Attelia: Reducing User's Cognitive Load due to Interruptive Notifications on Smart Phones

Tadashi Okoshi*, Julian Ramos‡, Hiroki Nozaki†, Jin Nakazawa†, Anind K. Dey‡ and Hideyuki Tokuda†
*Graduate School of Media and Governance, Keio University
slash@ht.sfc.keio.ac.jp
†Faculty of Environment and Information Studies, Keio University
chacha@ht.sfc.keio.ac.jp, jin@ht.sfc.keio.ac.jp, hxt@ht.sfc.keio.ac.jp
‡Human Computer Interaction Institute, Carnegie Mellon University
ingenia@andrew.cmu.edu, anind@cs.cmu.edu

***Abstract*—In today's ubiquitous computing environment where the number of devices, applications and web services are ever increasing, human attention is the new bottleneck in computing. To minimize user cognitive load, we propose Attelia, a novel middleware that identifies breakpoints in user interaction and delivers notifications at these moments. Attelia works in real-time and uses only the mobile devices that users naturally use and wear, without any modifications to applications, and without any dedicated psycho-physiological sensors. Our evaluation proved the effectiveness of Attelia. A controlled user study showed that notifications at detected breakpoint timing resulted in 46% lower cognitive load compared to randomly-timed notifications. Furthermore, our "in-the-wild" user study with 30 participants for 16 days further validated Attelia's value, with a 33% decrease in cognitive load compared to randomly-timed notifications.**

I. Introduction

The amount of information available for consumption has grown by orders of magnitude while the amount of attention users have has remained constant. This has, in part, driven users to multi-task more and use notifications on their computing devices, often resulting in greater numbers of interruptions. The number of networked computing devices belonging to users, along with those embedded in the environment, such as home, office, or urban space, have also been increasing quickly. Users own, carry and interact with (even simultaneously) [1], an increasing number of mobile networked devices [2]. On each device, the number of applications, services, and communication channels is increasing as well, being driven by both technological progress and market trends, such as maturing Web middleware and flexible cloud platform services for rapid service deployment, or global application markets such as the "AppStore". Moreover, the advent of social networking services in addition to conventional communication channels such as email and SMS increases the number of people that users communicate with daily.

Given this background of information overload, the limited resource of human attention is the new bottleneck [3] in computing. In this paper, we focus on interruption overload, a form of distraction caused by the excessive number and inappropriate delivery of notifications from computing systems. Typical notification systems deliver notifications immediately after they are available, and this has been shown to negatively affect users' work productivity [4], [5], [6], [7]. One possible solution is to defer notifications until the user's natural breakpoint [8], the boundary between two adjacent units of users activity, which can lower the impact on users cognitive load caused by the interruption.

In this paper, we particularly focus on user's "mobile experience" on the phone while they are actively using their own devices, and demonstrate our ability to detect breakpoints, towards the realization of user-attention-aware adaptive notifications. Our system, Attelia, (1) works on smartphone devices, (2) is applicable to situations of user mobility and use of a wide variety of applications, (3) performs real-time detection of breakpoints to support real-time adaptation, and (4) does not require the use of dedicated external psycho-physiological sensors.

A controlled user study with 37 participants showed that providing notifications at detected breakpoints resulted in 46% lower cognitive load, compared with conventional notifications presented to users at "random" times, for users who showed higher sensitivity to notification timings. Furthermore, our "in-the-wild" user study for 16 days with 30 participants showed similar results. Providing notifications at detected breakpoints resulted in 33% lower cognitive load for the users who showed higher sensitivity in their subjective evaluations. Also, users' response time to the notification was faster by 13% than using conventional (or random) notification timings.

The contribution of this paper is two-fold. First, we present the design and implementation of our novel middleware for real-time breakpoint detection on smartphones that does not require the use of dedicated external psycho-physiological sensors. Second, we present the results from both a controlled and an "in-the-wild" field user study that resulted in significantly lower cognitive load when the breakpoint detection system was used to defer notifications.

In the remainder of this paper, we describe the interruption overload problem caused by notifications from computing systems in Section II. Next we define the requirements for adaptive notification scheduling on smartphones after we introduce recent trends in notifications in Section III. We then present our design approach for Attelia in Section IV, and describe the Attelia system architecture in Section V. Section VI describes our controlled user study with 37 participants and Section VII reports on a follow-up field study for 16 days with 30 users. In Section VIII we discuss further research opportunities based on the findings from our user studies. Section IX describes related work, and we conclude this paper in Section X.

II. Interruption Overload

Interruption overload caused by large numbers of ill-timed notifications is one piece of the larger problem of information overload, and is increasing in frequency. As such, more research has focused on the topics of interruptions and multitasking [9]. The main source of interruption overload is "notifications" from computing systems. Notifications were originally designed to push information to users in a more speedy and timely manner, rather than requiring users to pull or manually look for new information.

Despite their obvious benefits, notifications have been shown to negatively affect users' work. Several researchers have found that they lead to a reduction in work productivity, including the resumption time from the interruption back to the primary task and the quality and amount of time available for decision making [4], [5], [6], [7], [10], [11]. Furthermore, other researchers have found increased negative affects or emotional states, social attribution[4] and psycho-physiological states [10] as a result of these interruptive notifications. Although notifications can be configured by users, and even be disabled, simply disabling notifications negates their benefits and cannot satisfy users' needs for the timely provision of information. Previous research has shown that users prefer to keep using notification systems for information delivery even given the interruption costs, rather than turning them off and checking for new information manually [12].

A. *Existing Research on Mitigating the Cost of Notifications*

There are two main approaches for addressing the problem of interruptive overload: (a) scheduling (deferring) notifications, and (b) mitigation of notifications. In the first approach, deferring notifications, several research projects have used "breakpoints" to target the timing of deferred notifications to users. A breakpoint [8] refers to a concept in psychology in which a human's perceptual system segments activities into a hierarchical structure of discrete sub-actions. The boundary between two adjacent action units is called a breakpoint. There are at least three granularities of breakpoints: Fine, Medium, and Coarse, that can be reliably detected by users performing interactive computing tasks [13]. Related research has shown that deferring notifications until users sensed breakpoints reduces interruption cost in terms of task resumption lag and subjective frustration [4], [14], [15]. Another approach, mitigation, tries to reduce a user's interruptive cognitive load by changing the modality used to deliver notifications, such as "silent" mode, "vibration" mode or only flashing an LED, while leaving the timing of the notifications unchanged. This approach serves to change the saliency of the interruption.

Although the two approaches described above are not mutually exclusive and can be complementary with each other, in this paper, we particularly focus on notification deferral. Given the growing number of notifications that users deal with, changing the timing of notifications rather than their saliency would seem to have greater potential impact on users' interruptive overload. To the best of our knowledge, existing research exploring the deferring strategy has focused on desktop computing, mainly with a single device, with evaluation in a controlled environment. We see an important research opportunity in realizing and evaluating a deferring strategy (1) in a mobile environment, (2) in real-time, (3) with a variety of applications, and (4) using"in-the-wild" users and data.

III. Adaptive Notification Scheduling on Smart Phones

To help scope our research contributions, we define the requirements for adaptive notification scheduling on smartphones after we introduce recent trends in notifications.

A. *Recent Trends of Notifications*

Reflecting the recent trends in ubiquitous computing described in Section I, we point out and focus on the following distinctive characteristics of notifications in such environments.

- **Increasing diversity in types and sources of notifications:** Based on an increasing number of applications and services, communication channels and connections between users, there have been an increasing diversity of types and sources of notifications, including updates from social networks, signals from sensors, and queries from participatory sensing systems [16],
- **Multiple mobile devices as targets:** Users are carrying multiple mobile (and wearable) devices, including smartphones, tablets, smart glasses and smart watches [1], [2], all of which can be targets of notifications.
- **Wider range of urgency level:** While most notifications are informative in nature, some require almost instant reaction: e.g., Early Earthquake Warning (EEW) [17] notifications for which users need to physically react within a few seconds.
- **Increasing length of interruptive periods:** Recent lifestyles include always having access to one's mobile devices, making interruption overload an issue all day long, even while a user is sleeping.

B. *Principles for Attention Status Sensing*

To defer notifications to user's breakpoints, we must be able to sense their attention state. Based on the previous literature, we denote the following as principles in attention status sensing.

- **Feasibility for mobile devices:** Users carry and use mobile devices, such as smart phones or tablets, for everyday computing and communication. Thus a breakpoint detection system needs to work on a mobile platform, in terms of energy-efficiency, available sensors, etc.
- **Real-time sensing:** To support notification adaptation and deferral on the fly, the sensing needs to be performed in real-time.
- **Applicability to diverse types of notification sources:** The breakpoint detection system needs to work for diverse types of notification sources.

- **All-day-long use:** Breakpoint sensing needs to be performed all day long, or at least as long as the user's notification system is available.

IV. Design of Attelia

In this section, we present our design of Attelia, based on the requirements we described in the previous section. Attelia detects appropriate timings for delivering notifications to users, with three distinctive features. First, it detects those timings on smartphones, without the use of an external server or any psycho-physiological sensors. Second, Attelia detects breakpoints in real-time (not post-hoc) so that it can be used to adapt notification timings at run-time. Finally, the breakpoint detection can be applied to a wide range of applications installed on users smartphones.

The following subsections describe our approach for performing breakpoint detection that satisfy these three features, including: (1) using breakpoints to temporally target interruptions, (2) using application usage as a sensor, and (3) using machine learning to perform real-time breakpoint detection.

Since our research focus is on the user's "mobile experience" during his/her active manipulation of devices, Attelia scopes breakpoint detection during their active engagement with mobile devices, and does not consider moments when users are not interacting with them.

A. *"breakpoint" as a Temporal Target for Interruption*

Related work in real-time sensing of available user attention or cognitive load shows that at least two psycho-physiological sensors are needed even in non-mobile situations [18]. Given the burden of wearing a psycho-physiological device constantly, our approach only uses the users' mobile devices, and attempts to sense more coarse-grained, but easier to sense signals, from which appropriate timings for notifications can be inferred.

B. *Application Usage as a Sensor*

With our scoping to active use of mobile devices, we focus on a user's application usage and use that information to detect a user's breakpoints. We focus on application usage and not physical sensors, despite their wide proliferation on mobile devices for two reasons: simplicity of implementation and reducing the reliance on a sensor that may not exist on all target mobile devices (or may be mounted in different locations).

Table I shows some possible knowledge sources for identifying breakpoint and and Table II shows, for each source type, how it can be acquired. The application-related knowledge and information can include both relatively static knowledge that is specific to each application, such as when users transition between multiple "stages" in game applications, and that are designed and implemented by the application developers in the development phase; and relatively dynamic information, such as run-time status and events that result from the running applications. Using knowledge from the internals of any specific application is not feasible given the huge number of applications available and the fact that application developers would need to expose internal information at development time. Instead, we collect run-time status events from the

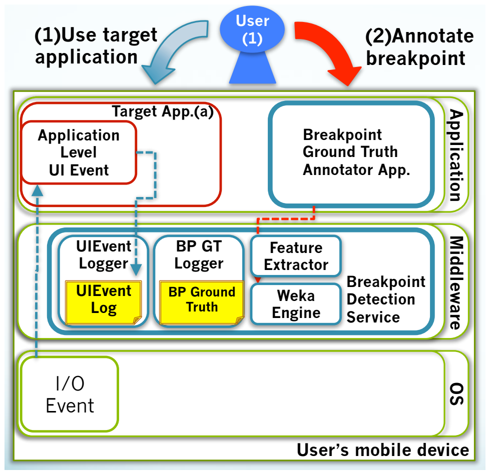

Fig. 1. System Architecture of Attelia on Android Platform

operating system and executing applications, and use them to identify relationships to ground truth values of interruptive overload provided by users, during a training phase.

C. *Real-Time Detection with Machine Learning techniques*

Similar to previous work on activity recognition, our approach also uses machine learning-based classification to understand these relationships. For each time frame T_f, a feature vector V is extracted from the sensed data, and a trained classifier identifies the time frame as a user breakpoint or not.

V. Attelia System Architecture

Figure 1 shows the system structure of Attelia implemented on the Android 4 platform. Attelia consists of an Android service that includes several internal components for UI event logging, breakpoint ground truth annotation logging, as well as the mechanisms for machine learning including feature extraction and classifying (using an embedded Weka [19] engine).

A. *Execution Modes*

Attelia can execute in ground truth annotation mode, off-line training mode or real-time breakpoint detection mode. In the annotation and detection modes, the UIEventLogger component listens to the stream of incoming UI events and records relevant events to the log file.

- **Ground truth collection:** In this mode, users manually provide ground truth about breakpoints during application usage. Figure 2 shows a screen-shot of Attelia, with our Annotation widget floating on the screen. While manipulating ordinary Android applications, users push the floating button when they are switching activities. The Attelia service records the stream of UI events (excluding those from the annotation button) and breakpoint timestamps (moments when the annotation button was pushed).
- **Off-line model training:** In this mode, feature extraction and classifier training is executed off-line, using the previously-stored sensor and ground truth data.

TABLE I. APPROACHES OF KNOWLEDGE COLLECTION FOR BREAKPOINT DETECTION

Approaches on Knowledge Source of Breakpoint	Examples of Data Types
Application-specific breakpoint knowledge	explicit breakpoint declaration inside application, explicit future breakpoint forecast inside application
Runtime status/event of systems and applications	stack trace, number of threads, thread names, memory consumption Android API invocation, system call invocation, rendered screen image, Low-level GUI events, switches between applications

TABLE II. TIMINGS OF KNOWLEDGE INPUT AND DATA COLLECTION

Approaches on Knowledge Source of Breakpoint	Knowledge on Breakpoints: When? By Who? and How?		Data Collection at Application Run-Time
	Application Development Phase	System Training Phase	
Application-specific breakpoint knowledge	Embedding additional API calls to provide explicit breakpoint knowledge (by application developer)	None	From API calls embedded inside running applications
Runtime status/event of systems and applications	None	Ground truth annotation of collected status/event information (by application users)	From the middleware and operating system

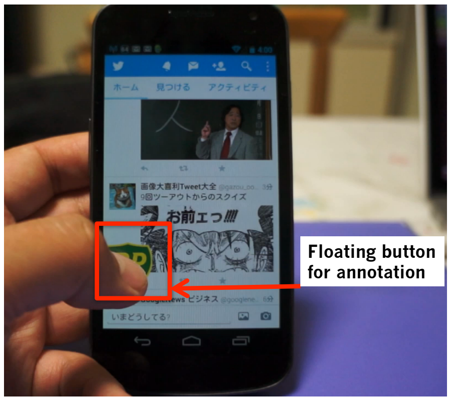

Fig. 2. Ground Truth Annotation with Attelia

TABLE III. UIEVENT COLLECTED IN ATTELIA

Event Types	Events
View	`View clicked`, `View long clicked`, `View selected`, `View focused`, `View text changed`, `View selection changed`, `View text traversed at movement granularity`, `View scrolled`
Transition	`Window state changed`, `Window content changed`
Notification	`Notification state changed`

- **Real-time mobile breakpoint detection:** Sensing, feature extraction, and classification with a previously-trained model is performed in real-time on a smartphone.

B. Sensing Data and Features

To obtain the stream of UI events from the middleware, we use the Android Accessibility Framework. Using this framework, Attelia can collect UI events and data about the UI components the user is interacting with. A list of the UI Events we collect is shown in Table III.

From these events, we extract the 45 features outlined in Table IV. These features are extracted for each "time frame" and stored during ground truth annotation, and are input to the Weka machine learning system for classifying during breakpoint detection. We attempted to be exhaustive in providing possible features to capture as many characteristics of the real execution environment as possible.

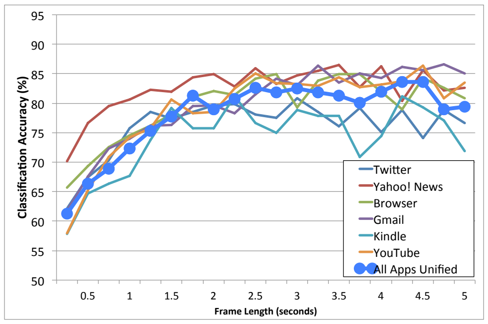

Fig. 3. Classification Accuracy and Frame Length

C. Frame Length

With an expectation that our choice of time frame length T_f will affect our ability to perform breakpoint detection, we conducted a small user study to investigate the impact of frame length on detection accuracy. Eight participants were recruited: university undergraduate and graduate students and staff with ages between 18 and 27 years, who use smartphones daily. Each participant manipulated five common Android applications (Twitter, Yahoo News, YouTube, Kindle, Browser) for 5 minutes each (per application) performing everyday tasks, and indicated their breakpoints using our floating annotator button. Participants used a Samsung Galaxy Nexus [20] smartphone running Android version 4 for the experiment.

Figure 3 shows the classification accuracy results with different frame lengths (0.25 to 5 seconds), using 10-fold cross validation on Weka 3.7.9 and J48 classifier. The data for each application is aggregated from all eight participants, and is represented as a separate line in the graph. An additional line in the graph (bolded) represents all application data aggregated together from all the participants. Accuracy is low when the frame length is very short (e.g., 0.25 seconds), because there are not enough sensed UI events within that time span to achieve a high classification accuracy. However, around 2 to 2.5 seconds, the accuracy begins to stabilize. At the 2.5-second setting, accuracy was 82.6%, precision was 82.7% and recall was 82.3%.

D. Power Saving

To save power, we disable real-time feature extraction and classification when the device screen is off, as we are

TABLE IV. FEATURES USED IN ATTELIA

Feature Types	Features
Rate of occurrence of each UI Event type inside the frame	snipped (one for each event type presented in Table III)
Statistics on the status of the event source UI component	*rate(*`isEnabled`*)*, *rate(*`isChecked`*)*, *rate(*`isPassword`*)*
Statistics on the events' timings in the frame	*min_timegap*, *mean_timegap*, *max_timegap*, *stdev_timegap*
Statistics on the location of the event source UI components	*min.*, *mean.*, *max.*, *stdev.*, *the value of the smallest rectangle*, *the value of the biggest rectangle* of `X-left`, `X-right`, `X-width`, `Y-top`, `Y-bottom`, `Y-height`

TABLE V. COMPARISONS OF POWER CONSUMPTION OVERHEAD

Sensor Type	Frequency (Hz)	Overhead (mW)
UI Events	10	51.70
Accelerometer	120	102.90
	60	48.76
	15	12.08
Gyroscope	100	158.88
	50	129.24
	15	74.04

concerned with detecting breakpoints when the user is engaged with the device. In addition, if no UI event occurs within a given time frame, no classification is performed.

Table V shows a power comparison between using our UI events and using common sensors. We used a Samsung Galaxy Nexus with Android 4.4.4 and measured the data with a Monsoon Power Monitor [21]. Each table value is the average of five 5-minute measurements. The result shows that the overhead of our UIEvent data collection software is quite low compared with other sensors and considering that multiple types of sensors, such as the accelerometer, gyroscope and GPS, are used in combination for many activity recognition systems,

In Attelia, since the number of incoming UI events depends on user interaction, we looked to our user study data to determine an appropriate number. Based on the data collected from 30 users for 16 days, the average number of UI events was 10.6 per second on average ($min = 1$, $max = 549$, $stdev. = 15.1$) during users' active manipulation of their device. We then logged the power consumption using Android instrumentation that fired approximately 10 UI events every second. To compare to the other sensors, we implemented a basic application which reads and stores the sensor data with the specified frequency.

E. Portable Implementation

Attelia is implemented as a "Service" inside the Android platform. By appropriately setting the permissions for the service, it can log the stream of UI events, such as tapping, clicking, and scrolling or modifications of UI components inside the currently-active Android application without requiring root privileges. This implementation allows the service to be distributed through the Google Play store and contributes to the deployability of the system to end users.

VI. CONTROLLED USER STUDY

Next, using this implementation, we conducted a controlled user study to better understand how the Attelia service can be used. Specifically, we investigated whether notifications provided to users at their detected breakpoints leads to reduced cognitive load, compared to different notification provision strategies, including random, and non-breakpoint timings.

A. Participants

We recruited 37 participants for the study, including university students, staff members and research engineers, with ages between 19 and 54. All of the participants use smartphones in their daily lives. Subjects were not paid for their participation and were not told the specific purpose of the study.

B. Experimental Setup

For the study, we provided participants with Samsung Galaxy Nexus smartphones running Android 4.3. The notification feature of the Android platform was disabled. The Attelia service, along with six representative Android applications (Twitter, Gmail, Yahoo News, YouTube, Kindle, Browser) were installed on each phone. The Attelia service was configured to detect breakpoints in real-time, with a J48 decision tree classifier (trained on our previous users) that executes periodically with a 2.5-second time frame T_f.

Users were exposed to four different strategies for being interrupted by notifications: disabled (no notification), random timing, breakpoint timing (our approach) and non-breakpoint timing (interrupting at times that our system determines as inopportune). The three latter strategies were configured to wait at least 30 seconds between two consecutive interruptions.

C. Interruptive Tasks

When users were interrupted, a full screen pop-up appeared on the screen to ensure that the interruption would not go unnoticed. The pop-up contains the first paragraph from a news article, and users were given a task to perform: read the paragraph and select an appropriate title for the article given three options. This interruptive task was taken from similar studies of interruptions [22], [23]. Subjects were instructed to answer the question as quickly and accurately as possible. After the user answered the question, the pop-up disappeared and the user could return to the original task that she was performing.

D. Experiment Procedure

Our experimental procedure contained two parts. In the first part, each user was given a printed email and was told to compose and send an email with this specified text using the Gmail app. Each user repeated this task five times, with different text and a different strategy. In the second part, each user was asked to use each of the other selected applications (Twitter, Yahoo News, YouTube, Kindle, Browser) as they "normally would" for 5 minutes each, and experienced a different strategy with each application.

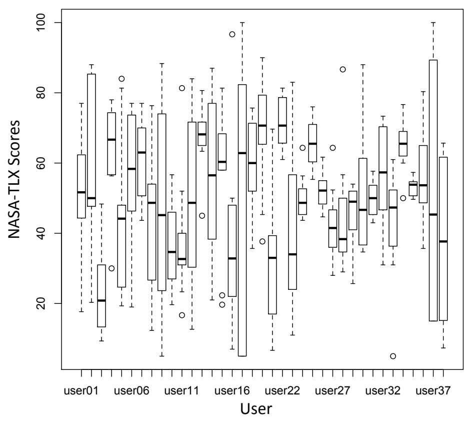

Fig. 4. Variance of NASA-TLX WWL Scores

TABLE VI. TWO CLUSTERS IN THE CONTROLLED USER STUDY

Cluster name	Users	Mean WWL Stdev.
“sensitive”	19	23.11
“insensitive”	18	9.92

The order of the email texts (part 1), applications (part 2), and notification strategies were counterbalanced using a balanced Latin Square to remove ordering effects. As there were 4 strategies, and the email and app use tasks were performed 5 times, each user saw one strategy twice, which was randomly selected. A repeated measures within-subject design was used with the notification strategy as factors.

E. Measurements

To measure user's subjective cognitive load, users answered the NASA-TLX[24] questionnaire after each task (total of 10 times per user).

F. Results: Subjective Cognitive Load

Across users, we saw differences in the range of subjective cognitive load (or weighted workload (WWL)) means and variances across the different strategies, as shown in Figure 4. This fact motivated us to try to identify clusters within our user population.

Based on a hierarchical clustering using the Ward method and Euclidean distance on the variance of each users NASA-TLX WWL scores, we identified 2 distinct clusters. Table VII shows the number of users and the mean personal WWL score standard deviation in each cluster respectively, based on the K-means clustering with the Hartigan-Wong method. Since this clustering was based on the variance of each user's scores, we named those two clusters “sensitive” (those with higher variance among the different strategies), and “insensitive”. Figure 5 shows the average NASA-TLX WWL scores for the different notification strategies, for the two clusters respectively.

For the “sensitive cluster”, the most significant finding is that our breakpoint strategy (“BP”) results in a 46% reduction in cognitive load, compared to the random strategy (“Random”), which approximates how people are currently interrupted by notifications with the standard Android notification

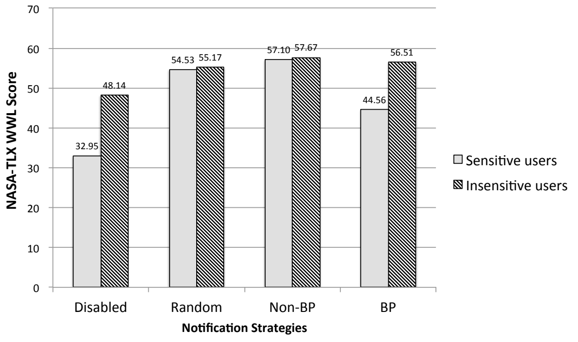

Fig. 5. NASA-TLX WWL Scores for Each Cluster

system. When compared to the baseline (“Disabled” strategy with a cognitive load score of 32.95) with no notifications, the breakpoint strategy resulted in only an increase of 35% in cognitive load, while the random strategy resulted in an increase of 66%. Also, as we expected, the non-breakpoint strategy (“Non-BP”) where the system was configured to intentionally deliver notifications only at the non-breakpoint timings, resulted in the highest increase in cognitive load from the baseline: 73%. A Friedman test revealed a significant effect of notification strategy on the WWL score ($\chi^2(3) = 16.5, p < 0.05$).

A post-hoc pair-wise comparison using Wilcoxon rank sum tests with Holm correction showed the significant differences between “Disabled” and “Random” ($p < 0.01$, $\gamma = 0.34$), between “Disabled” and “non-BP” ($p < 0.01$, $\gamma = 0.39$), between “Disabled” and “BP” ($p < 0.05$, $\gamma = 0.29$), between “Random” and “BP” ($p < 0.05$, $\gamma = 0.24$), and between “non-BP” and “BP” ($p < 0.05$, $\gamma = 0.26$). Between “Random” and “non-BP”, a statistical difference was not observed.

For the “insensitive cluster”, the graph shows the insensitivity of the users. As expected for this user group, our Friedman test and pair-wise test with Wilcoxon rank sum tests show only significant differences between “Disabled” and the other strategies. (Friedman test with the effect of notification strategy on the WWL score ($\chi^2(3) = 9.4, p < 0.05$)). The significant differences from the post-hoc test using Wilcoxon rank sum tests with Holm correction are observed between “Disabled” and “Random” ($p < 0.01$, $\gamma = 0.30$), between “Disabled” and “non-BP” ($p < 0.01$, $\gamma = 0.35$), and between “Disabled” and “BP” ($p < 0.01$, $\gamma = 0.34$).

VII. IN-THE-WILD USER STUDY

Based on these promising findings from our controlled study, we conducted an “in-the-wild” or field user study to better understand how the Attelia service can perform in users' real environments. By installing Attelia on each participant's own smart phone, we investigated whether notifications provided to users at their detected breakpoints leads to reduced cognitive load during participants' everyday smartphone usage.

A. Participants

We recruited 30 (20 male and 10 female) participants for the study. They are university students and staff members, with ages between 18 and 29. 20 participants came from computer science and information technology related departments, while the other 10 came from other schools, such as economics, psychology, or social sciences. All of the participants use smartphones with Android OS version 4.3 (or above), in their daily lives. Subjects were paid $60 for their participation.

B. Experimental Setup

For the study, we prepared the Attelia service with some additional experiment-related logics and parameters for installation onto each participant's smartphone. Similar to the controlled user study, the service was configured to detect breakpoints in real-time, using the same J48 decision tree classifier trained earlier, that executes with a 2.5-second time frame T_f. The service was also configured to use three different notification strategies: disabled (no notification), random timing and breakpoint timing (our approach). Each day, our study software randomly chose one of the three strategies for use, for notifying the user throughout the day.

The minimum interval between two consecutive notifications was set to 15 minutes, the maximum interval to 30 minutes, and the daily maximum number of interruptive tasks to 12. The study software also was configured to only send interruptions from 8AM to 9PM daily. The parameter values were carefully chosen after interviewing prospective participants about their daily lives, to get a sufficient number of data samples without overburdening them.

C. Interruptive Tasks

When users were interrupted, two full screen pop-ups appeared on the screen. The first screen asked if the timing was during a natural breakpoint. Regardless of the answer to the first question, the second pop-up appeared, presenting the same interruptive task as we used for the controlled user study. Users were instructed to answer the question as quickly and accurately as possible. After answering, the pop-up disappeared and the user could return to her original task.

D. Experiment Procedure

Our experimental procedure consists of three parts. (1) At the beginning of the user study, each participant had a meeting with a researcher. After the participant received information and instructions for the user study, he signed the consent form to join the study. We installed our Attelia service on his smartphone, and turned it on. (2) After the meeting, each participant experienced our notifications for 16 days. As described above, the notification strategy was changed by the service daily. At the end of each day, each participant evaluated his/her notification experience for that day, filling out the NASA-TLX survey. (3) After the 16-day period was completed, participants filled out the post-experiment survey, uninstalled the Attelia service, and were paid.

E. Measurements

The Attelia service recorded the time taken to respond to the first and second notifications, time to answer the quiz, and the answer to the quiz. The data was uploaded to our server every night. The NASA-TLX questionnaires (implemented as a web page on our web server) were sent to each user via email every night, thus the survey results were stored inside our database on the server.

TABLE VII. Two Clusters in the Wild User Study

Cluster name	Users	Mean WWL Stdev.
"sensitive"	13	21.38
"insensitive"	14	8.19

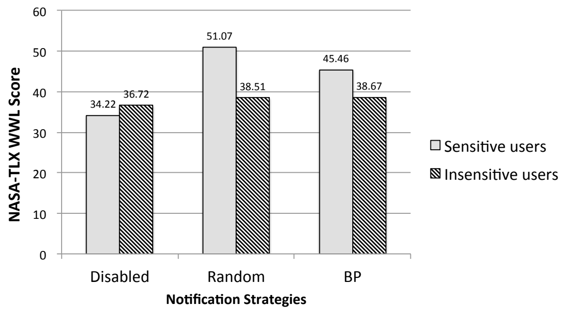

Fig. 6. NASA-TLX WWL Scores for Each Cluster

F. Results: Subjective Cognitive Load

We collected NASA-TLX surveys from each of the 30 participants for 16 days. Data from 27 seven users were used in the analysis as the logged data for the remaining 3 users was not properly recorded and uploaded to the server or they forgot to fill the daily survey.

Again, for the result, due to the observed differences in the variance of NASA-TLX WWL scores ("sensitivity") between users, we first split users into 2 clusters, according to the results from a hierarchical clustering. Table VII shows the number of users and the mean personal WWL score standard deviation in each cluster respectively. We did not observe differece in score-sensitivity between gender. Figure 6 shows the average NASA-TLX WWL scores for the different notification strategies, for the two clusters respectively.

For the "sensitive cluster", we observe the same trend as we saw in our controlled user study. With the baseline cognitive load 34.22 for the "Disabled" strategy, our breakpoint strategy ("BP") results in an 33% reduction in cognitive load, compared to the random strategy ("Random"), which approximates how people are currently interrupted by notifications. When compared to the baseline ("Disabled" strategy) with no notifications, the "breakpoint" strategy resulted in only an increase of 33% in cognitive load, while the "Random" strategy resulted in an increase of 49%. A Friedman test revealed a significant effect of notification strategy on the WWL score ($\chi^2(2) = 8.5$, $p < 0.05$). A post-hoc pairwise comparison using Wilcoxon rank sum tests with Holm correction showed significant differences between "Disabled" and "Random" ($p < 0.01$, $\gamma = 0.37$) and between "Random" and "BP" ($p < 0.05$, $\gamma = 0.20$),

G. Results: Response Time for the First Pop-up

Next we present the response time to the first pop-up as shown in Figure 7. The response time denotes the timestamp difference between when the first pop-up appeared on the screen to when it was answered by the user. We collected 1130 data points for "random" and 1032 data points for "breakpoint" strategies. The average response time was 3.18

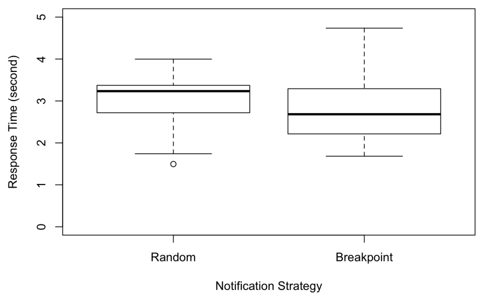

Fig. 7. Response Time to the First Pop-up

seconds in "random" and 2.77 seconds in "breakpoint" strategy respectively. Our Wilcoxon Signed-rank test showed that there is a significant effect of strategy ($W = 343$, $Z = -3.19$, $p < 0.05$, $\gamma = 0.37$).

VIII. Discussion

Through our user studies, we proved the effectiveness of Attelia not only in a controlled environment but also in "in-the-wild" real user environments. We showed that notifications delivered at breakpoint timings identified in real-time by Attelia, resulted in cognitive load levels that were 33% less than notifications delivered at random moments, and decreased the response time to the notification by 12%. Now we discuss further research opportunities that this research enables.

Firstly, we are interested in further studying "insensitive" users and determining whether other forms of notification management may be of use to them. We see another opportunity for further improvement of the system in terms of the detection performance, possibly with personalization of the model with active learning and a longitudinal user study.

Deployment of the system with "real" notifications from "real" Android applications is yet another challenge for us. Due to the limitation of the Android platform where no APIs for customizing or overriding the standard Android notification system exist, Attelia currently uses the artificial interruptive notification system we built for our evaluation. However, Attelia could certainly export its "interruptibility API" , based on the real-time detection results, for other applications to use.

We have reached a time when users carry and use multiple mobile and wearable devices, so support for those devices and sensors are our obvious next challenge. We have already confirmed that our Attelia service runs on Google Glass in addition to conventional Android smartphones and tablets. Our hypothesis is that, as long as those devices have any kind of user interface with which to interact, our approach of "application usage as a sensor" will be applicable to those devices. Breakpoint detection while the user is not actively manipulating the device is another very interesting research area, probably requiring the use of additional sensors, such as GPS, accelerometer, proximity and light sensors.

IX. Related Work

In early work on finding appropriate moments for interruption, Horvitz et al. inferred interruptibility accurately in desktop computing environments, by using context information, such as interaction with computing devices, visual and acoustical analyses, and online calendars [25]. For this, recognition was performed in a posteriori manner.

Work by Begole et al. [26], Horvitz et al. [27] are in the first generation of systems with real-time model construction and detection of interruptibility although their systems used dedicated custom hardware. Iqbal et al.[28] built OASIS which defers desktop-based notifications until suitable timings of interruption were detected in real-time. They focused on the detection of breakpoints [8], based on user interactions with an application and provided user annotations.

More recently, interruptibility research has focused on mobile devices. Ho et al. used wireless on-body accelerometers to trigger interruptions in the timing of user's switch between activities [29]. The authors found that the users' annoyance was minimal when interruptions were triggered at the moments of switching between activities. While their approach needs an external on-body sensor, Attelia uses only the smartphone. Fischer et al. focused on the interruptibility immediately after phone activities including completion of phone calls and text messages [30]. They found that the users tend to be more responsive to notifications after mobile phone activities than at random other timings. While the authors' approach focused on phone-related activities, our approach uses applications available on the market and installed on the phone, including phone-call and text messaging. Smith et al. focused on disruptive phone calls and took the approach of "mitigation" by automatically setting phone call ring tones to different modes, such as silent answering, declining, and ignoring [31]. A user study showed that their approach was useful, even with user concept drift. Their mitigation approach is orthogonal to our scheduling approach, thus a combination of both approaches is possible.

Hofte et al. used smartphones for interruptibility study. They used the experience sampling methodology on location, transit status, company and activities to build a model for interruptibility [32]. Also Pejovic et al. explored whether, and how, suitable moments for interruption can be identified and used in smartphones [33]. Based on "broader context" including activity, location, time of day, emotions and engagement, their InterruptMe system decides interruptibility. Their approach determines timing based on smartphone sensor data. In contrast, our approach relies on user interaction, focusing on the period while the user is actively manipulating the device. According to our power consumption measurement in Section V, we found that the power overhead for our approach was significantly lower than the physical sensor approaches. Also as users will continue to have an increasing number of devices, we believe our approach will be more effective because our approach can be easily deployed to devices with and without physical sensors. Also their implementation relies on information on user's activity, such as work mode, emotion, and company, manually provided by the user in order to infer interruptibility. On the other hand, our system does not need any manually-provided information, simply relying on the UI events coming from the Android system.

X. Conclusion

In this paper, we proposed Attelia, a novel middleware that identifies when interruptive notifications should be delivered to minimize impact on user's cognitive load, supporting attention-aware adaptation for maintaining a user's productivity. Attelia detects such moments in real-time, uses only the mobile devices that users naturally use and wear, without any modification to applications, and without any dedicated psycho-physiological sensors or physical sensors on the devices. Our evaluation proved the effectiveness of Attelia. Both our controlled and "in-the-wild" evaluations showed that notifications presented at breakpoints detected by Attelia resulted in significantly lower cognitive load and response time compared to randomly-timed notifications.

ACKNOWLEDGEMENTS

This work was supported (in part) by Ministry of Education, Culture, Sports, Science and Technology (MEXT) Grant-in-Aid for the "Research and Development for Big Data Use and Application" and for the "Program for Leading Graduate Schools".

REFERENCES

[1] Google Inc., "The new multi-screen world — think with google," Aug. 2012. [Online]. Available: http://www.google.com/think/research-studies/the-new-multi-screen-world-study.html

[2] Deloitte Touche Tohmatsu Limited, "Deloitte global mobile consumer survey 2012," Sep. 2012. [Online]. Available: http://www.deloitte.com/view/en_GB/uk/industries/tmt/telecommunications/global-mobile-consumer-survey-2012/

[3] D. Garlan, D. Siewiorek, A. Smailagic, and P. Steenkiste, "Project aura: toward distraction-free pervasive computing," *Pervasive Computing, IEEE*, vol. 1, no. 2, pp. 22 –31, april-june 2002.

[4] P. D. Adamczyk and B. P. Bailey, "If not now, when?: the effects of interruption at different moments within task execution," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2004, pp. 271–278.

[5] B. P. Bailey and J. A. Konstan, "On the need for attention-aware systems: Measuring effects of interruption on task performance, error rate, and affective state," *Computers in Human Behavior*, vol. 22, no. 4, pp. 685 – 708, 2006.

[6] M. Czerwinski, E. Cutrell, and E. Horvitz, "Instant messaging: Effects of relevance and timing," in *People and computers XIV: Proceedings of HCI*, vol. 2. British Computer Society, 2000, pp. 71–76.

[7] J. G. Kreifeldt and M. E. McCarthy, "Interruption as a test of the user-computer interface," in *JPL Proceeding of the 17 th Annual Conference on Manual Control*, 1981, pp. 655–667.

[8] D. Newtson and G. Engquist, "The perceptual organization of ongoing behavior," *Journal of Experimental Social Psychology*, vol. 12, no. 5, pp. 436–450, 1976.

[9] S. Gould, D. Brumby, A. Cox, V. González, D. Salvucci, and N. Taatgen, "Multitasking and interruptions: a sig on bridging the gap between research on the micro and macro worlds," in *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, 2012, pp. 1189–1192.

[10] F. R. Zijlstra, R. A. Roe, A. B. Leonora, and I. Krediet, "Temporal factors in mental work: Effects of interrupted activities," *Journal of Occupational and Organizational Psychology*, vol. 72, no. 2, pp. 163–185, 1999.

[11] C. Speier, J. S. Valacich, and I. Vessey, "The influence of task interruption on individual decision making: An information overload perspective," *Decision Sciences*, vol. 30, no. 2, pp. 337–360, 1999.

[12] S. T. Iqbal and E. Horvitz, "Notifications and awareness: A field study of alert usage and preferences," in *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, ser. CSCW '10, 2010, pp. 27–30.

[13] S. T. Iqbal and B. P. Bailey, "Understanding and developing models for detecting and differentiating breakpoints during interactive tasks," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '07, 2007, pp. 697–706.

[14] S. T. Iqbal and B. P. Bailey, "Investigating the effectiveness of mental workload as a predictor of opportune moments for interruption," in *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '05, 2005, pp. 1489–1492.

[15] S. T. Iqbal and B. P. Bailey, "Leveraging characteristics of task structure to predict the cost of interruption," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '06, 2006, pp. 741–750.

[16] J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, "Participatory sensing," in *In: Workshop on World-Sensor-Web (WSW 2006): Mobile Device Centric Sensor Networks and Applications*, 2006, pp. 117–134.

[17] M. Hoshiba, O. Kamigaichi, M. Saito, S. Tsukada, and N. Hamada, "Earthquake early warning starts nationwide in japan," *Eos, Transactions American Geophysical Union*, vol. 89, no. 8, pp. 73–74, 2008.

[18] E. Haapalainen, S. Kim, J. F. Forlizzi, and A. K. Dey, "Psycho-physiological measures for assessing cognitive load," in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, 2010, pp. 301–310.

[19] *Weka 3: Data Mining Software in Java*, http://www.cs.waikato.ac.nz/ml/weka/, Machine Learning Group at the University of Waikato.

[20] *Galaxy Nexus*, http://www.samsung.com/us/mobile/cell-phones/GT-I9250TSGGEN, Samsung Electronics Co., Ltd.

[21] *Monsoon Power Moniter*, http://www.msoon.com/LabEquipment/PowerMonitor/, Monsoon Solutions Inc.

[22] B. Bailey, J. Konstan, and J. Carlis, "Measuring the effects of interruptions on task performance in the user interface," in *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, vol. 2, 2000, pp. 757–762 vol.2.

[23] B. P. Bailey, J. A. Konstan, and J. V. Carlis, "The effects of interruptions on task performance, annoyance, and anxiety in the user interface," in *IFIP TC.13 Conference on Human Computer Interaction (Interact 2001).*, 2001, pp. 593–601.

[24] S. G. Hart and L. E. Staveland, "Development of nasa-tlx (task load index): Results of empirical and theoretical research," in *Human Mental Workload*, ser. Advances in Psychology, P. A. Hancock and N. Meshkati, Eds. North-Holland, 1988, vol. 52, pp. 139 – 183.

[25] E. Horvitz and J. Apacible, "Learning and reasoning about interruption," in *Proceedings of the 5th International Conference on Multimodal Interfaces*, ser. ICMI '03, 2003, pp. 20–27.

[26] J. B. Begole, N. E. Matsakis, and J. C. Tang, "Lilsys: Sensing unavailability," in *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, ser. CSCW '04, 2004, pp. 511–514.

[27] E. Horvitz, P. Koch, and J. Apacible, "Busybody: Creating and fielding personalized models of the cost of interruption," in *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, ser. CSCW '04, 2004, pp. 507–510.

[28] S. T. Iqbal and B. P. Bailey, "Oasis: A framework for linking notification delivery to the perceptual structure of goal-directed tasks," *ACM Trans. Comput.-Hum. Interact.*, vol. 17, no. 4, pp. 15:1–15:28, Dec. 2010.

[29] J. Ho and S. S. Intille, "Using context-aware computing to reduce the perceived burden of interruptions from mobile devices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '05, 2005, pp. 909–918.

[30] J. E. Fischer, C. Greenhalgh, and S. Benford, "Investigating episodes of mobile phone activity as indicators of opportune moments to deliver notifications," in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, ser. MobileHCI '11, 2011, pp. 181–190.

[31] J. Smith and N. Dulay, "Ringlearn: Long-term mitigation of disruptive smartphone interruptions," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2014 IEEE International Conference on*, March 2014, pp. 27–35.

[32] G. H. H. ter Hofte, "Xensible interruptions from your mobile phone," in *Proceedings of the 9th International Conference on Human Computer Interaction with Mobile Devices and Services*, ser. MobileHCI '07, 2007, pp. 178–181.

[33] V. Pejovic and M. Musolesi, "InterruptMe : Designing Intelligent Prompting Mechanisms for Pervasive Applications," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '14, 2014, pp. 395–906.