

SOFTWARE-BASED SYSTEM USABILITY EVALUATION FOR PROJECT54-EQUIPPED POLICE CRUISERS

Edward Bourbeau¹⁾

Andrew L. Kun¹⁾

Abstract

This paper examines the use of log files and post-processing techniques to quantify how police officers utilize the Project54 system. Project54 is a software-based package designed to integrate off-the-shelf electronic components commonly found within police cruisers. This system allows police officers to interact with the in-vehicle devices using either speech commands or a touch-screen graphical user interface, while still being able to utilize each device's original hardware controls. The collected in-car data will be a valuable tool for guiding future Project54 development, including context-cue recognition from the logged human-computer interaction information. Preliminary results are presented from a testing environment.

1. Introduction

For the past several years, one of the main goals in pervasive computing has been to provide context awareness within ubiquitous computing applications. To this end researchers have packed sensors into their testing devices in an attempt to account for as many situations involving human-computer interactions (HCI) as possible. The information supplied by these sensors would allow conclusions to be drawn regarding the manner in which the devices were used.

While using sensors may present useful results that spur on further developments there are environments, like the one for which the Project54 system is designed, where adding extra equipment may not be permissible. One of the attractive Project54 features is that the in-vehicle device integration was done via software, using only minimal hardware for connectivity purposes [4]. As a result, police officers are free to interact with their environment in whatever way is most comfortable and natural for them (see Fig. 1). A software solution is the most viable approach since additional sensor equipment would both take up space (which is at a high premium inside police cruisers) and potentially cause the officers to use the system in an unnatural manner (due to constantly being reminded that their actions are monitored). The two main design steps were to first, develop a method of logging information regarding Project54 HCI events and second, to analyze those logs to form comprehensive conclusions. These results could later be used to direct any necessary system improvements as well as form the basis for context-aware application development.

¹ University of New Hampshire, Durham, NH 03824 USA

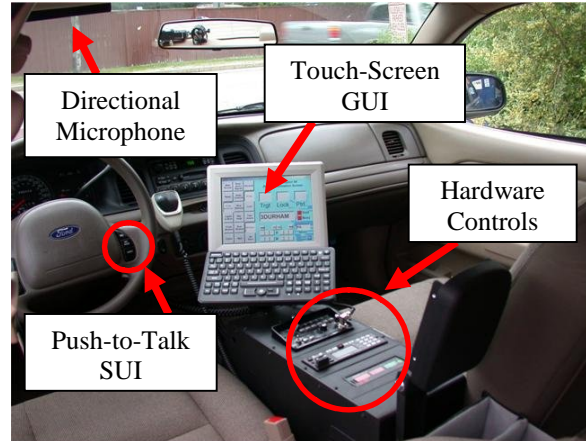


Fig. 1. The Project54 system installation lets police officers control their in-vehicle devices via Speech User Interface (SUI), Graphical User Interface (GUI), or the original hardware controls.

2. Data Collection

The data collection process was built up from the Project54 application messaging system. This software, already featured in Project54-equipped police cruisers, uses a text stream to provide inter-application commands and status updates [5]. The text messages provided direct evidence of software-based interactions, inferable evidence of hardware control usage, and were readily available for the taking. However, retrieving the relevant information, while ignoring “background” messages, was more complicated than merely monitoring and capturing communication traffic. Hilbert and Redmiles had addressed similar challenges behind preserving data resolution during high-volume data collection [3]. The data logging software also had to track graphical user interface (GUI) events in a more direct fashion because the text messages did not contain information such as the active window or button labels. Another feature of the data logging, particularly useful for determining context, was the inclusion of GPS velocity tags for each of the log entries. With the vehicle speed known for any given user interaction, post-processing would then be able to differentiate between which tasks police officers tend to perform while parked, patrolling, and while driving at high speeds.

3. Log File Visualization

Guzdial, et. al. indicated that, even streamlining the logging process would not make the large amount of collected data manageable enough for a person to manually analyze the information [1]. Therefore, this research employed several post-processing methods that would result in concise findings. The challenge was to develop an automated process capable of generating multiple data abstractions from the original in-vehicle log files. Specifically, the following were some of the questions that needed to be addressed using post-processed log files: How often do police officers use Project54 vs. the classical hardware controls? Is the use of a particular interface task-based? What, if any, discernible patterns emerge from the way police officers use Project54? And finally, what role does driving vs. parking play in how police officers interact with their in-car devices? Through testing alone, there was enough information available to judge which methods of visualizing the in-car data would be able to generate clear and meaningful results. For example, a count of the number of times police officers issued a speech command or pressed a button on the computer monitor’s touch screen, versus the number of times hardware controls were used, could be a simple and accurate way for detecting whether the officers prefer to use Project54 or the standard hardware controls. Another, more

interesting method was to use a Project54 window screenshot and overlay a color gradient onto that window's buttons in order to determine each button's usage frequency. This method stemmed from research which stated that displaying information using the system from which the results were derived is a very effective way to demonstrate the importance any results may possess [2]. These results could have implications for future versions of Project54 software if, for example, the results showed widespread trends towards using buttons based on their location within the window.

4. Testing

Once the data logging and analysis tools were developed, testing was performed to verify that Project54 HCI event information could be gathered and analyzed. The logging software was loaded into a laboratory car mock-up, a driving simulator, and two test vehicles. The tests were conducted in three phases – simulated HCI event recording under simulated driving conditions using the lab-car, actual HCI event recording under simulated driving conditions using the driving simulator, and actual HCI event recording under actual driving conditions using test cars.

The laboratory car test involved executing Project54 speech and GUI commands every four seconds for two weeks with the data logging software enabled. Results concluded that the logging software was able to run at all times without causing delays in command execution, generating any system crashes, or other negative side-effects. The test-vehicle and driving simulator testing both consisted of a human driving while using the speech user interface (SUI), GUI, or hardware controls to operate the in-vehicle equipment. These tests were conducted for approximately five hours apiece, in total, and confirmed the original results from the laboratory car tests.

The HCI event test logs were retrieved and used to validate the automated post-processing application. The image shown in Figure 2 represents the results of one possible button-press analysis of the driving simulator test data, using the screenshot and color-coded button usage visualization technique mentioned earlier.

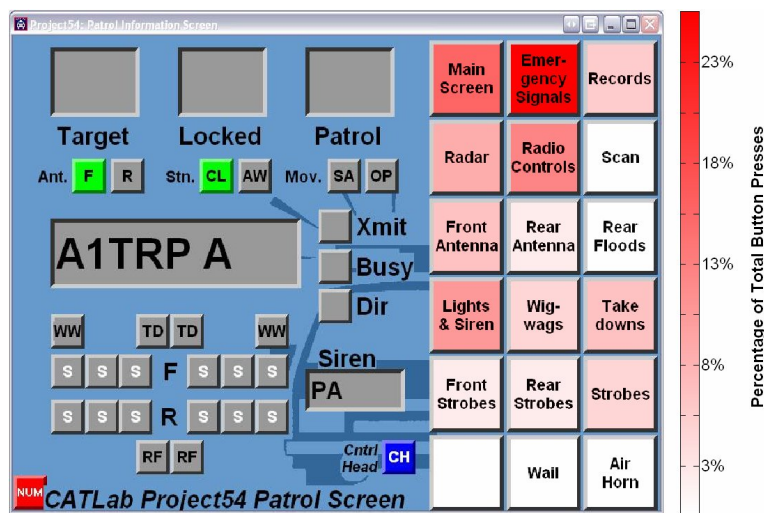


Fig. 2. The three columns of buttons are color-coded based on the amount of times each button was pressed. The color bar illustrates the percentage of the total presses that each button in this window received.

Conclusion and Future Work

Software solutions for gathering and post-processing HCI event information from Project54-equipped police cruisers were developed and tested in the laboratory. The data-logging software was capable of collecting three different forms of interactions (i.e. GUI, SUI, and hardware controls) and tracking the vehicle's speed during each recorded event. The process was made to be invisible to the police officers in order to prevent them from using Project54 in an irregular fashion. The post processing application was designed to automate log-file sorting and data filtering, so results could be clearly understood while preserving the content of the collected information.

The next step with this software is to conduct field testing. Laboratory testing was able to generate an adequate method for presenting results but could not possibly produce applicable, "real-world" results. For example, current analysis schemes have not been able to properly harness the logged vehicle speed data because the testing process used to gather the information was inherently without any useful context. Field testing will provide an opportunity to expand the involvement of recorded vehicle-speed information in determining any context that may be involved in user interactions. Information collected from commissioned police cruisers should provide insight into how the next generation of Project54 software could be tailored to suit the officers' usage tendencies.

5. Acknowledgements

This work was supported by the U.S. Department of Justice under grants 2001LTBXK010 and 2005CKWX0426.

6. References

- [1] GUZDIAL, M., SANTOS, P., BADRE, A., HUDSON, S., GRAY, M., Analyzing and Visualizing Log Files: A Computational Science of Usability, Presented at HCI Consortium Workshop, 1994
- [2] HILBERT, D. M., REDMILES, D. F., Extracting Usability Information from User Interface Events, ACM Computing Surveys, Volume 32, Number 4, pp. 384 – 421, December 2000
- [3] HILBERT, D. M., REDMILES, D. F., Large-Scale Collection of Usage Data to Inform Design, Proceedings of the 8th IFIP TC.13 Conference on Human-Computer Interaction (INTERACT '01), 2001
- [4] KUN, A. L., MILLER III, W. T., LENHARTH, W. H., Computers in Police Cruisers, IEEE Pervasive Computing, Volume 3, Issue 4, pp. 34 – 41, October – December 2004
- [5] MILLER III, W. T., Project54 Application Manager Messaging, Technical Report ECE.P54.2003.2, University of New Hampshire, March 2003