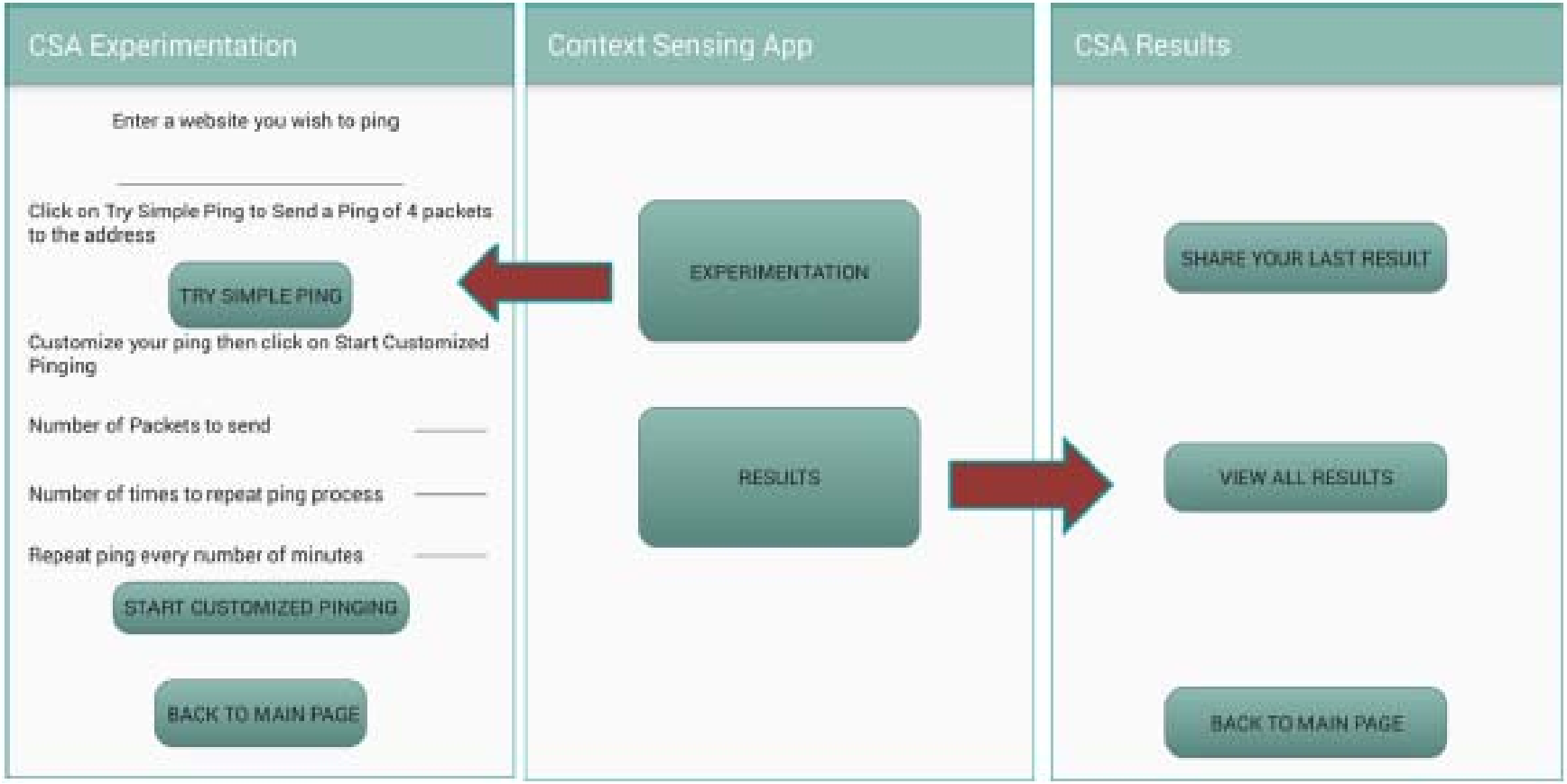
5

Contextual Situation Modeling

Context Sensing Application CSA

- It is developed for Android operating systems. It can run in the background.
- It is able to sense context variables and log the information in a text file stored locally on the mobile internal memory.

CSA (Cont.)


CSA Experimentation
Enter a website you wish to ping
Click on Try Simple Ping to Send a Ping of 4 packets to the address
TRY SIMPLE PING
Customize your ping then click on Start Customized Pinging
Number of Packets to send
Number of times to repeat ping process
Repeat ping every number of minutes
START CUSTOMIZED PINGING
BACK TO MAIN PAGE
Context Sensing App
EXPERIMENTATION
RESULTS
CSA Results
SHARE YOUR LAST RESULT
VIEW ALL RESULTS
BACK TO MAIN PAGE

Screen Shot of CSA results

```
Ping Request Num.: 3
Ping State: Normal
Device ID: 353768079510702
Time: 2017-06-14 11:46:41
Pinging: offsiteart.it
location: 42.362269,13.369942
Light level: 10870.0 / 60000.0
Connection Info: mobile network: 3G HSPA+  I WIND
signal strength:  -71 dbm
Battery Level: 53.0
Battery State: NotCharging
ping success: 0% packet loss
Packets sent: 4
Packets received: 4
avg round trip: 394.282 ms
```

Context Variables Modeling

- To model the context variables retrieved by CSA.
- Each type of context variable is handled by using a UML Statechart where a state represents the current context variable value at the time when users demand for services.
- The transition probabilities are then calculated with the following formula:

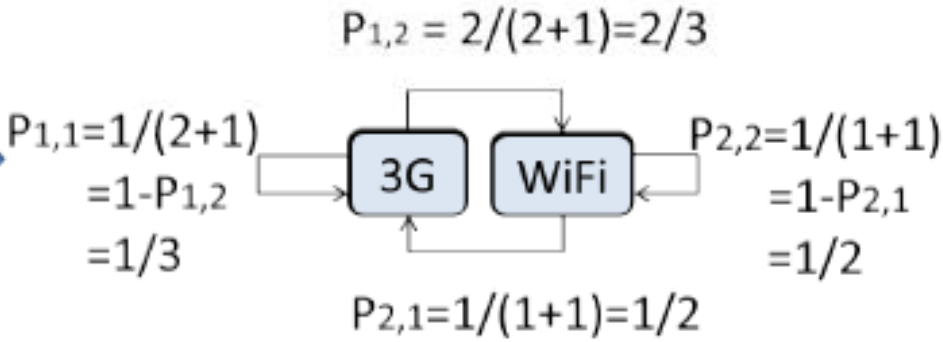
$$p_{s_i,s_j} = \frac{\#(v_i \rightarrow v_j)}{\#(v_i \rightarrow v_i) + \#(v_i \rightarrow v_j)}$$

An example on computing transition probabilities

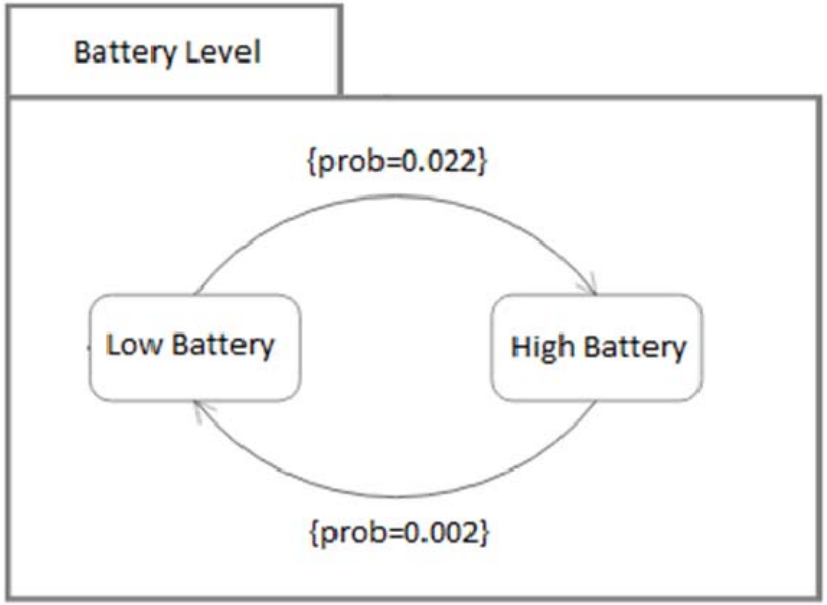
Time	Network
1:00	3G
2:00	3G
3:00	WiFi
4:00	3G
5:00	WiFi
6:00	WiFi

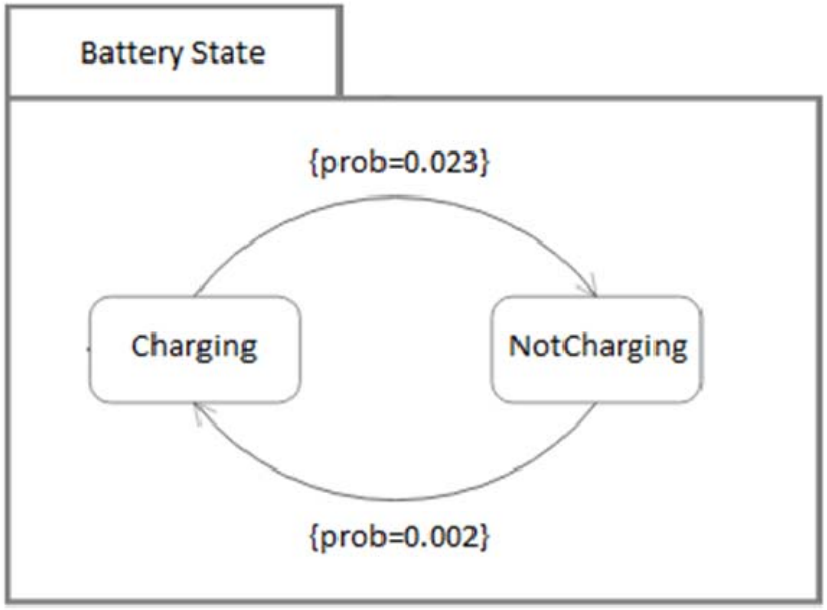
If
V_1= 3G
V_2=WiFi

Then:
#(V_1->V_2) = 2
#(V_1->V_1) = 1
#(V_2->V_1)=1
#(V_2->V_2)=1



Battery Level and Battery State Context Variables Model Example



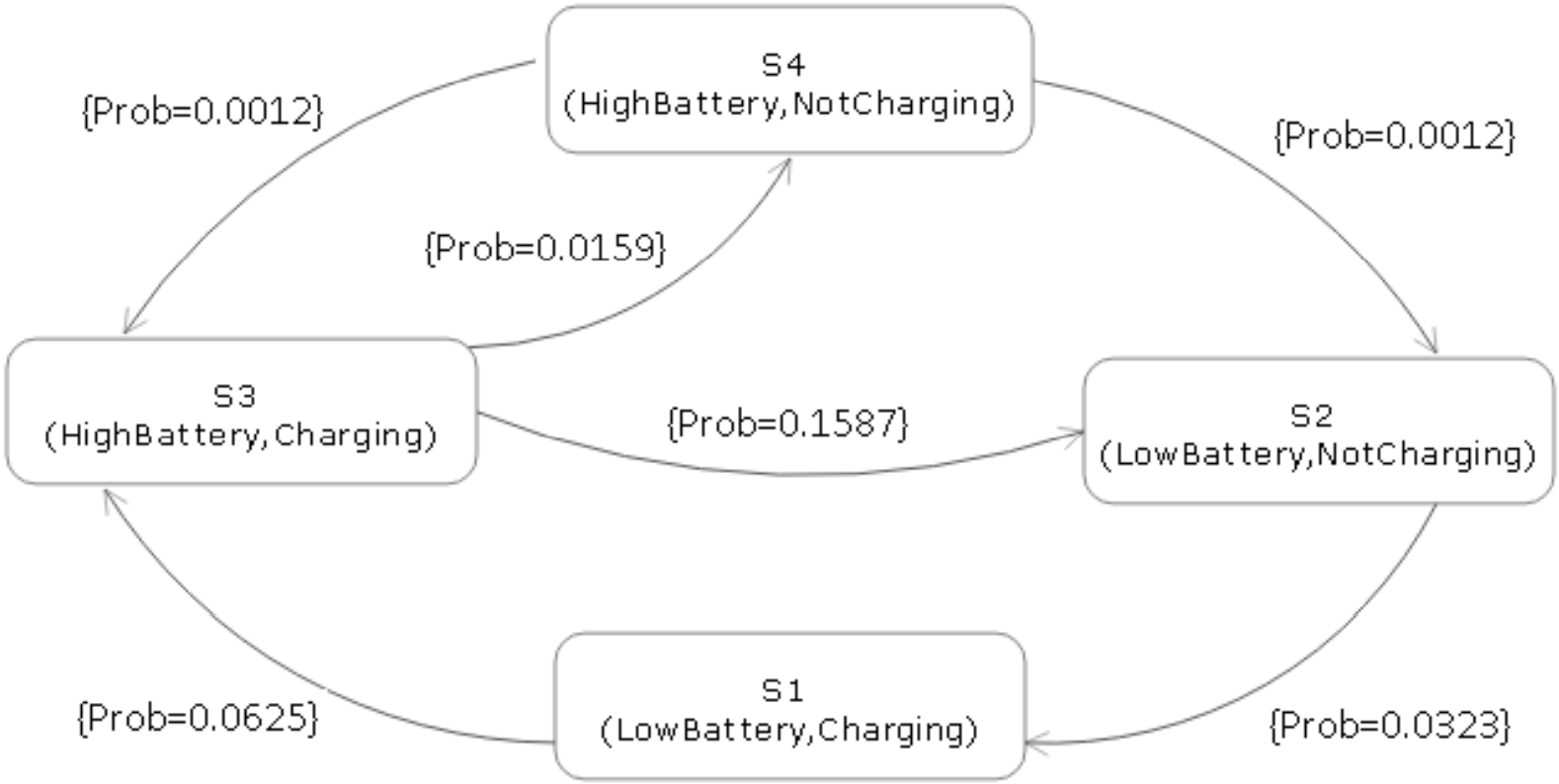


Contextual Situations Modeling

- The statecharts derived for the context variables are lumped into one statechart, called contextual situation model, that models the runtime context evolution of a mobile context-aware software system.

- Each state in the contextual situation model, is a, super state, obtained from the combination of certain number of states.

Contextual Situation Model Example



Deriving User Behavior

- By analyzing the data logged by CSA, we found that it is possible to extract from them additional information.

- In particular, reasoning on values retrieved for a given user, it is possible to derive insights on user daily behavior and activities.

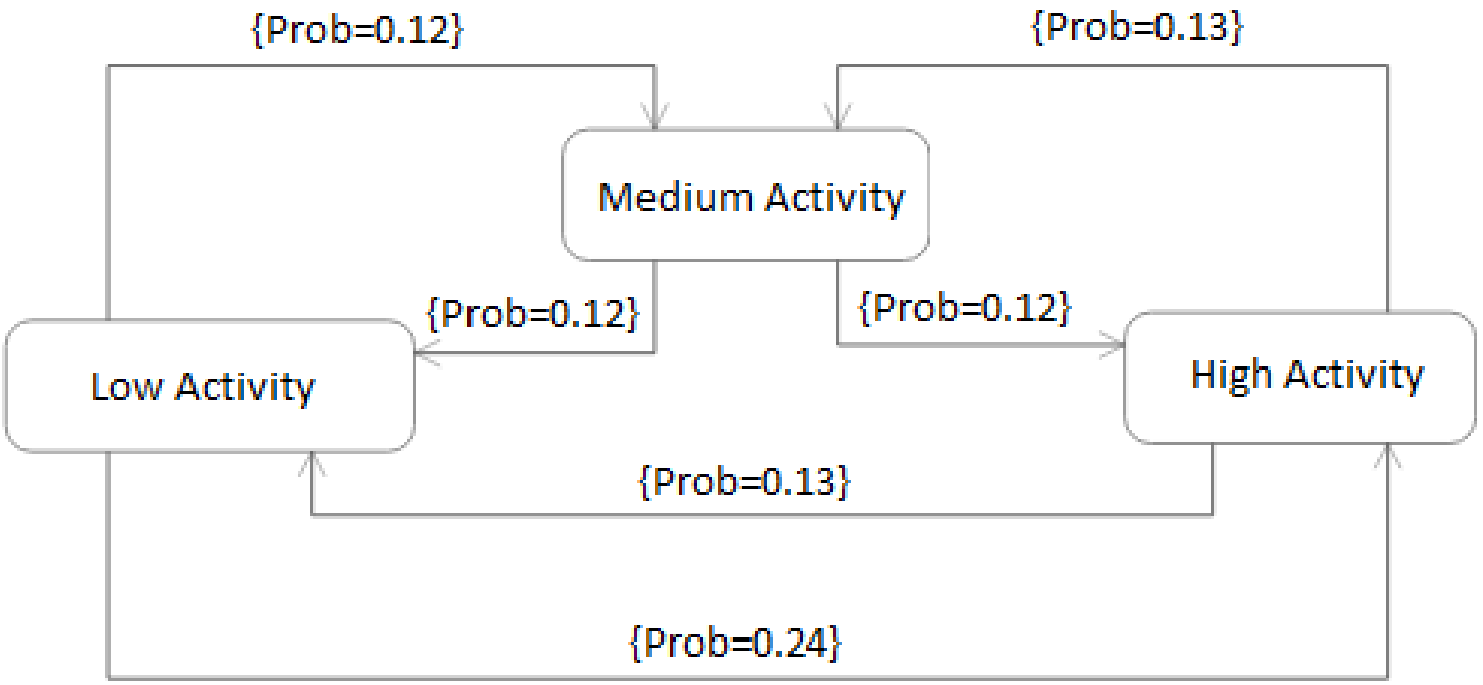
Deriving User Behavior Example

When we reason on the battery charging and level degradation over time, we discovered two user behaviors:

- The mobile user, who performed the test, charges her mobile phone in an average of 1.2 hours per day. We conclude this from the time the user device keeps plugged and unplugged during the running period of the CSA app.

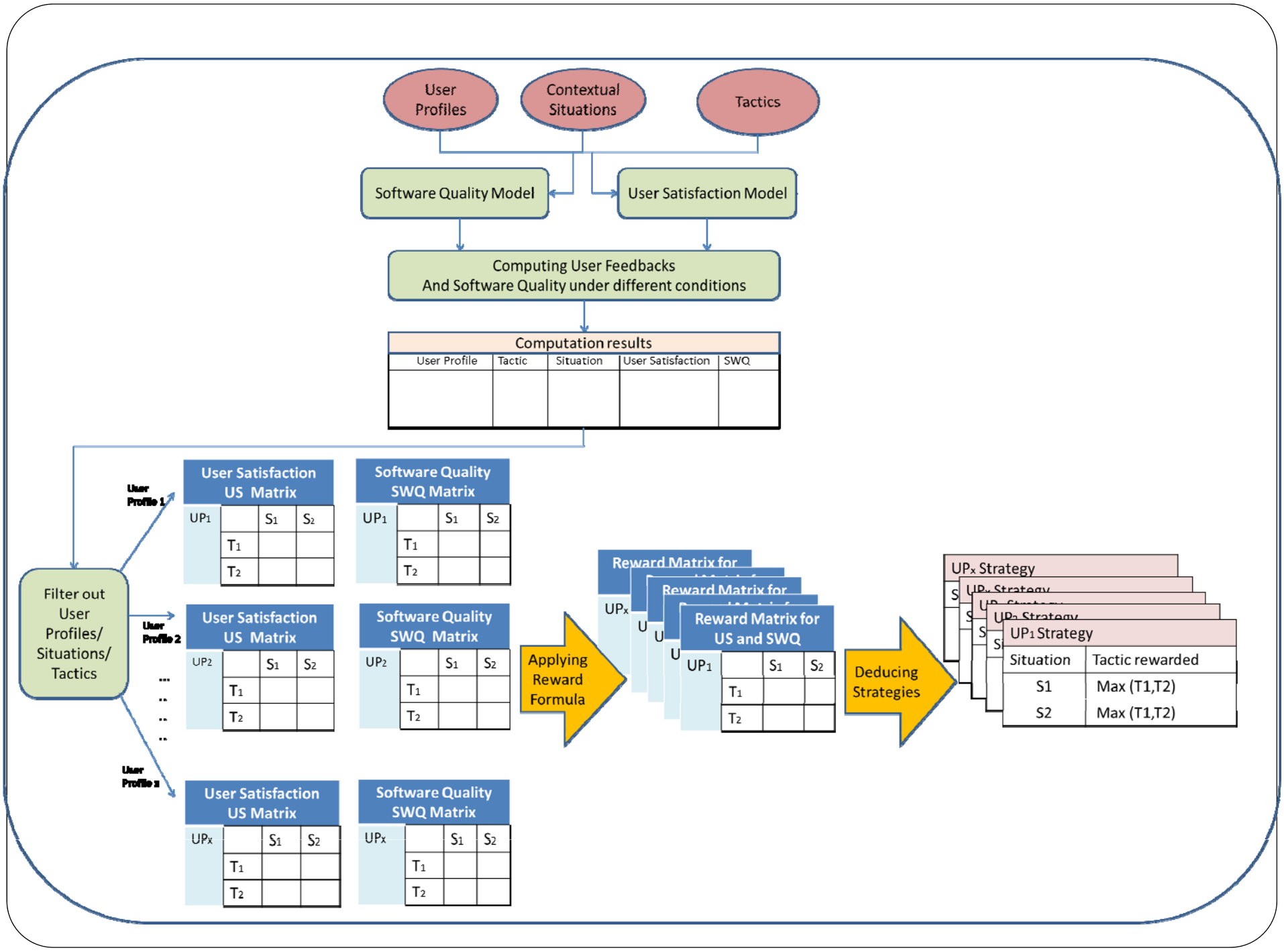
- The mobile user has a noticeable different kind of activities while she uses her mobile phone. We captured, from the battery degradation occurred during the day, periods where the battery level drawn faster than other periods.

User Behavior Example



Applicable Example For The Method

- We use contextual situations to analyze at each state the mobile application behaviors in terms of service availability and user satisfaction, and to determine consequently the best adaptation at each contextual situation.


User Profiles
Contextual Situations
Tactics
Software Quality Model
User Satisfaction Model
Computing User Feedbacks And Software Quality under different conditions
Computation results
User Profile
Tactic
Situation
User Satisfaction
SWQ
Filter out User Profiles/ Situations/ Tactics
User Profile 1
User Profile 2
User Profile x
User Satisfaction US Matrix
Software Quality SWQ Matrix
UP1
UP2
UPx
S1
S2
T1
T2
Applying Reward Formula
Reward Matrix for US and SWQ
Deducing Strategies
UPx Strategy
UP1 Strategy
Situation
Tactic rewarded
S1
Max (T1,T2)
S2
Max (T1,T2)

Conclusion

- We have introduced context sensing application CSA, to be used in a framework for monitoring, modeling context variables and contextual situations in mobile software systems.
- The type of contexts can be modeled as statecharts whose states and transitions are based on collected context variables values.
- Determining contextual situations transitions that represents the context changes at a time, is essential for analyzing the system quality when it goes under these changes, it can also help in deciding the context awareness to be considered when designing the adaptation.
- The results show that from the context evolution one can conclude the user behavior that can help in describing the user profile.

Thanks for listening!!!