# Sunshine: The Service for YOU to eNhance Self-management Helpfully and Intelligently from Now to forEver

Bohun Seo
School of Computing
KAIST
Daejeon, South Korea
bo_hoon_e@kaist.ac.kr

Hayeon Lee
School of Computing
KAIST
Daejeon, South Korea
hayeon926@kaist.ac.kr

Insu Jang
School of Computing
KAIST
Daejeon, South Korea
insu.jang@kaist.ac.kr

***Abstract*—In recent days, as smartphones came into wide use, many people use smartphones all the time and are disturbed by smartphones, students cannot focus on their study, employee cannot concentrate on their job, and so on. There are smartphone apps that help people solve this problem, however, they are not flexible and inefficient to get the effect. We suggest SUNSHINE, which runs as a part of Android system. It provides natural app control experience and automatically filters out the Internet web contents by using semantic analysis. Also, it adopts gamification concept to give users a motivation to keep using SUNSHINE.**

***Keywords*—*addiction; mobile; gamification; semantic analysis;***

## I. Introduction

Recently, smartphones help people lead comfortable life. However, the increasing use of smartphones can disturb people from concentrating their everyday life [1-3]. For example, there is a phenomenon that students cannot concentrate on their study during exam period in the library due to smartphones. To solve this problem, many apps are released in Google Play, which is an application market provided by Google. Study helper manages students' study time and restrict all other apps running during measuring study time [4]. Self-control for study also restricts apps running but users can check whether each app should be restricted or not when using [5]. Hence, the motivation of usage is low and users can freely use all apps if they want. Also, both applications are running as an application, not a part of system, so app blocking is unnatural and malfunctions at some point.

To solve the problems and provide more natural experience, we suggest SUNSHINE that runs as a system service and provides many functions for users to concentrate their work. When SUNSHINE mode is enabled, the dedicated launcher starts and users can only run allowed apps. We also provide gamification service that users can gather points by studying some time and they can user these points to make level up or use blocked apps such as Facebook, the most famous social networking app. App control is done in the system level, so it works perfectly and naturally. Also, people tend to search a lot of things related to study in the Internet. However, there are also a lot of contents that people can be deceived. To figure out these contents, we provide a semantic analysis web browser, automatically analyzes each web content and block if it is determined not to be related with productivity.

The remaining of this paper is organized as follows. In section 2, we present related works and background knowledge that we used. In section 3, we describe an overview of the SUNSHINE architecture. In section 4, we explain each component in the architecture in detail. Finally, in section 5, we discuss our contribution, limitations, and future works.

## II. Backgrounds

In this section, we introduce essential background knowledge to understand our project. We analyze several existing applications and devised gamification contents. Moreover, we also adopt semantic analysis for providing flexibility.

### *A. Related Works*

We searched various applications that have similar functionalities with our purpose as in Fig. 1. However, they operate as a third-party application which is limited for controlling and observing system states. Due to these characteristics, they have some limitations. Specifically, major inefficiency is in blocking other applications which may disturb user's concentration. First they have android service component to monitor current foreground application name, and if name is not matched with predefined list, service would

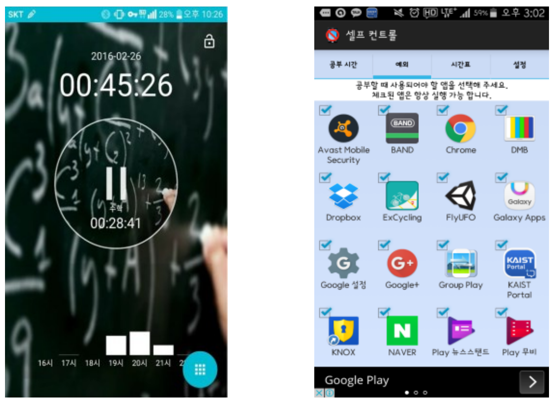


Fig. 1. Existing applications which help users enhance their productivity

make their own application be foreground. This sequence must require the context-switching, because all application can be executed by users anyway. Another limitation is that these applications delegate the authority of selecting blocked applications to users. If users select application which category is Communication or Game, user would be able to use this application though above applications are running. It still leaves temptation to users. Moreover, due to third-party application of android, they can't protect themselves perfectly. We had performed some experiments for these applications with applying malicious attempt to stop by pushing home key to call 'Recent task list'. The results showed they have possibility to be stopped forcibly by users.

### B. *Semantic Analysis*

Semantic analysis is a study of semantics, or the structure and meaning of speech [7]. Latent Semantic Analysis (LSA) is one of the semantic analysis techniques and is used to estimate similarity ratings of word pairs [8, 9]. However, it requires a neural net mechanism and learning procedure to extract high quality of classification. To avoid it, D. Bollegala et al proposed an algorithm measuring semantic similarity between words by using web search engines, such as Google [9]. They combined the result from two techniques: page count based similarity scores and extracting lexico-syntatic patterns from snippets. They integrated patterns and page counts to increase accuracy. The second method also requires learning with dataset to measure patterns, however, it is still enough to extract reasonable words similarity only with page counts if one of two words to be compared is fixed.

### C. *Gamification*

Gamification is a form of service packaging where a core service is enhanced by a rules-based service system that provides feedback and interaction mechanisms to the user with an aim to facilitate and support the users' overall value creation [10]. The effectiveness of gamification relies on feedback loops which influence user behavior. Scores, i.e., quantitative evaluations of behavior in a game play an essential role in this loop. Designing score relates the situated user context to other behaviors and allow users to internalize externally intended behavior and goals. For score design, First, selecting the *activity types* and deciding which *quality standardization* of these activities should be taken into account. Second, qualities need to be evaluated with respect to *goals* which turns qualities into criteria. Taking many goals into account increases the chance that users can find their own relevant goals reflected in a gamified application [11].

## III. SUNSHINE ARCHITECTURE.

SUNSHINE is implemented by modifying framework source code, so we can control operations of other system services for our dedicated purposes. SUNSHINE has three components as you can see in Fig. 2. They are PAService, Dedicated launcher and browser. Each component takes its specific role and communicates with others. Dedicated launcher and browser are independent applications. PAService is implemented as system service and provides several API for dedicated launcher. SUNSHINE mode operates as 'another mode' compared with 'normal mode'. In SUNSHINE mode, launcher is changed automatically and PAService changes its

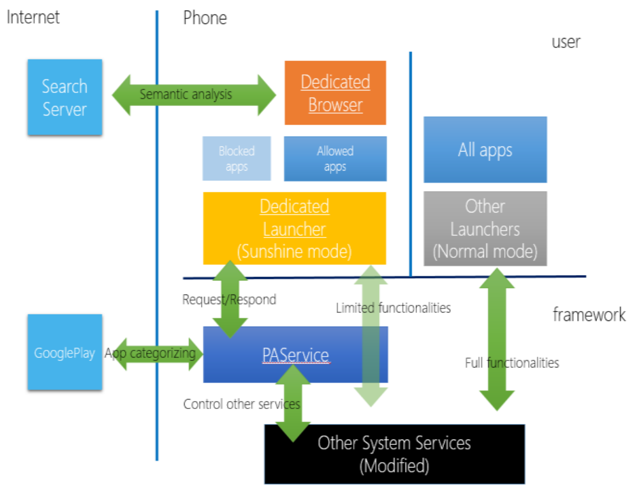


Fig. 2. SUNSHINE architecture

state to 'activated'. Then operations of several system services are modified to provide limited functionalities to our dedicated launcher. User who uses SUNSHINE mode, can't use blocked application. If sunshine mode is remaining, user can get points. Points can be consumed for using blocked application or making level up. Almost browser applications are blocked cause their category is Communication. Therefore, we provide dedicated browser. It allows user to search for only allowed areas which are relevant to studying. Moreover, all notifications are postponed in SUNSHINE mode. If user turns off the SUNSHINE mode, PAService turns to deactivated state which leads to be just same with normal usage.

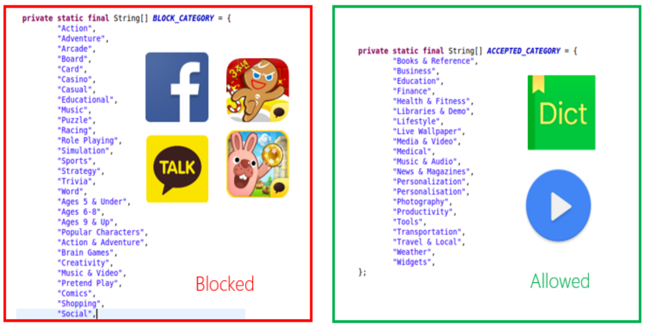
## IV. IMPLEMENTATIONS

In this section, we explain our implementation issues and solution specifically for each component respectively.

### A. *App Categorization*

Preexisting applications in Google Play are delegating authority of selecting blocked applications to user. Otherwise, our project performed app categorization in PAService automatically.

PAService is started when booting is complete. One of its main purpose is making and managing blocked application list after the booting or new application installed. For categorizing application, PAService uses predefined category list in Fig.3. Every application is categorized only once.



```
private static final String[] BLOCK_CATEGORY = {
        "Action",
        "Adventure",
        "Arcade",
        "Board",
        "Card",
        "Casino",
        "Casual",
        "Educational",
        "Music",
        "Puzzle",
        "Racing",
        "Role Playing",
        "Simulation",
        "Sports",
        "Strategy",
        "Trivia",
        "Word",
        "Ages 5 & Under",
        "Ages 6-8",
        "Ages 9 & Up",
        "Popular Characters",
        "Action & Adventure",
        "Brain Games",
        "Creativity",
        "Music & Video",
        "Pretend Play",
        "Comics",
        "Shopping",
        "Social",
```

```
private static final String[] ACCEPTED_CATEGORY = {
        "Books & Reference",
        "Business",
        "Education",
        "Finance",
        "Health & Fitness",
        "Libraries & Demo",
        "Lifestyle",
        "Live Wallpaper",
        "Media & Video",
        "Medical",
        "Music & Audio",
        "News & Magazines",
        "Personalization",
        "Personalisation",
        "Photography",
        "Productivity",
        "Tools",
        "Transportation",
        "Travel & Local",
        "Weather",
        "Widgets",
};
```

Fig. 3. Predefined category list for blocked application from Google play

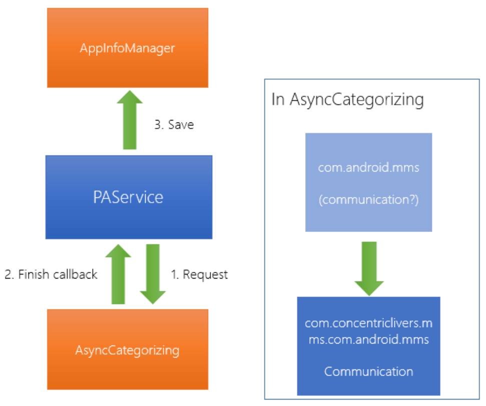


Fig. 4. Process of App categorization and two-phase-checking

We had examined all categories which are provided by Google Play and classified each category is proper for SUNSHINE mode or not. However, we used only categories in the Google Play. So there would be problem, if PAService tries to categorize application which is not registered on Google Play. For this case, we adopted two-phase-checking for app categorization. First PAService tries to categorize target app with exact package name. If it doesn't exist, use similar name. As you can see in Fig.4, default MMS application is not registered on Google Play. In this case, we just search the com.android.mms which is the package name of MMS application in Google Play. Then pick the first app name of result as similar name.

## B. *SUNSHINE System Service*

For managing SUNSHINE mode, we implemented system service called PAService. It has SUNSHINE mode activation flag which is referred by other system services. It provides API for dedicated launcher for getting blocked application list and initializing timer for point. It also has a role to allocate AsyncTask for categorizing application.

In Fig.5, we enumerate the coverage of PAService management. Left-side one is relevant to app categorizing that is already handled previous section. Right-side is the list of modified system services or classes when system is SUNSHINE mode.

### *1) App visibility and execution control*

App visibility control is essential for switching launcher and hiding our dedicated applications. Dedicated launcher and browser are basic components of SUNSHINE service, but these application are just third-party applications which can be uninstalled directly by user. We analyzed several system services like PackageManagerService (PMS), PhoneWindowManager (PWM) and ActivityManagerService (AMS) to gather information about launcher switching and app visibility [12-14].

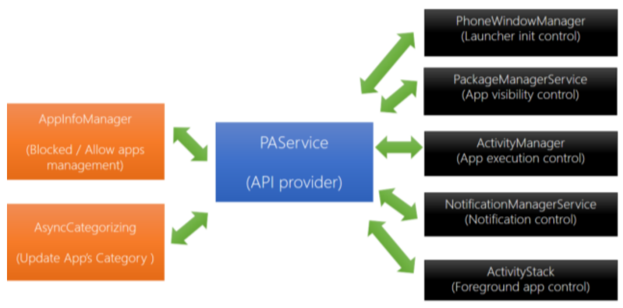


Fig. 5. Coverage of PAService management

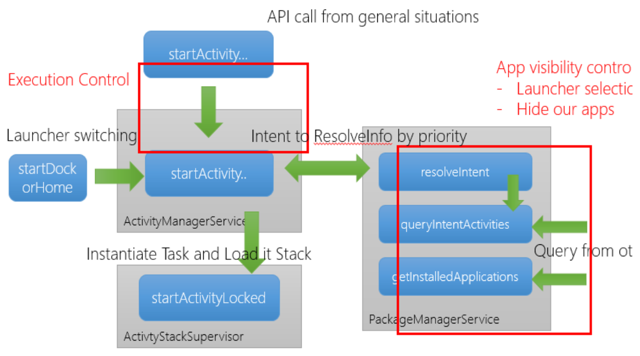


Fig. 6. App visibility and execution analysis

Basically, all installed packages are managed by PMS. So when request to start specific application by intent is generated, system checks target application's intent is valid by calling resolveIntent API of PMS. resolveIntent also calls queryIntentActivities method which is not only used for resolveIntent but also other purposes such as getting all installed applications as explained in Fig.6.

In SUNSHINE service, our launcher is only valid when SUNSHINE mode is activated. Otherwise, it should be hidden. So we exploited some PMS methods to drop our dedicated launcher by package name when SUNSHINE mode is activated. Similarly, our dedicated browser is only seen by our dedicated launcher, so when our launcher send request for getting installed application list, it pushes magic word to intent. Several modified PMS methods check its magic word and determine dedicated browser's destiny.

When our dedicated launcher is initialized, it loads blocked application list from PAService by calling API in Table I. and drops blocked applications. Hence user can't try to start blocked application in SUNSHINE mode. However, there are some attempt of starting activity from system like resolveActivity. So to make more perfect execution control, we exploited API call from general situations as explained in Fig.6.

### *2) Foreground activity detection*

In SUNSHINE mode, user can accumulate points by remaining it. User can consume points to execute blocked application. PAService also take a role to manage point acquiring and consuming, so efficiency method is essential to trace the foreground activity name.

TABLE I. SEVERAL PROVIDED API OF PASERVICE

| API name | description |
|---|---|
| getBlockedPackageList() | Get blocked package name list |
| isAppRestricted(packageName) | Check whether package is blocked |
| enableTempExecutePermission() | Aquire TEP for blocked apps |
| disableTempExecutePermission() | Release TEP |

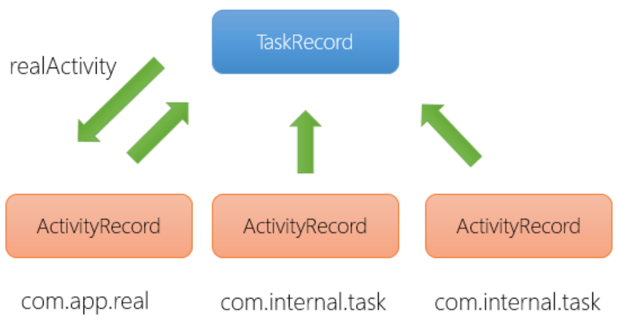


Fig. 7. Relationship between TaskRecord and ActivityRecord

We modified system service, so we can get a callback method invoked when foreground activity changed. When foreground activity changed, state of focused activity stack changed too [13]. So we exploited resumeTopActivityLocked method in ActivityStack. It notifies to PAService.

PAService classifies all applications by its package name. However, activity name would be differed in same package, because each package can have multiple activities. So directly usage of activity name leads wrong operations. For instance, in Fig. 7, installed package name is com.app.real, but it has three activities and two different package names. Two of them can't be detected as blocked application by PAService. So we use the package name of realActivity in TaskRecord, because it is activity which makes task start.

### 3) Notification control

When SUNSHINE mode is activated, all notifications are postponed. We analyzed NotificationManagerService [12] and modified it to intercept all notifications while SUNSHINE mode is running. The reason we selected enqueneNotificationInternal to intercept notification, in that method it determines target notification is duplicated in notification list. In that case, it just changes the essential information of notification. It requires many parameters to call, we made class to save its parameters. When user turns off SUNSHINE mode, all postponed notifications are notified by calling postDelayedNotification which is generated by us.

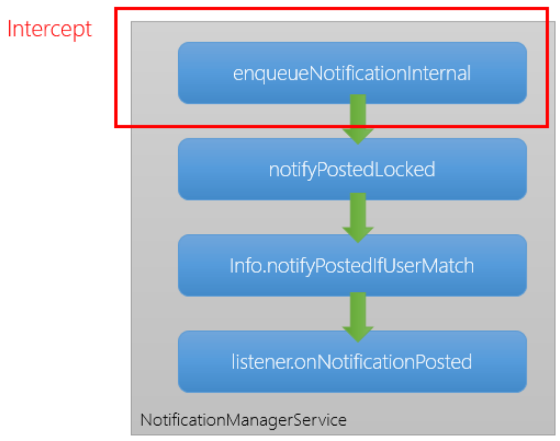

Intercept
enqueueNotificationInternal
notifyPostedLocked
Info.notifyPostedIfUserMatch
listener.onNotificationPosted
NotificationManagerService

Fig. 8. Method flow for each notification in NotificationManagerService

## C. Semantic Analysis Web Browser

There are many third-party web browsers that users can use, however, only the dedicated web browser can be used while sunshine mode in activated. The difference from them is that our web browser analyzes each web page content and block the it if the content is determined to be related to blocked categories. To find how the web page is related to blocked categories, we implemented a page count based semantic analysis algorithm to get a block score. The block score is used to determine whether the current web page should be blocked or not. The algorithm consists of five steps: perform readability extraction, remove unnecessary contents using Sanitizer, extract nouns with their term frequencies, query top ranked nouns with blocked or unblocked categories, and calculate the block score. We will examine each step in detail.

### 1) Performing readability extraction

Readability extraction functionality extracts the main article in a web page. Currently, it is being used in famous web browsers, i.e., Firefox, Chrome, Safari, and so on. Also, Mozilla Foundation implemented a standalone readability function as a Javascript script [15] and it is available to use the same function in Node.js server that we are using with much more improved performance, developed by Zihua [16]. After performing readability extraction, many unrelated elements such as pictures, advertisements, and menus are eliminated.

### 2) Removing unnecessary contents using Sanitizer

Readability extraction extracts main article including HTML tags and HTTL links. As they can affect the accuracy of semantic analysis, they should be eliminated and this is what Sanitizer does. Additionally, HTML tags still remaining and whitespaces are eliminated by using regular expressions. The result is a pure plaintext without any metadata.

### 3) Extracting nouns with their term frequencies

To use page count analysis, we need words, not a plaintext. Hence, we extract nouns in the plaintext. It can be done with wordpos module, a set of part-of-speech (POS) utilities using fast lookup in the WordNet database [17]. WordNet is a large lexical database of English containing nouns, verbs, adjectives, and adverbs. By using WordNet, we can extract English nouns in the plaintext really fast. Also, we count each word's frequency that how many each word is shown in the web page.

### 4) Querying top ranked nouns with blocked/unblocked categories

The reason we measure each word's frequency is to restrict the number of queries. With a realistic reason, we have limited number of permitted query when we use search API (In case of ours, we use Microsoft's Bing API that provides 5,000 queries per month at free of charge). Hence, we only query top three ranked words with blocked categories and unblocked categories as 'word AND (b1 OR b2 OR b3 OR …) where bn represents blocked categories or unblocked categories.

In addition to this mechanism, we determined that querying same words repeatedly is inefficient, so we also implemented a cache database by MongoDB. Hence, Sunshine first searches the data in the DB and if there is no data, query them in the search engine. After finishing querying, the search result information is saved in the DB.

### 5) *Calculating the block score*

After querying, the result will be six numbers: two numbers for each word that are the number of search result when a word is queried with blocked categories and with unblocked categories. According to page count based similarity extraction algorithm, the word is more related to the blocked categories than the unblocked categories when the former number is larger than the latter one for each word. Aggregating three determinations, the block score is calculated as the equation 1.

$$score = \sum_{i=1}^{3} d_i * \left(\frac{num_{blocked}}{num_{unblocked}}\right)^{d_i} * freq_i \qquad (1)$$

where $d_i = \begin{cases} 1 & if\ num_{blocked} > num_{unblocked} \\ -1 & otherwise \end{cases}$, $num_{blocked}$ is the number of search result with the i-th word and blocked categories, $num_{unblocked}$ is the number of search result with the i-th word and unblocked categories, and $freq_i$ is the frequency of the i-th word. If the calculated block score is larger than 0, sunshine determines that this web page should be blocked.

Page count based similarity extraction algorithm showed a reasonable result, but not a perfect block mechanism, so it may malfunction. Hence, we also implemented a whitelist that Sunshine would not perform semantic analysis in websites in it. Users can freely add websites in the whitelist if the semantic analysis malfunctions in the website that they are using for studying or working.

## D. *Dedicated Launcher for Self Management*

Launchers in Android are the part of user interface that lets users change the home screen for customizing smartphone or tablet experience [18]. Therefore, SUNSHINE customize SUNSHINE launcher suited to enhance ability of self-management by modifying open source launcher [19].

### 1) *Simplifying user interface of launcher*

SUNSHINE changes user interface of launcher intuitive and simple to increase user concentration. First of all, developers divide apps, web pages, functions of Android to disturbing components and helping components for productivity using SUNSHINE. Then, SUNSHINE eliminates or limits disturbing components from launcher.

**Disturbing component**: back key and history key of launcher are blocked. Disturbing apps and web pages determined by policy of SUNSHINE categorizing are prohibited by default, allowed only under limited condition. Widget and other unnecessary settings are eliminated from launcher. All notifications are delayed and transmitted to user after SUNSHINE mode turning off.

**Helping component**: SUNSHINE web browser, allowed app list determined by SUNSHINE app categorizing are used freely in SUNSHINE launcher.

### 2) *Applying gamification policy*

SUNSHINE aims to motivate users to do their work by gamified launcher. It sets policy for gamification to launcher. For score design, *activity types* are user's self-management [10] and *Quality standardization* is time remained.

*Score* is point. User can acquire the points by remaining the sunshine mode. SUNSHINE sets two *goal*. One is leveling system and the other is rewarding system. Leveling system makes user level up using cumulated points and rewarding system is a kind of quest in game. User can save their point considerably. Then cumulated points are used for using blocked apps in certain time as reward. These two goals give users a motivation to intended behavior and stay self-managing.

In prototype version, developer consider easy to test variation of user states. Quality standardization is 10points/10sec. Level up is 1 level / 20 points and using blocked apps is 3secs / 1point. SUNSHINE also provides system functions which change these gamification components to change the policy in future works.

### 3) *SUNSHINE launcher menu*

When user turn on SUNSHINE icon from the quick bar, dedicated launcher started and three main menus are displayed.

#### a) *User state*

Home screen of dedicated launcher displays user state. User state consist of level, point, timer, total cumulated time, unlock button in Fig.10. Screen of user state corresponds to game state of character. Therefore, user can regard him/herself as game character and take an interest in using SUNSHINE. When SUNSHINE mode on, timer and point are reset and launcher requests to PAService to load user level and cumulated total time from system by system function. When user turns off SUNSHINE, launcher requests PAService to save user states to system by system function. If timer times up after 10 seconds, 10 points are increased and UI updated. Points are used to use blocked apps or level up.

#### b) *Allowed app list*

Launcher received blocked app list from system when SUNSHINE mode on. Then an allowed apps list is displayed in Allowed App List menu as in Fig. 9. In this menu, every apps can be always used freely. SUNSHINE web browser exists in this menu.

#### c) *Blocked app list*

As you can see in Fig. 10, blocked apps are displayed in blocked apps list menu. This menu is only activated when points are above 20 points then lock button is changed to unlock button and blocked app list can be activated by clicking unlock button. Permission about blocked apps is allowed only

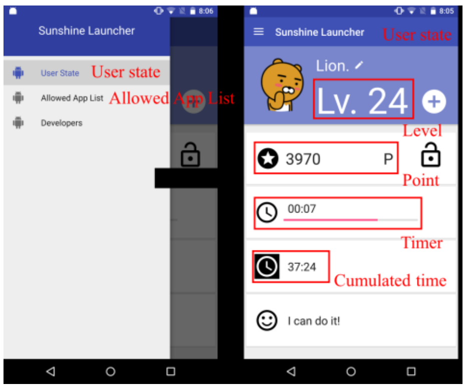


Fig. 9. Launcher menu & user state

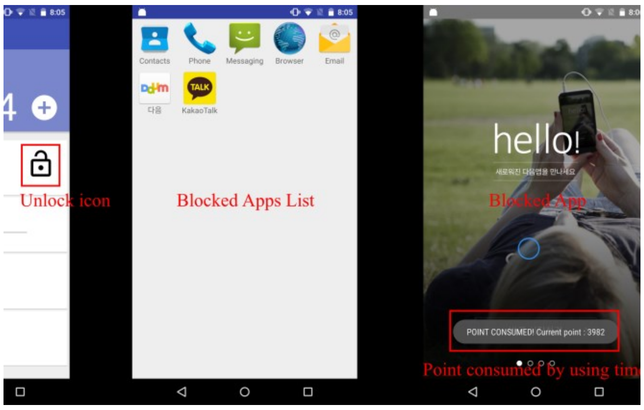


Fig. 10. Blocked apps list & running blocked apps

when blocked apps list activity is started. When activity is in onCreate state, request permission is enabled for blocked apps. In onDestroy state, request permission is disabled for blocked apps to system. While using blocked apps, SUNSHINE consumes points and notifies it to user. If no more point remains, SUNSHINE automatically quits the running apps

## V. Conclusion

### A. *Contributions*

SUNSHINE has four main contributions. First of all, SUNSHINE provides smart system which classify disturbing contents and helping contents for productivity. It performs classification by semantic analysis in web browser and app categorizing. It displays helping contents and blocked disturbing contents or allowed disturbing contents under limited environment. Secondly, other apps for self-management existing before SUNSHINE are incomplete to control apps. On the other hand, controlling other apps and functions in SUNSHINE could be performed in system layer, it can operate more stable. Thirdly, SUNSHINE provides not only control system but also motivation system. It uses gamified components so users would feel more fun and aware of goals, then make a more effort. Lastly, smartphone targeted to kids or test-taker who need controlled environment make good use of SUNSHINE.

### B. *Limitations*

First, semantic analysis web browser still needs higher accuracy. Even if not blocked contents, it can be blocked. Second, smartphone needs to change system to install SUNSHINE. SUNSHINE could not be downloaded by app store and would be embarked by a smartphone company before launch. Lastly, categorizing standards is static. Although disturbing components and helping components could depend on situations, SUNSHINE has no idea to know a context. For example, game contents would be blocked when researchers study about performance of game.

### C. *Future Works*

Qualitative investigation for semantic analysis would be performed to enhance accuracy of web browser. Additionally, categorization would be decided by users and users can set standards for productivity more dynamically. Furthermore, domain of gamification components would be extended to social networks. Then, users can compete with their friends. It motivates user to work hard.

## References

[1] M. Sushma, D. Peter, G. Natalya, L. Gregory, and C. Donald, “The Impact of Internet Addiction on University Students and Its Effect on Subsequent Academic Success: A Survery based Study,”Issues in Information Systems, vol. 15, no. 1, p344-352, Jan, 2014

[2] B. T. Sampath Kumar and G. Manjunath, "Internet use and its impact on the academic performance of university teachers and researchers," Higher Education, Skills and Work-Based Learning, vol. 3, no. 3, pp. 219–238, Sep. 2013.

[3] A. Lepp, J. E. Barkley, and A. C. Karpinski, "The relationship between cell phone use, academic performance, anxiety, and satisfaction with life in college students," Computers in Human Behavior, vol. 31, pp. 343–350, Jan. 2014.

[4] T. Education, "Study helper – Android Apps on Google play," 2016. [Online]. Available: https://play.google.com/store/apps/details?id=kr.co.tamseng.StudyHelper. Accessed: Jun. 19, 2016.

[5] S. J. Tools, "Self control for study – Android Apps on Google play," 2016. [Online]. Available: https://play.google.com/store/apps/details?id=com.specialj.selfcontrol. Accessed: Jun. 19, 2016.

[6] Q. Li, "What is Semantic Analysis?," 2013. [Online]. Available: http://www.slideshare.net/qiuyuel2/what-is-semantic-analysis. Accessed: Jun. 19, 2016.

[7] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis,"Discourse Processes, vol. 25, no. 2-3, pp. 259–284, Jan. 1998.

[8] S. Simmons and Z. Estest, “Using latent semantic analysis to estimate similarity,” 2006.

[9] S. A and S. S., "Measuring semantic similarity between words using web documents,"International Journal of Advanced Computer Science and Applications, vol. 1, no. 4, 2010.

[10] K. Huotari and J. Hamari, "‘Gamification’ from the perspective of service marketing," Proc. CHI’ 11 Workshop Gamification , Apr. 2011.

[11] S. Scheider, M. Raubal, P. Kiefer, C. Sailer, and P. Weiser, "Score design for meaningful gamification," CHI’15 Gamifying Research: Strategies, Opportunities, Challenges and Ethics, Apr. 2015.

[12] "Android 4.4 KitKat NotificationManagerService," 2014. [Online]. Available: http://blog.csdn.net/yihongyuelan/article/details/41084165. Accessed: Jun. 18, 2016.

[13] "Android 4.4 (KitKat) window management subsystem framework," 2014. [Online]. Available: http://prog3.com/sbdm/blog/jinzhuojun/article/details/37737439. Accessed: Jun. 18, 2016.

[14] "Android - WindowManagerService, Activity ( Starting Window ) ," 2013. [Online]. Available: http://blog.csdn.net/luoshengyang/article/details/8577789. Accessed: Jun. 18, 2016.

[15] Mozilla, "Readability," GitHub, 2016. [Online]. Available: https://github.com/mozilla/readability. Accessed: Jun. 19, 2016.

[16] Luin, "Readability," GitHub, 2016. [Online]. Available: https://github.com/luin/readability. Accessed: Jun. 19, 2016.

[17] "Wordpos," GitHub, 2016. [Online]. Available: https://github.com/moos/wordpos. Accessed: Jun. 19, 2016.

[18] A. MARTONIK, "The best Android launchers," in androidcentral, 2016. [Online]. Available: http://www.androidcentral.com/best-android-launchers. Accessed: Jun. 18, 2016.

[19] Arnabc, "Simplelauncher," GitHub, 2016. [Online]. Available: https://github.com/arnabc/simplelauncher. Accessed: Jun. 18, 2016.