

Demo: Mobile Application Resource Optimizer (ARO)

Feng Qian
University of Michigan

Z. Morley Mao
University of Michigan

Zhaoguang Wang
University of Michigan

Subhabrata Sen
AT&T Labs Research

Alexandre Gerber
AT&T Labs Research

Oliver Spatscheck
AT&T Labs Research

Categories and Subject Descriptors:

C.2.1 [Computer-Communication Networks]: Network Architecture and Design – Wireless Communication;
C.4 [Performance of Systems]: Measurement Techniques

General Terms: Design, Measurement, Performance

Despite the popularity of mobile applications, their performance and energy bottlenecks remain hidden due to a lack of visibility into the resource-constrained mobile execution environment with potentially complex interaction with the application behavior. We design and implement ARO [1], mobile Application Resource Optimizer, the first tool that efficiently and accurately exposes the cross-layer interaction to enable the discovery of inefficient resource usage.

System Overview. ARO consists of two components: the data collector and the analyzers. The data collector runs efficiently on a handset to capture information essential for understanding resource usage, user activity, and application performance. The collected traces are subsequently fed into the analyzers, which run on a PC, for offline analysis. We describe the workflow of ARO as outlined in Figure 1.

1. The ARO user invokes on her handset the data collector, which collects relevant data, *i.e.*, all packets in both directions and user input. ARO also identifies the packet-to-application correspondence. This is used to distinguish the *target application*, *i.e.*, the application to be profiled, from other applications simultaneously accessing the network.

2. The ARO user launches the target application and uses it as an end user. To obtain a representative understanding of the application studied, ARO can be used across multiple runs or by multiple users to obtain a comprehensive exploration of different usage scenarios of the target application.

3. The ARO user loads the ARO analysis component with the collected traces. ARO then configures the RRC analyzer with handset and carrier specific parameters, which influence the model used for analysis. The TCP, HTTP, and burst analyzers are generally applicable.

4. ARO then performs a series of analyses across several layers. In particular, the RRC (Radio Resource Control) state machine analysis accurately infers the radio states from packet traces so that ARO has a complete view of radio resource and radio energy utilization during the entire data collection period. ARO also performs transport protocol and application protocol analysis to associate each packet with its transport-layer functionality and its application-layer semantics. ARO next performs burst analysis, which utilizes aforementioned cross-layer analysis results, to understand the triggering factor of each short traffic burst, which is the key reason of low efficiency of resource utilization.

5. ARO profiles the application by computing for each burst (with its inferred triggering factor) its radio resource and radio energy consumption so that an ARO user can accurately identify and quantify the resource bottleneck for the application of interest.

Implementation and Demo Setup. We have fully implemented ARO for Android 2.2 (8.5K LoC). We apply ARO to more than 20 real Android applications each with at least 250K downloads from the Android market as of Mar 2011. ARO reveals that many of these very popular applications (*e.g.*, Fox News, Pandora, and eBay) have significant resource utilization inefficiencies that are previously unknown. In the demo, we will show step-by-step how to interactively use ARO to identify resource inefficiencies of smartphone applications by analyzing real traces (Figure 2). No special equipment is required other than a demo booth.

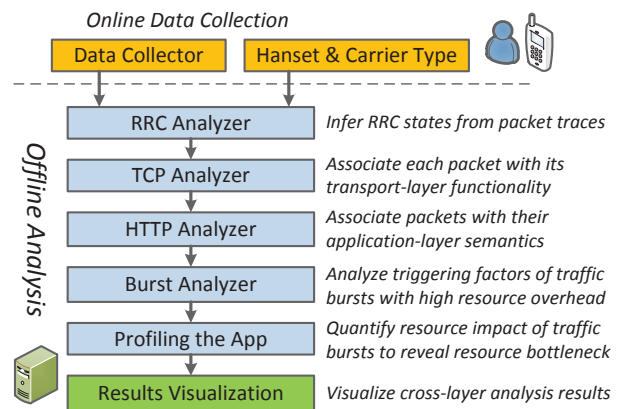


Figure 1: The ARO System

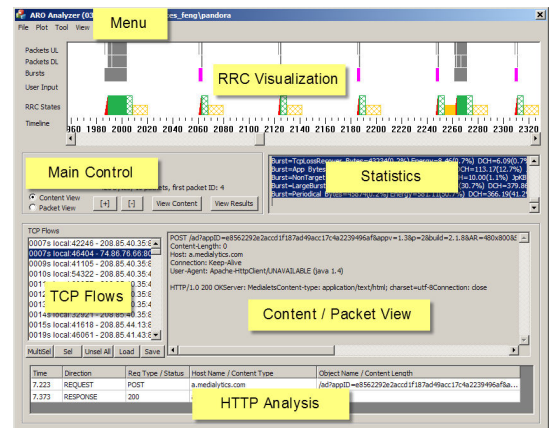


Figure 2: The ARO analyzer user interface

1. REFERENCES

- [1] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Profiling Resource Usage for Mobile Applications: a Cross-layer Approach. In *Mobisys*, 2011.