A Framework for Region-based Instrumentation of Energy Consumption of Program Executions

Simon Ostermann, Thomas S. Eiter,
Vlad Nae, and Radu Prodan

Institute of Computer Science,
University of Innsbruck, Austria

simon@dps.uibk.ac.at

Outline

- Problem
- Solution
- Architecture
- Results
- Conclusion

Problem

- Energy efficiency is a key issue in computer science research and development
- Driving forces behind this are:
  - Growing energy consumption of computers worldwide,
  - Trend towards mobile devices
  - Green IT initiative

Problem (II)

- Techniques and technologies have been invented for energy:
  - Most of them focusing either on hardware or on software
  - Software application defines which hardware components are used -> energy of hardware
- Total energy consumption of a code execution on a specific machine depends on both the machine and on the code characteristics.

Problem (III)

- Energy measurements in the field of data centers are coarse grained
  - E.g. power-rail level, rack level,
- Difficult to collect
  - E.g. proprietary interfaces, sometimes not available to the users,
- Not associated to the codes being executed.
  - Or only available post mortem after execution

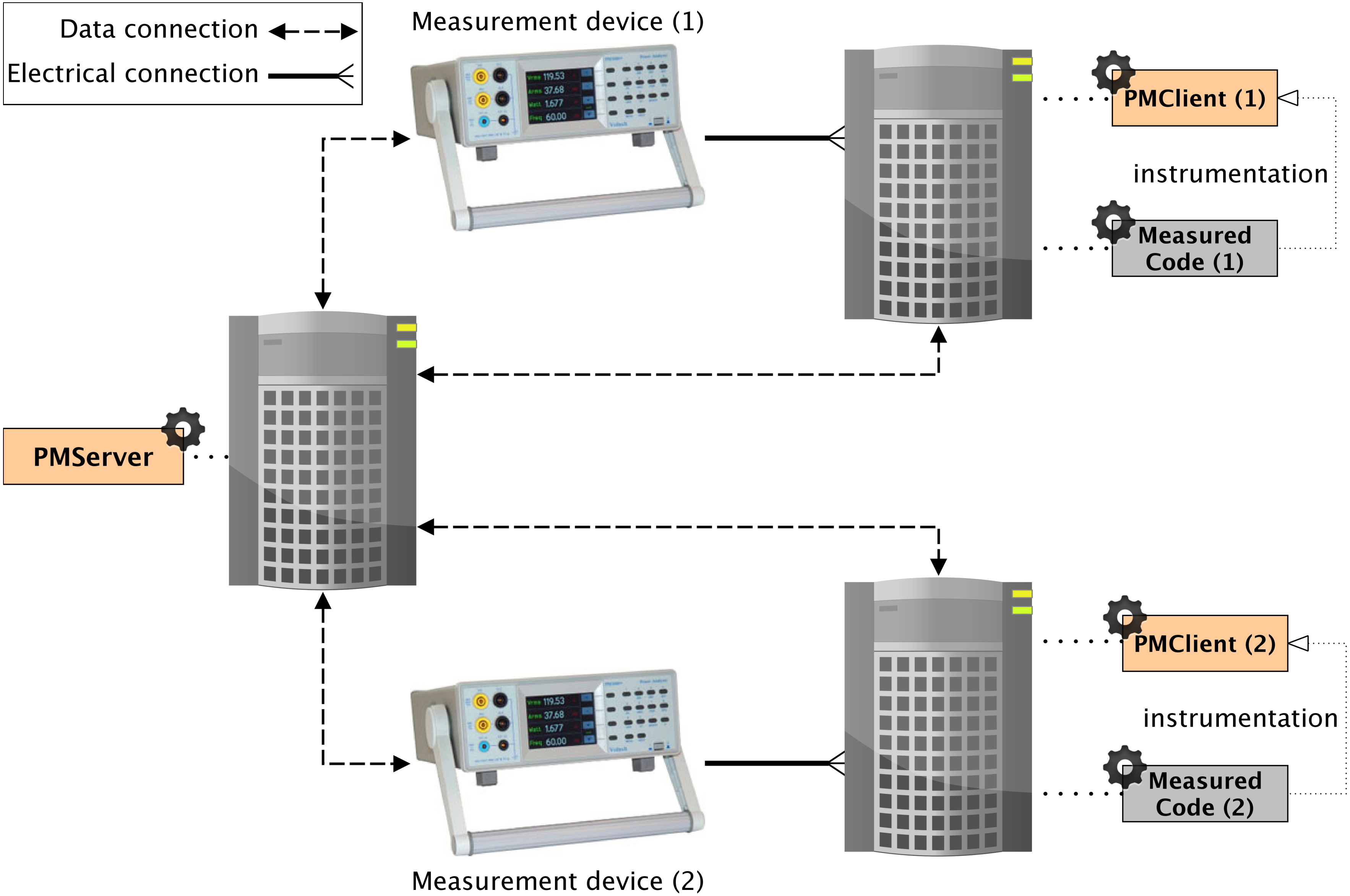
Solution

- Energy instrumentation and measurement framework that allows:
  - Implementation and easy deployment of self-tuning codes in data centers.
  - Both offline and online (i.e. during execution) analyses by providing detailed energy consumption data.
  - Software program executions with a region-level granularity of measurements.

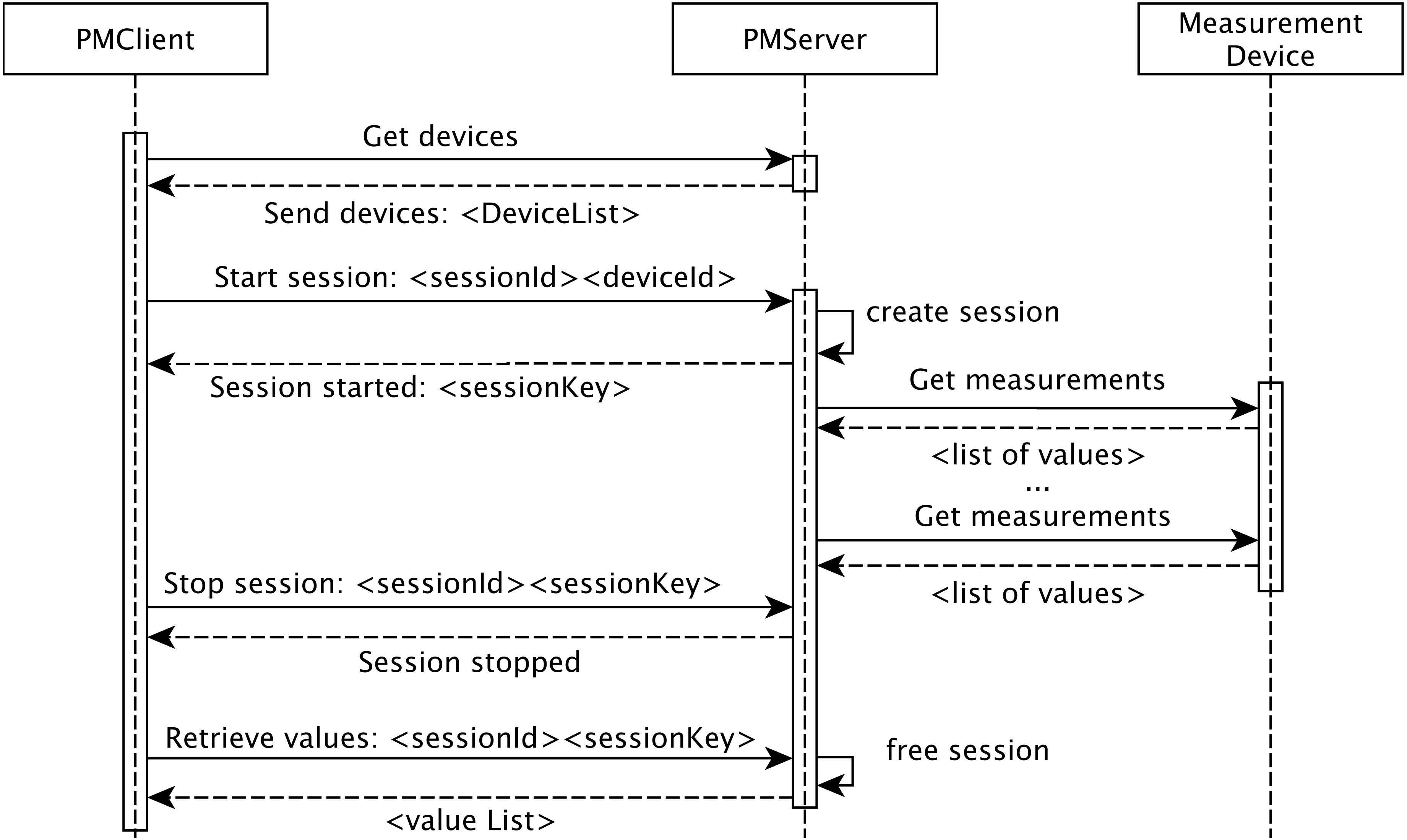
Architecture

- Client-server architecture
- Server running on different host to minimize measurement overheads
- Measurement devices communicate with server
- Clients can get measurements from server
  - Easy portable to all languages using the server-API

Architecture

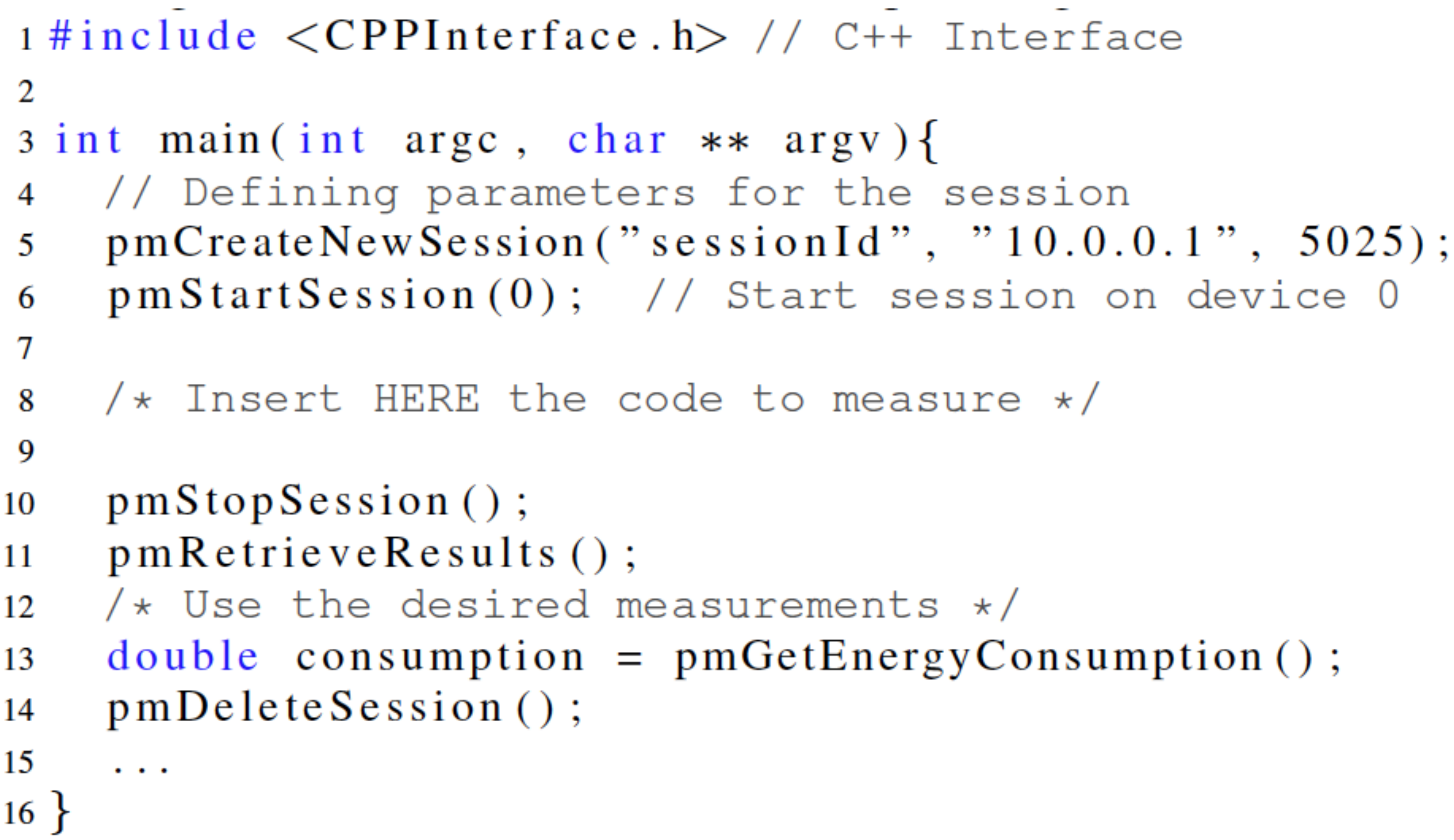


Communication



PMClient
PMServer
Measurement Device
Get devices
Send devices: <DeviceList>
Start session: <sessionId><deviceId>
create session
Session started: <sessionKey>
Get measurements
<list of values>
...
Get measurements
<list of values>
Stop session: <sessionId><sessionKey>
Session stopped
Retrieve values: <sessionId><sessionKey>
free session
<value List>

Code instrumentation

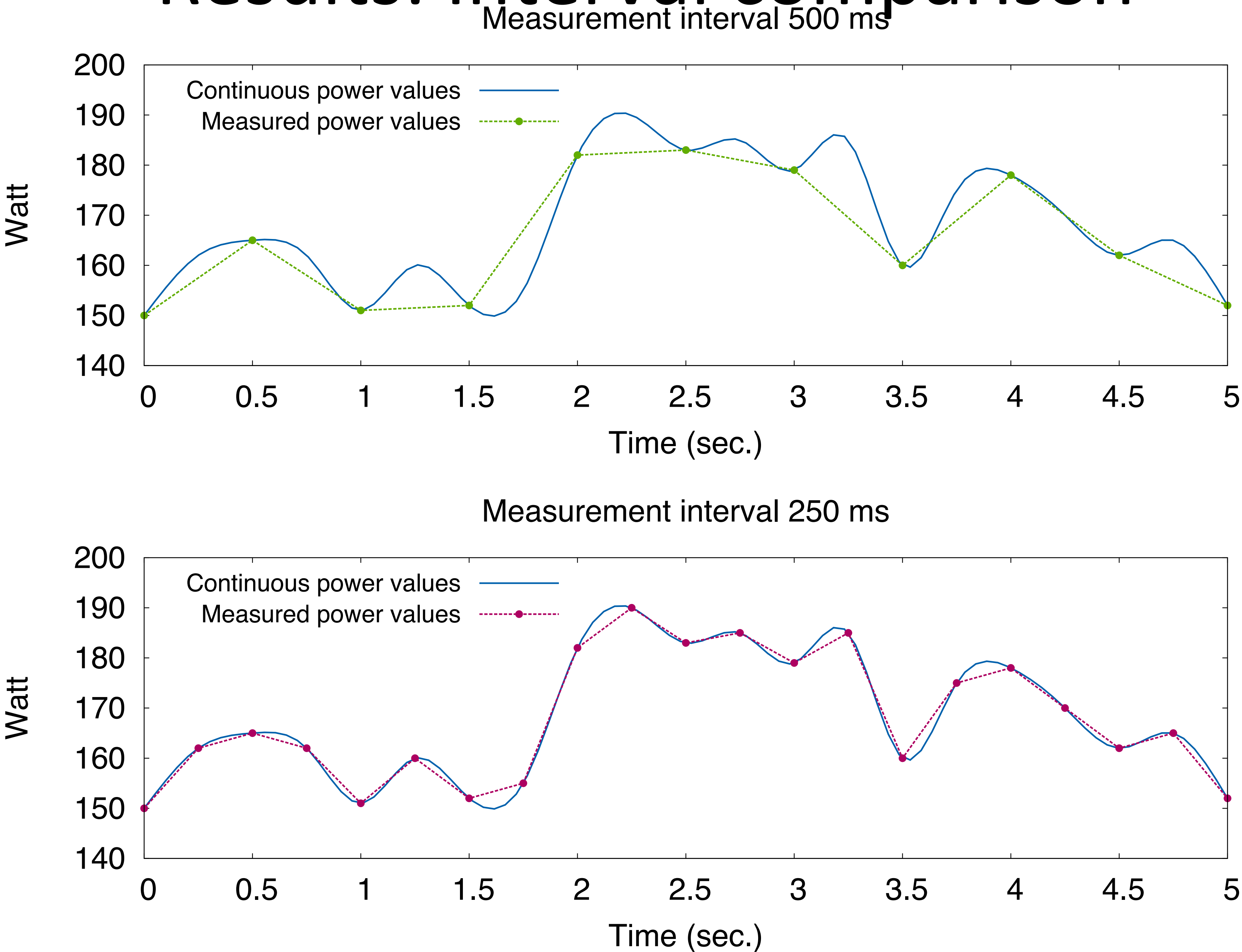


```cpp
#include <CPPInterface.h> // C++ Interface

int main(int argc, char ** argv){
  // Defining parameters for the session
  pmCreateNewSession(”sessionId”, ”10.0.0.1”, 5025);
  pmStartSession(0);  // Start session on device 0

  /* Insert HERE the code to measure */

  pmStopSession();
  pmRetrieveResults();
  /* Use the desired measurements */
  double consumption = pmGetEnergyConsumption();
  pmDeleteSession();
  ...
}
```

Accuracy

- Measurement interval
- Workload – power consumption delay
  - Measure the characteristic machine delay
- Measurement truncation
  - Pre and post measurements for each session
- Measurement aggregation functions
  - Different functions available to match different use cases: Skewing, Graph and Basic

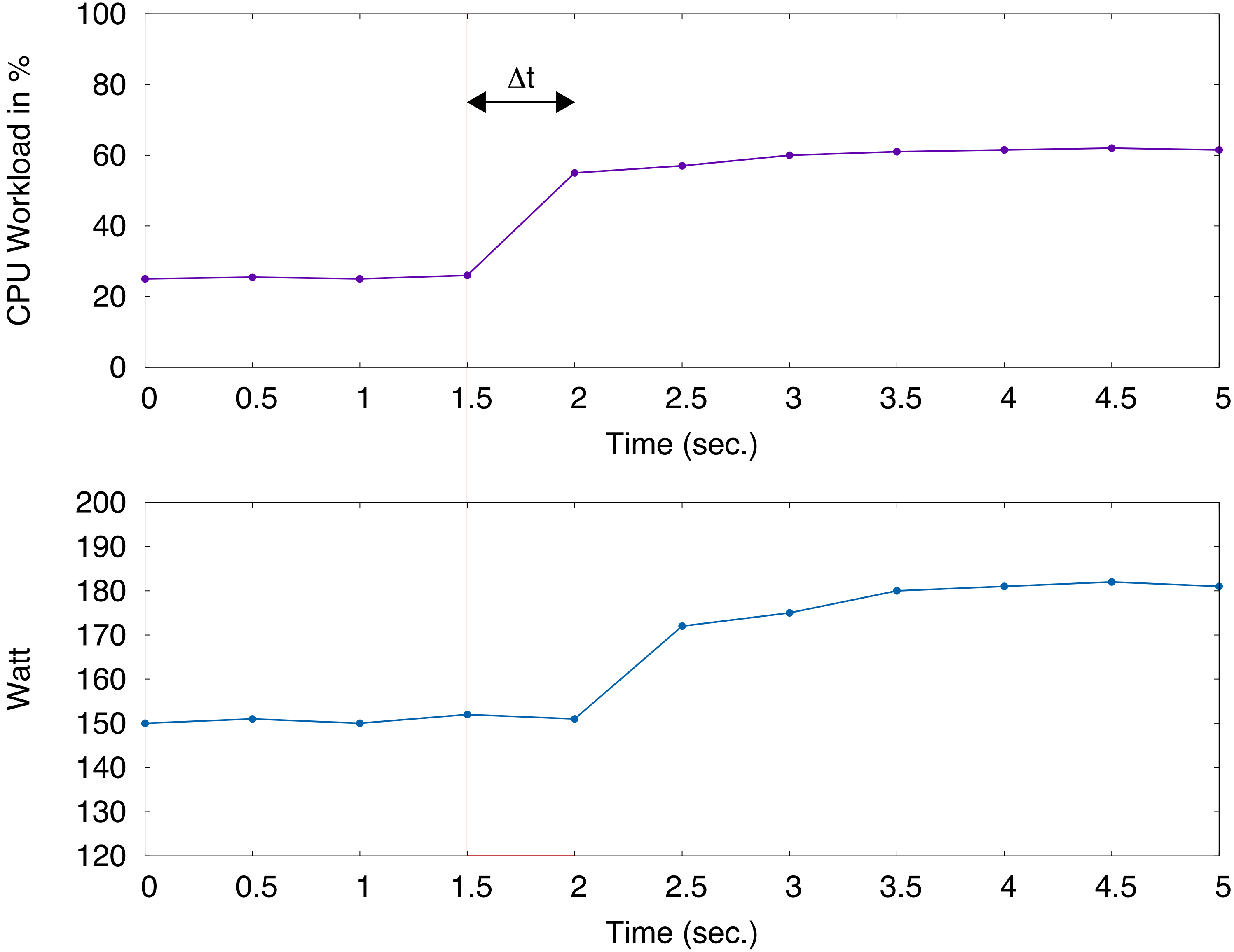
Results: Interval comparison



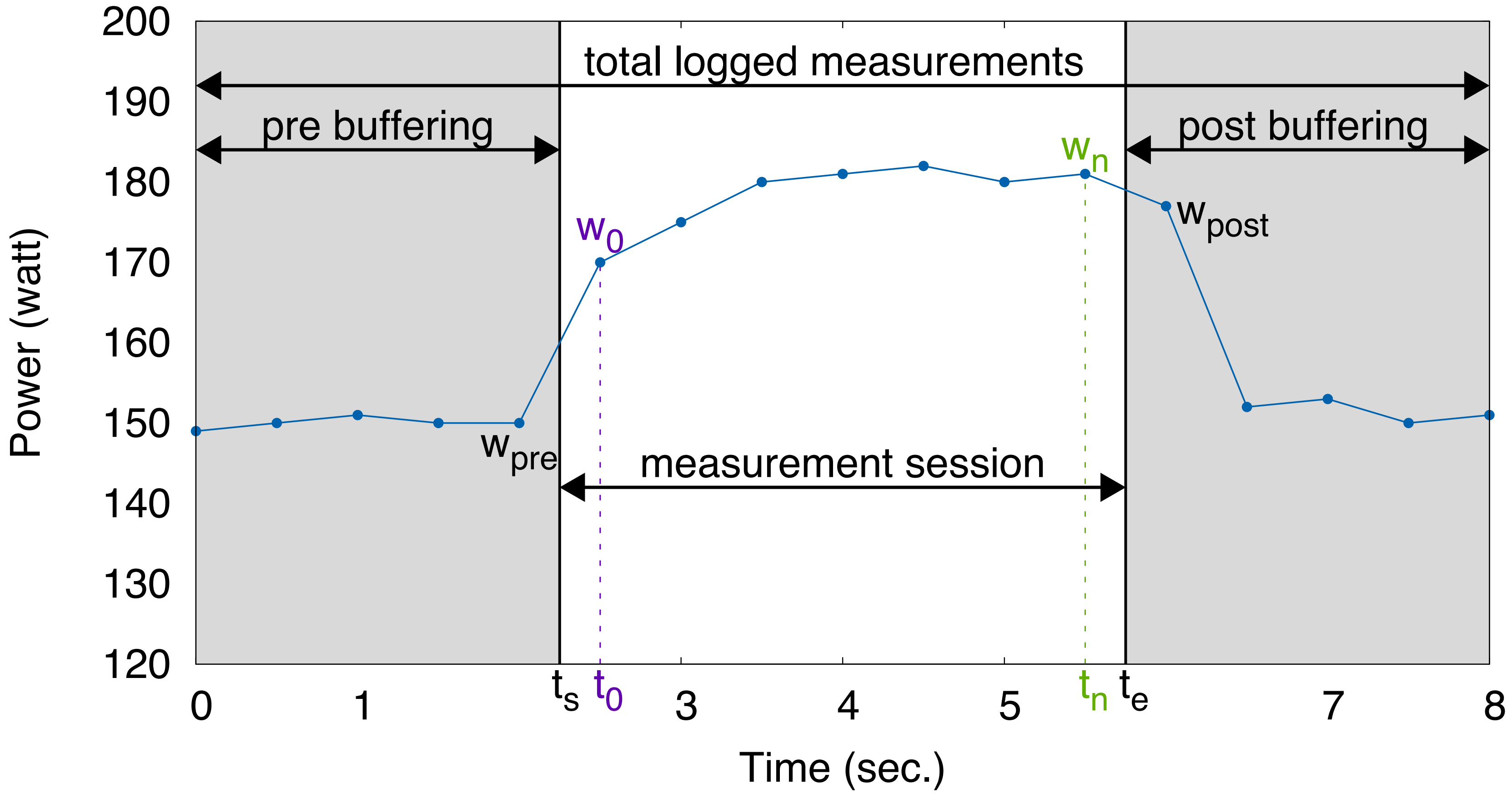
Measurement interval 500 ms

Measurement interval 250 ms

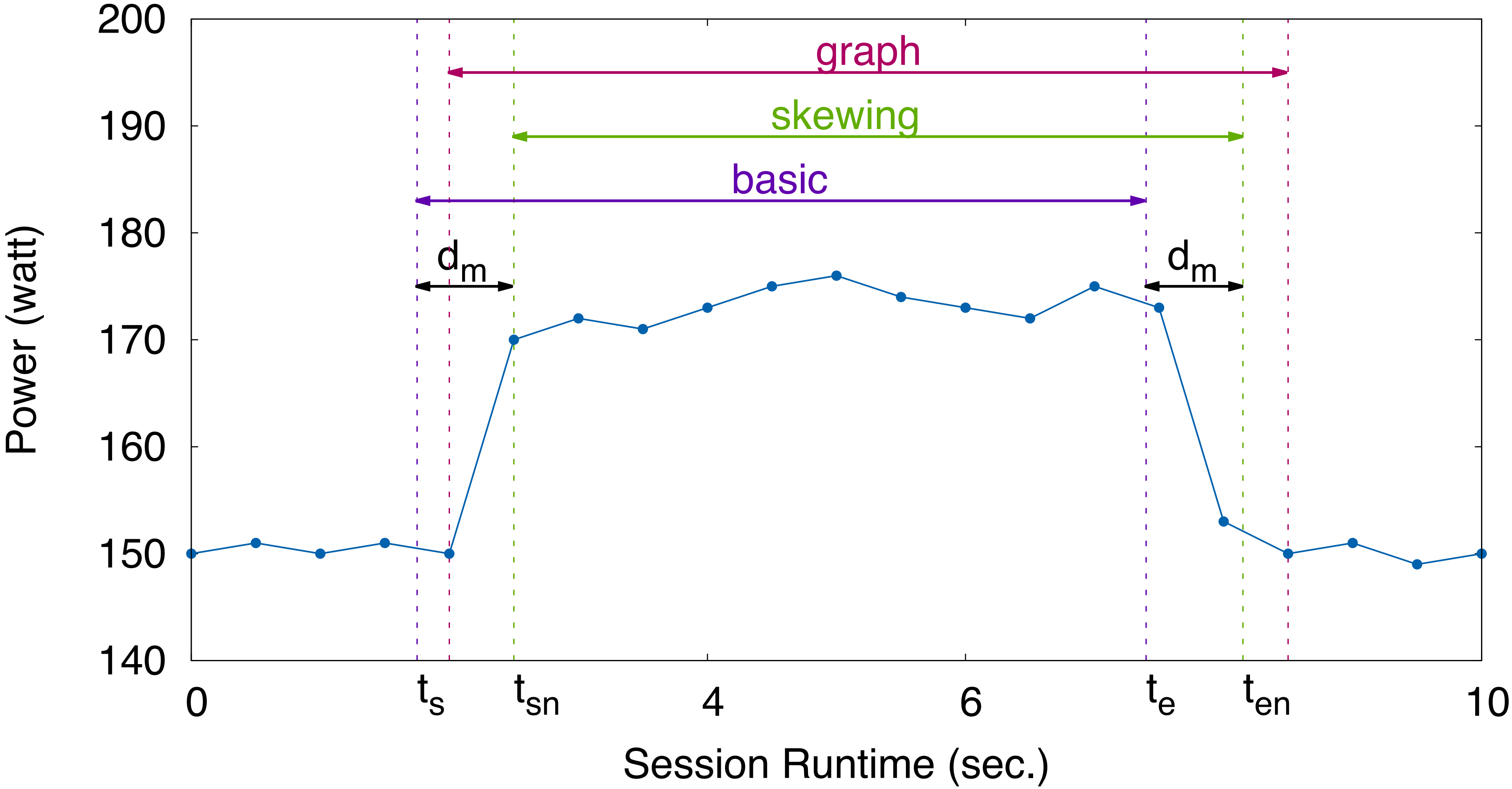
Results: Delay



Results: Buffering



Results: Aggregation functions



Conclusion

- Energy instrumentation and online measurement framework targeting the deployment of self-tuning codes
- Offline and online analyses by providing detailed energy consumption data for software program executions
- Region-level granularity
- Modular architecture shields the users from energy measurement hardware
- Allows the development of measurement and instrumentation code independent of the instrument's proprietary interface.
- Efficient method for increasing the accuracy of measurements for low sampling rate measurement devices (0.5 - 10 Hertz sampling frequency).
- Using Voltech PM1000+ devices we increased accuracy by 20% for measurements lasting 5 seconds and 9% for 10 seconds measurments.