# Poster: *Observe. Patternize. Mimic.* Leveraging Patterns in Mobile-User Behavior for Enterprise Applications

Uma Parthavi Moravapalle[a], Shruti Sanadhya[b], Cheng-Lin Tsao[c] and Raghupathy Sivakumar[a]
[a]Georgia Tech.; [b]Hewlett Packard Labs; [c]Anthem Innovation Studio

## 1. OVERVIEW

Application Mobilization, or the ability of an enterprise employee to rely on mobile devices such as smartphones and tablets, to continue to perform business workflows even when mobile, is seen as a game changer to improve productivity[1]. However, the practical adoption of enterprise mobility is very much in its infancy, and seemingly has barriers. We posit that these barriers include *heavy user-burden in accomplishing tasks* (e.g. number of actions required to execute a workflow), *high cost of mobile access* (e.g. latency for content fetching), and *irrelevance of available mobile functions* (e.g. mobile app defeaturization done inappropriately).The novelty of our research is in a **unified observe-patternize-mimic paradigm** we explore to address these barriers, based on a simple question: *could patterns in user-behavior be learned, and leveraged for reducing user-burden?* If patterns are discovered, then we show that intelligent mimicking of these patterns at appropriate junctures can considerably relieve the mobile-user burden. We motivate this paradigm through three application scenarios representing $read$, $write$, and $act$ usage modalities.

## 2. OBSERVE. PATTERNIZE. MIMIC.

***Precog (read):*** An enterprise worker might consume web content from different locations (e.g. work, home, customer office etc.), sometimes over high cost/latency networks (e.g. 3G/4G cellular). By observing the user's browsing behavior to extract content access patterns, and using these patterns to intelligently prefetch web content that the user is likely to need in the future over cheaper networks (e.g. WiFi), we can achieve cost/latency benefits. Existing solutions are either name-based (prefetch most used URLs and hence irrelevant for dynamic web) or are network-unaware (do not consider differing network costs i.e. WiFi is cheaper than cellular). Thus, we propose *Precog*, a solution that leverages consistency in page layouts to identify repeating user actions (e.g. clicking top stories) on cellular networks and preplays them on WiFi before the cellular access. We find that *Precog* can achieve up to 37% latency reduction through mobile browser action traces collected from 8 users (see fig. 1a).

This work was funded in part by the National Science Foundation under grants CNS-1513884 and CNS-1319455, and the Wayne J. Holman Endowed Chair.

[1]http://www.techrepublic.com/article/infographic-byod-is-popular-but-not-widely-supported-by-it/

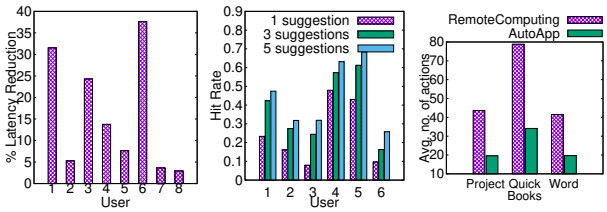


*MobiSys'17 June 19-23, 2017, Niagara Falls, NY, USA*



ACM ISBN 978-1-4503-4928-4/17/06.

DOI: http://dx.doi.org/10.1145/3081333.3089311




**(a)** Read **(b)** Write **(c)** Act

**Figure 1:** Benefits of intelligent pattern mimic

*Precog observes and extracts patterns in user's actions on a webpage while on cellular and preplays these actions over WiFi for latency and cost benefits.*

***Dejavu (write):*** An enterprise worker spends an inordinate amount of time on responding to email queries from co-workers, many of which can be similar or identical in nature. Our proposed research *Dejavu* observes the user's mailbox and automatically suggests email responses for emails using the content already present on the mailbox. *Dejavu* extracts important keywords from an inbox email, matches these keywords to emails already in the mailbox, and suggests emails that are closest to the current email (min. 60% similarity) as responses. Using the mailbox data from 6 users (see fig. 1b), we found that this simple matching approach in *Dejavu* can find suggestions that have >60% match with actual response (hitrate) for 33% of all emails (1 suggestion) on average.

*Dejavu observes emails in a user's mailbox to suggest responses that closely match keywords in an inbox email and reduces user burden of typing the response.*

***AutoApp (act):*** One promising strategy for application mobilization is defeaturization - where-in the number of features exposed on the mobile app is a tiny fraction of what the desktop version supports. This is accomplished today through manual and static design and remains the same for all users. Our proposed solution, *AutoApp*, is based on application refactoring on remote computing, with a server running the application and a client with a remote view. Unlike remote computing, *AutoApp* observes desktop usage patterns, defeaturizes the application to contain only commonly used features, condenses some patterns into macros, and transforms the UI elements of the original application view to smartphone native UI elements. A preliminary prototype for *AutoApp* for Windows applications (Quickbooks, Project and Word) with a manual transformation of application views, showed up to 66% reduction in the number of actions to perform a task compared to remote computing (fig. 1c).

*AutoApp observes desktop application usage to customize defeaturization for each individual user, thereby reducing the task burden of performing workflows.*