The approach to predict operational time for Symbian mobile devices

Vera Kononova, Kirill Krinkin

OSLL/SPbETU

6th FRUCT Seminar,
Helsinki, November 2009

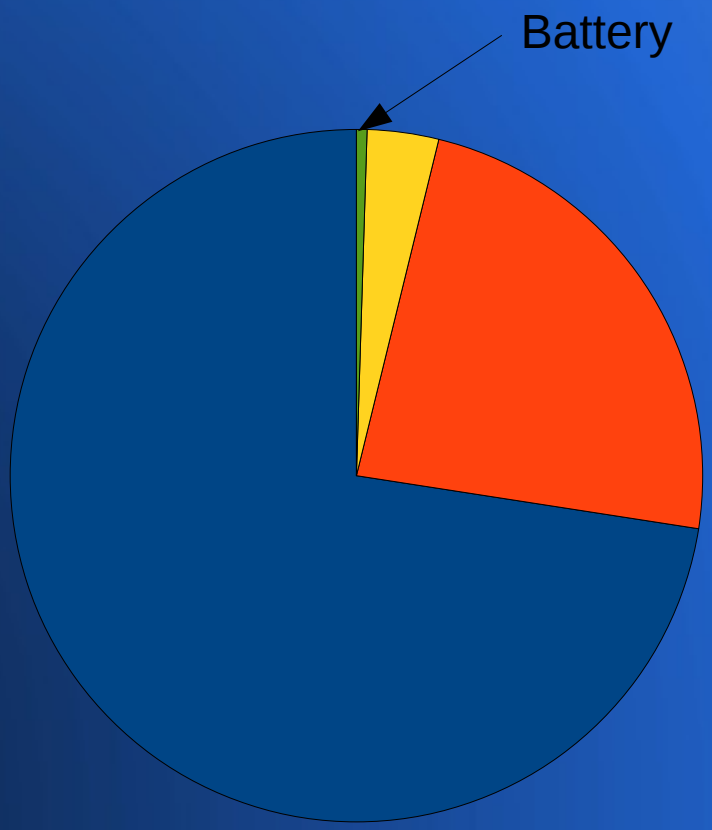
Agenda

- Motivation and technology overview;
- Short look at the EPM Framework;
- Project (plug-in) goals;
- Algorithm definition;
- Implementation design;
- Results and conclusion.

Technology increase

Parameter	Increase ratio*
Storage size	1200
CPU Performance	393
Memory size	128
Network performance	18
Battery capacity	2.7

***) since 1991 till 2001**



Motivation

- There are technologies lags for power sources;
- There's no mobile operation time predictor for mobile phones;
- NRC people are working on EPM Framework;
- EPM contest has been announced during 5th FRUCT;

EPM Framework

- Provides event driven environment and plug-in based extensibility;
- Designed as platform independent;
- Is being implemented for Symbian S60;
- Provides high level API for python;
- Provides access to number of measurements and parameters.

Available measurements

- current and voltage measurements;
- remaining mAh in battery;
- cell ID (e.g. home, work, home city);
- foreground application and running application list;
- current cellular network voice (2G or 3G) and data usage (2/2.5/3/3.5G);
- backlight On/Off and Charger In/Out/Charging/NotCharging;
- wireless signal strengths (2G, 3G, WLAN, BT, GPS, DVB-H)
- installed applications (listed when installed)

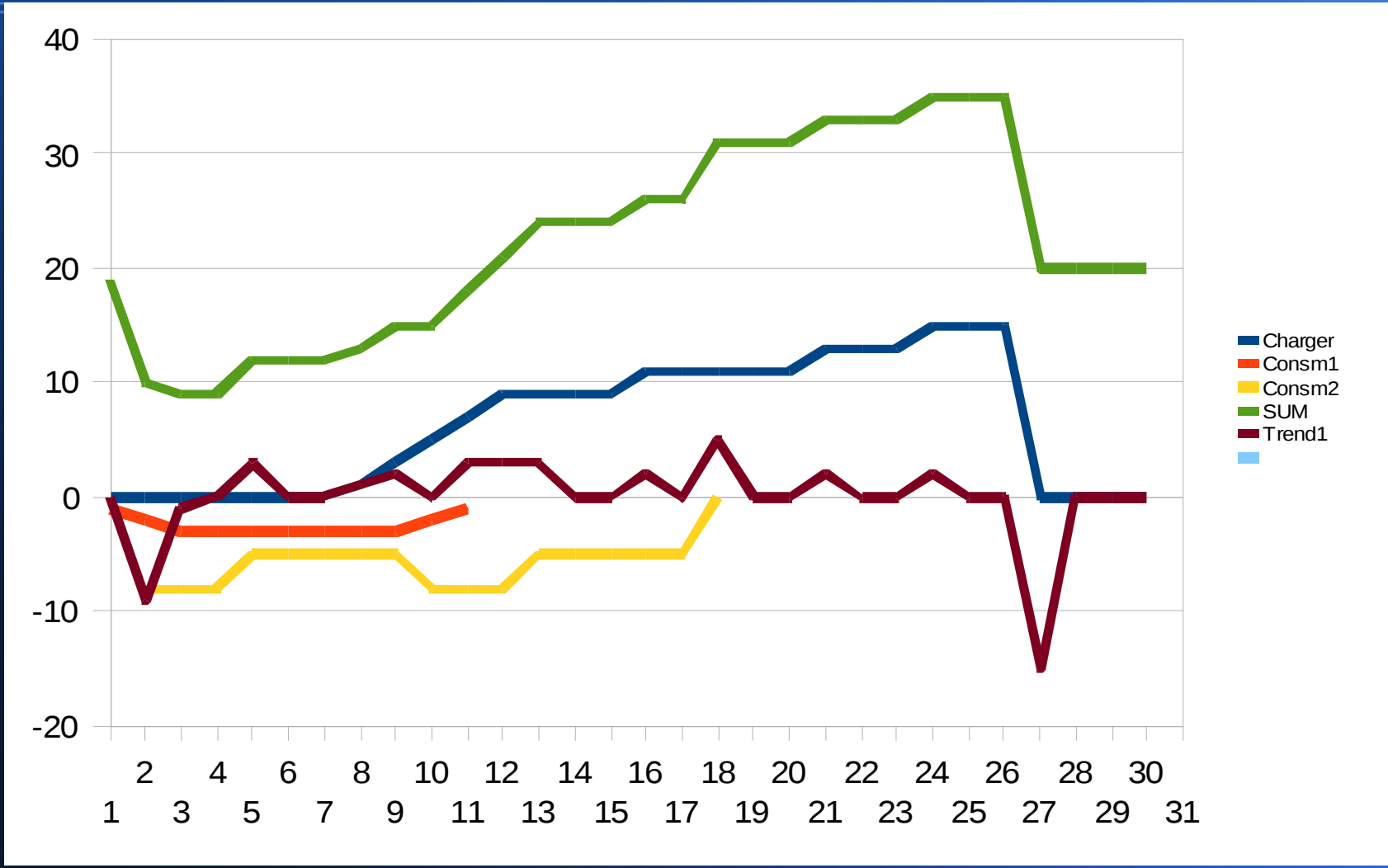
Project goals and stages

- Main goal:
  - To develop an approach to predict mobile phone operational time and implement predictor plug-in for EPM Framework
- Project stages:
  - May-Jun 2009 algorithm definition;
  - Jul-Sep 2009 prototyping;
  - Sep-Dec 2009 algorithm tailoring and improving
- Team:
  - 1 Postgraduate, 2 students;

Terms

- *Operational profile* – application parameter set for particular environment (like “Home”, “Work” “Business trip”, “Traveling” and so on);
- *Consuming profile* – set of parameters which defines a one power consume operation/event (e.g. phone call, 3G modem session,...);
- *Application frequency* – freq of application runs in concrete Operational profile.

Power consuming and charging



Math view

The power consumption function looks as next sum

$$P_c = \sum_i p_i(t)$$

Where:

- p_i (t) – *contribution of particular application (power consumer or charger) to the power spending picture for one action (e.g. phone call);*

The Goal:

- *To find a* ***t*** *value when* $P_c = 0$

Approximation for p(t)

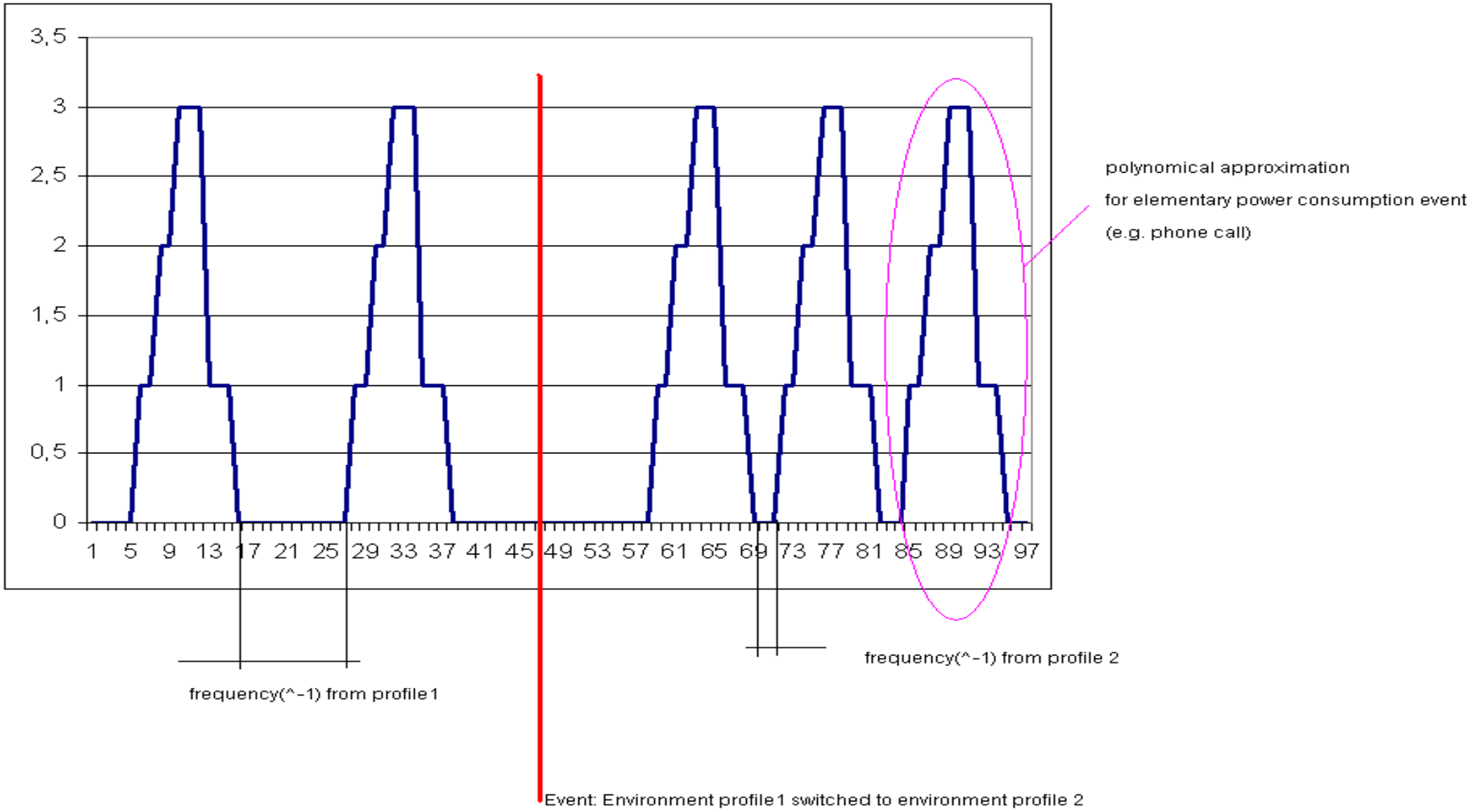
- p(t) could be presented as 2rd order polynomial like

  $p(t) = at^2+bt+c,$

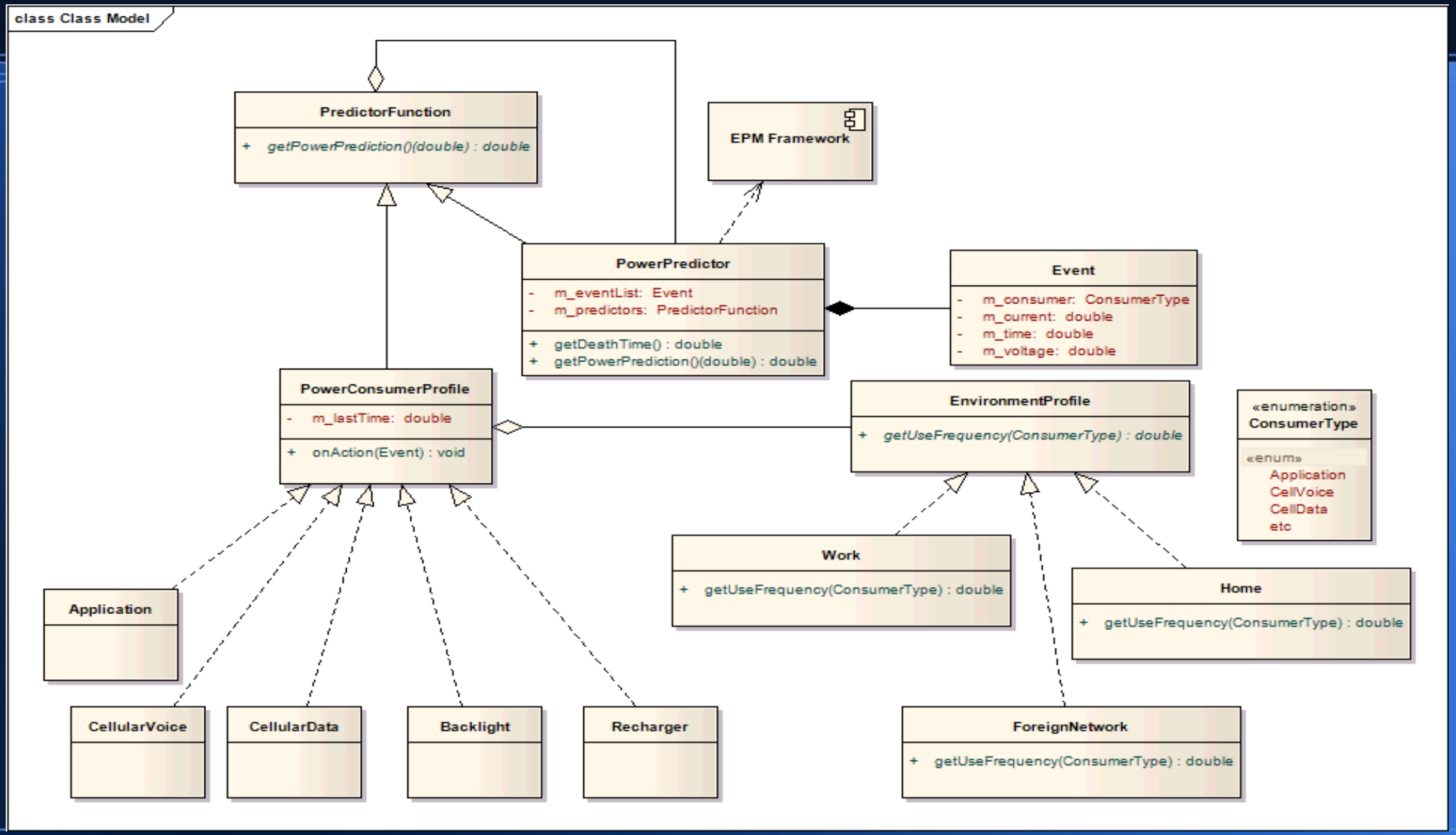
  where:

  - a,b,c should be corrected by the least square method;
- In easiest case p(t) is a constant;

Profile switching



Design overview



What's done

- Mathematical implementation;
  - Including correction for:
    - Average p(t) value;
    - Application frequency;
- Event driven model on python;
- Event emulator and testing framework on S60;
- Profile data persistence on S60;

What's in progress

- Integration with EPM Framework;
- Implementation of 2rd order p(t) polynomial approximation;
- Extending model towards to use additional parameters;
- Some GUI implementation;

What's next

- To define initial values for plug-in;
- Concordance predictions with real battery life;
- Porting to MAEMO (the challenge);
- Testing in reality :)

Contact information

- Project page:
  - osll.spb.ru/projects/epm
- OSLL maillists
  - osll@osll.spb.ru
  - osll@googlegroups.com