**Well-Being App – Methodological Specifications for Ping Strategy**

User defined parameters:

1. Number of notifications (i.e., “pings”) to be received each day

* Response required (i.e., hard edit)

*num\_pings* = 2, 3, 4 or 5 (default is 5)

1. Time when sleeping

* Respondent options (asks for times when can be pinged):
* Daytime only – 8am to 6pm;
* Daytime and evenings – 8am to 10pm;
* Evenings only – 6pm to 10pm
* Should be displayed in 24-hour format for French and 12-hour format for English
* For programming purposes, everything should be defined in terms of 24-hour clock
* Must be consecutive hours (if not a whole number, will need to update behind the scenes; e.g., 11:30pm – 7:30am will need to be changed to 11pm to 8am). Suggest round down for time when they go to bed and round up for time when they wake up (to minimize disturbing respondents during sleep).
* This is the time when respondent is NOT to receive notifications
* Maximum sleep interval: 14 hours
* Minimum sleep interval: 2 hours

*sleep\_hour* = time when respondent goes to bed (default is 22h or 10pm)

*awake\_hour* = time when respondent wakes up (default is 6h or 6am)

*awake\_interval* = number of hours respondent is awake = *sleep\_hour* - *awake\_hour* (default is 16 hours)

***Note: if awake\_hour>sleep\_hour then add 24 to awake\_interval***

Ping intervals (i.e., one hour increments when pings can occur):

* There will be *awake\_interval* potential one hour increments where pings can occur **(maximum of 22 and minimum of 10; based on defaults)**
* Condition: If *num\_pings*>*awake\_interval* then set *num\_pings = awake\_interval* (this shouldn’t happen if we are setting a maximum sleep time as 12 hours but adding this to be safe).

Example (for the defaults):

|  |  |
| --- | --- |
| **Interval #** | **One Hour Time Intervals (Default)** |
| 1 | 6h to 7h |
| 2 | 7h to 8h |
| 3 | 8h to 9h |
| 4 | 9h to 10h |
| 5 | 10h to 11h |
| 6 | 11 h to 12h |
| 7 | 12h to 13h |
| 8 | 13h to 14h |
| 9 | 14h to 15h |
| 10 | 15h to 16h |
| 11 | 16h to 17h |
| 12 | 17h to 18h |
| 13 | 18h to 19h |
| 14 | 19h to 20h |
| 15 | 20h to 21h |
| 16 | 21h to 22h |

Generic (for algorithm):

|  |  |
| --- | --- |
| **Interval #** | **One Hour Time Intervals** |
| 1 | *awake\_hour* to *awake\_hour+1* |
| 2 | *awake\_hour+1* to *awake\_hour+2* |
| … | … |
| **i** | ***awake\_hour+(i-1)* to *awake\_hour+i*** |
| … | … |
| *awake\_interval – 1* | *awake\_hour+(awake\_interval-2)* to *awake\_hour+awake\_interval-1)* |
| *awake\_interval* | *awake\_hour+(awake\_interval-1)* to *awake\_hour + awake\_interval* |

Ping Strategy:

* Restriction: need to allow a minimum of 1 hour between pings
* *num\_pings* intervals of one hour will be selected based on following probabilities and respondent will receive *num\_pings* pings throughout the day at random times in selected interval (while respecting the one hour limit restriction)

Selection Probabilities:

* Simple strategy to avoid hundreds of distributions for all different potential sleep schedules
* Probabilities of selection are based on GSS29 data and frequency of respondents attending cultural events

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **One Hour Time Intervals** |  |  | **Probability of Selection** | |
|  | **Weekdays** |  | **Weekends** |  |
| 0h to 1h |  |  |  |  |
| 1h to 2h |  |  |  |  |
| 2h to 3h |  |  |  |  |
| 3h to 4h |  |  |  |  |
| 4h to 5h |  |  |  |  |
| 5h to 6h |  |  |  |  |
| 6h to 7h |  |  |  |  |
| 7h to 8h |  |  |  |  |
| 8h to 9h |  |  |  |  |
| 9h to 10h |  |  |  |  |
| 10h to 11h | A | **4** | N | **4** |
| 11h to 12h | B | **5** | O | **5** |
| 12h to 13h | C | **5** | P | **8** |
| 13h to 14h | D | **5** | Q | **10** |
| 14h to 15h | E | **5** | R | **11** |
| 15h to 16h | F | **5** | S | **8** |
| 16h to 17h | G | **4** | T | **7** |
| 17h to 18h | H | **5** | U | **5** |
| 18h to 19h | I | **10** | V | **7** |
| 19h to 20h | J | **13** | W | **9** |
| 20h to 21h | K | **15** | X | **9** |
| 21h to 22h | L | **12** | Y | **7** |
| 22h to 23h | M | **6** | Z | **4** |
| 23h to 24h |  |  |  |  |

Where A=4%, B=5%, C=5%, D=5%, E=5%, F=5%, G=4%, H=5%, I=10%, J=13%; K=15%; L=12%; M=6%; and N=4%; O=5%; P=8% Q=10%; R=11%; S=8%; T=7%; U=5%; V=7%; W=9%; X=9%; Y=7%; Z=4%

Note that any of A through J could =0 if respondent is sleeping during that time.

For weekdays: split remaining (100-A-B-C-D-E-F-G-H-I-J-K-L) % = 6% equally among all non-sleeping time intervals.

For weekends: split remaining (100-M-N-O-P-Q-R-S-T-U-V-W) %=6% equally among all non-sleeping time intervals.

**If respondent will be sleeping between any of 10h and 22h then redistribute excess distribution equally among non-sleeping time intervals.**

**Also, we want to ensure that “prime” time intervals won’t have lower probabilities than the “non-prime” time intervals so we need an extra condition (i.e., a second re-distribution):**

* **For example, if probabilities in blue/red boxes end up being more than one of the non-zero prime time probabilities (i.e., 4%), set them equal to the minimum (non-zero) prime time probability and redistribute that excess proportionally among “prime” time intervals (i.e., all intervals between 10h and 22h).**

Example 1: Respondent sleeps between 23h and 6h

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
| 0h to 1h | 0% | 0% |
| 1h to 2h | 0% | 0% |
| 2h to 3h | 0% | 0% |
| 3h to 4h | 0% | 0% |
| 4h to 5h | 0% | 0% |
| 5h to 6h | 0% | 0% |
| 6h to 7h | 6/5=1.2% | 6/5=1.2% |
| 7h to 8h | 6/5=1.2% | 6/5=1.2% |
| 8h to 9h | 6/5=1.2% | 6/5=1.2% |
| 9h to 10h | 6/5=1.2% | 6/5=1.2% |
| 10h to 11h | 4% | 4% |
| 11h to 12h | 5% | 5% |
| 12h to 13h | 5% | 8% |
| 13h to 14h | 5% | 10% |
| 14h to 15h | 5% | 11% |
| 15h to 16h | 5% | 8% |
| 16h to 17h | 4% | 7% |
| 17h to 18h | 5% | 5% |
| 18h to 19h | 10% | 7% |
| 19h to 20h | 13% | 9% |
| 20h to 21h | 15% | 9% |
| 21h to 22h | 12% | 7% |
| 22h to 23h | 6% | 4% |
| 23h to 24h | 6/5=1.2% | 6/5=1.2% |
| **Total** | **100%** | **100%** |

Set times when respondent is sleeping to 0, fix the probabilities for times between 10am and 11pm then evenly distribute remaining 6% among 5 non-sleeping intervals.

Example 2: Respondent sleeps between 21h and 4h

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
| 0h to 1h | 0% | 17/7=2.43% |
| 1h to 2h | 0% | 17/7=2.43% |
| 2h to 3h | 0% | 17/7=2.43% |
| 3h to 4h | 0% | 17/7=2.43% |
| 4h to 5h | 24/7=3.43% | 17/7=2.43% |
| 5h to 6h | 24/7=3.43% | 17/7=2.43% |
| 6h to 7h | 24/7=3.43% | 17/7=2.43% |
| 7h to 8h | 24/7=3.43% | 17/7=2.43% |
| 8h to 9h | 24/7=3.43% | 17/7=2.43% |
| 9h to 10h | 24/7=3.43% | 17/7=2.43% |
| 10h to 11h | 4% | 4% |
| 11h to 12h | 5% | 5% |
| 12h to 13h | 5% | 8% |
| 13h to 14h | 5% | 10% |
| 14h to 15h | 5% | 11% |
| 15h to 16h | 5% | 8% |
| 16h to 17h | 4% | 7% |
| 17h to 18h | 5% | 5% |
| 18h to 19h | 10% | 7% |
| 19h to 20h | 13% | 9% |
| 20h to 21h | 15% | 9% |
| 21h to 22h | 0% | 0% |
| 22h to 23h | 0% | 0% |
| 23h to 24h | 24/7=3.43% | 17/7=2.43% |
| **Total** | **100%** | **100%** |

Set times when respondent is sleeping to 0, fix the probabilities for times between 10am and 11pm (except for two intervals where respondent is sleeping), then evenly distribute the remaining 24% among the 7 non-sleeping intervals for weekdays and the remaining 17% among the 7 non-sleeping intervals for weekends. Note that in this case, the redistribution is different for weekends versus weekdays. In the previous example, they happened to be the same.

Example 3: Respondent sleeps between 20h and 8h (requires a second redistributing since want to ensure that the prime times will be pinged more often)

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
| 0h to 1h | 0% | 0% |
| 1h to 2h | 0% | 0% |
| 2h to 3h | 0% | 0% |
| 3h to 4h | 0% | 0% |
| 4h to 5h | 0% | 0% |
| 5h to 6h | 0% | 0% |
| 6h to 7h | 0% | 0% |
| 7h to 8h | 0% | 0% |
| 8h to 9h | ~~19%~~ 19%-15%=4% | ~~13%~~ 13%-9%=4% |
| 9h to 10h | ~~19%~~ 19%-15%=4% | ~~13%~~  13%-9%=4% |
| 10h to 11h | 5%+30(5/62)%=7.42% | 4%+18(4/74)%=4.97% |
| 11h to 12h | 5%+30(5/62)%=7.42% | 5%+18(5/74)%=6.22% |
| 12h to 13h | 5%+30(5/62)%=7.42% | 8%+18(8/74)%=9.95% |
| 13h to 14h | 5%+30(5/62)%=7.42% | 10%+18(10/74)%=12.43% |
| 14h to 15h | 5%+30(5/62)%=7.42% | 11%+18(11/74)%=13.68% |
| 15h to 16h | 5%+30(5/62)%=7.42% | 8%+18(8/74)%=9.95% |
| 16h to 17h | 4%+30(4/62)%=5.94% | 7%+18(7/74)%=8.70% |
| 17h to 18h | 5%+30(5/62)%=7.42% | 5%+18(5/74)%=6.22% |
| 18h to 19h | 10%+30(10/62)%=14.84% | 7%+18(7/74)%=8.70% |
| 19h to 20h | 13%+30(13/62)%=19.29% | 9%+18(9/74)%=11.19% |
| 20h to 21h | 0% | 0% |
| 21h to 22h | 0% | 0% |
| 22h to 23h | 0% | 0% |
| 23h to 24h | 0% | 0% |
| **Total** | **100%** | **100%** |

Set times when respondent is sleeping to 0, fix the probabilities for times between 10am and 11pm (except for three intervals where respondent is sleeping), evenly distribute the remaining % (39% for weekdays and 26% for weekends) among the 12 non-sleeping intervals for weekdays but since it will make the probabilities too high set to the minimum non-zero prime time (4%), and then re-distribute remaining % (30% for weekdays and 18% for weekends) proportionally among the available prime-time intervals to ensure that the prime-times will be pinged more often than non-prime times.

Generic Algorithm:

Step 1 – Initialize based on respondents sleep schedule

Set probabilities to 0 for intervals where respondent is sleeping

*num\_prime\_int = # prime-time intervals where respondent is awake* (0, 1, …, 13)

*num\_nonprime\_int = # nonprime-time intervals where respondent is awake*

*=awake\_interval-num\_prime\_int*

If *awake\_hour>sleep\_hour* then for i=*sleep\_hour* to (*awake\_hour-1),* set Pi=0

If *awake\_hour<=sleep\_hour* then for i=*sleep\_hour* to 24 and for i=0 to *awake\_hour-1,* set Pi=0

Step 2 – Set Probabilities

For prime times:

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
|  |  |  |
| 10h to 11h | P10=A | P10=N |
| 11h to 12h | P11=B | P11=O |
| 12h to 13h | P12=C | P12=P |
| 13h to 14h | P13=D | P13=Q |
| 14h to 15h | P14=E | P14=R |
| 15h to 16h | P15=F | P15=S |
| 16h to 17h | P16=G | P16=T |
| 17h to 18h | P17=H | P17=U |
| 18h to 19h | P18=I | P18=V |
| 19h to 20h | P19=J | P19=W |
| 20h to 21h | P20=K | P20=X |
| 21h to 22h | P21=L | P21=Y |
| 22h to 23h | P22=M | P22=Z |
|  |  |  |
|  |  |  |

\*Note: Probabilities should be 0 if respondent is sleeping at that time.

Else, for all other time intervals (not sleeping and nonprime-time), set Pi based on:

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
| ***awake\_hour+(i-1)* to *awake\_hour+i*** | Pi=(100-**X**)/(*awake\_interval-num\_prime\_int)* | Pi=(100-**Y**)/(*awake\_interval-num\_prime\_int)* |

Where **X**=A+B+C+D+E+F+G+H+I+J+K+L+M and **Y**=N+O+P+Q+R+S+T+U+V+W+X+Y+Z

Step 3 – Need to redistribute if Pi for any prime-time interval have smaller probability than non-prime time intervals (excluding 0)

In other words, for non-prime intervals,

If Pi>min(P10, …, P22 excluding 0) then update probabilities as follows:

Let Pi\_new = min(P10, …, P22 excluding 0) = **M**;

Pi\* = Pi – **M**;

sum to allocate =

|  |  |  |
| --- | --- | --- |
| **One Hour Time Intervals** | **Probability of Selection** | |
|  | **Weekdays** | **Weekends** |
| ***awake\_hour+(i-1)* to *awake\_hour+i*** | Pi\_new=**M** | Pi\_new=**M** |
| 10h to 11h | P10\_new=P10 + ((P10/**X**)\**sum to allocate)* | P10\_new=P10 + ((P10/**Y**)\**sum to allocate)* |
| 11h to 12h | P11\_new=P11 + ((P11/**X**)\**sum to allocate)* | P11\_new=P11 + ((P11/**Y**)\**sum to allocate)* |
| 12h to 13h | P12\_new=P12 + ((P12/**X**)\**sum to allocate)* | P12\_new=P12 + ((P12/**Y**)\**sum to allocate)* |
| 13h to 14h | P13\_new=P13 + ((P13/**X**)\**sum to allocate)* | P13\_new=P13 + ((P13/**Y**)\**sum to allocate)* |
| 14h to 15h | P14\_new=P14 + ((P14/**X**)\**sum to allocate)* | P14\_new=P14 + ((P14/**Y**)\**sum to allocate)* |
| 15h to 16h | P15\_new=P15 + ((P15/**X**)\**sum to allocate)* | P15\_new=P15 + ((P15/**Y**)\**sum to allocate)* |
| 16h to 17h | P16\_new=P16 + ((P16/**X**)\**sum to allocate)* | P16\_new=P16 + ((P16/**Y**)\**sum to allocate)* |
| 17h to 18h | P17\_new=P17 + ((P17/**X**)\**sum to allocate)* | P17\_new=P17 + ((P17/**Y**)\**sum to allocate)* |
| 18h to 19h | P18\_new=P18 + ((P18/**X**)\**sum to allocate)* | P18\_new=P18 + ((P18/**Y**)\**sum to allocate)* |
| 19h to 20h | P19\_new=P19 + ((P19/**X**)\**sum to allocate)* | P19\_new=P19 + ((P19/**Y**)\**sum to allocate)* |
| 20h to 21h | P20\_new=P20 + ((P20/**X**)\**sum to allocate)* | P20\_new=P20 + ((P20/**Y**)\**sum to allocate)* |
| 21h to 22h | P21\_new=P21 + ((P21/**X**)\**sum to allocate)* | P21\_new=P21 + ((P21/**Y**)\**sum to allocate)* |
| 22h to 23h | P22\_new=P22 + ((P22/**X**)\**sum to allocate)* | P22\_new=P22 + ((P22/**Y**)\**sum to allocate)* |

Additional notifications that are required:

* After the one month study period is complete, a notification should be sent thanking the respondent for their participation in the study (subject matter/CPRD/communications to determine the exact wording of this message). Can appear immediately after respondent completes the final questionnaire.

After the respondent has completed 20 responses, a notification should appear asking them to rate the app in the app store. Exact wording to be determined by subject matter/CPRD/communications. Note to programmers: Feel free to adapt if there is a simpler way to program this! And feel free to contact Amanda Halladay or Caroline Pelletier to further discuss.

Document from github requirement ticket:

* + Pinging Strategy can be the same for both probabilistic and non-probabilistic pilots
  + If the respondent does not agree to notifications, there should be a pop up explaining the requirement of notifications/they will not be eligible for this survey if they do not turn on notifications
  + Respondent agrees to receive notifications
  + Respondent is then asked how many pings they agree to receive, and identifies their blackout period
    - Please chose the number of times we can ping each day during this study:
      * Two
      * Three
      * Four
      * Five
    - When do you normally sleep? We will ensure you are not pinged at this time.
      * Respondent enters hours
  + Respondent parameters remain the same unless they change their notification settings during the collection period
  + The respondent will always receive the number of notifications they have chosen, but the time will be randomized (ie, if they choose 5 notifications, they will receive 5 at random times each day they are in sample)
  + There must be a minimum of one hour between notifications
  + Respondent completes Questionnaire A
  + First ping comes immediately after questionnaire A is complete, requesting they complete questionnaire B
  + No action will be taken if the respondent does not complete Questionnaire B when pinged. If they ignore ping 1 and 2 but respond to ping 3, they should only have access to questionnaire B once.
  + We will require an output of times the respondent was pinged, so this can be aligned with response times by the respondent.
  + The respondent should receive a notification asking if they want to continue to receive notifications from the app after noticing no interaction after \_\_\_\_ (number of days - currently being defined by SM and methodology)
  + Pings will be randomized but with a higher probability of being sent during week nights and weekends to maximize chance of capturing cultural participation.
  + Methodology to work with IT to stratify time slices and select a random time each day with a higher probability in evenings and weekends. A simple example:

1. The Notifications Algorithm starts when the user finish the Survey A ( socio-demographic + wellbeingcheck) at handleSurveyAdone(){} in EqSurveyScreen
2. Notifications days: 4 days because we only allowed to send a reminders to the user for 4 days (based on Statistics Canada policy)
3. Even If the user browse to setting screen, the Notification algorithm can run only if the user has already finish the Survey A
4. At the end of the 4 days we send one notification saying (We haven’t heard from you in a while. Sign in for a Well-being Check!, Nous n’avons pas eu de vos nouvelles depuis un certain temps. Connectez-vous pour obtenir un Bilan bien-être!)
5. If the user does not come back to the wellbeingcheck survey (mean does not complete a survey B again, even if the user open the application and browse this doesn't count) We stop sending the user notifications and we do not schedule the notification for another 4 days.
6. The survey will be scheduled for 30 days from the day the user complete Survey A + 29 days.
7. If the user skip a day and doesn't complete a survey that day will be included in the 30 days.
8. The algorithm will be divided by intervals of 4 days. by default the user will not receive notification beyond the 4 days if he did not complete any survey b in that period (Well-being check).

The Notification Algorithm will start after the user finishes Survey A ( socio-demographic + wellbeingcheck) at handleSurveyAdone(){} in EqSurveyScreen. The settings screen cannot trigger the start of the notification algorithm.  
• Notifications are to be scheduled for 4 days at a time. If the user does not complete a survey B submission within 4 days, we will send one additional notification saying (We haven’t heard from you in a while. Sign in for a Well-being Check!, Nous n’avons pas eu de vos nouvelles depuis un certain temps. Connectez-vous pour obtenir un Bilan bien-être!).  
• If the user still does not complete a Survey B submission, we will stop sending the user notifications and not schedule the next round of 4 days. This respects a decision by OPMIC, SM and CPRD based on respondent burden practices and current collection reminder strategies.  
• The user can receive notifications for a maximum of 30 days from the day they complete Survey A    
   (survey A + 29 days).  
• If the user skips a day and doesn't complete a survey, that day will be included in the 30 days.

* The Notification Algorithm will start after the user finishes Survey A ( socio-demographic + wellbeingcheck) at handleSurveyAdone(){} in EqSurveyScreen. The settings screen cannot trigger the start of the notification algorithm.
* Notifications are to be scheduled for 4 days at a time. If the user does not complete a survey B submission within 4 days, we will send one additional notification on day 5 saying (We haven’t heard from you in a while. Sign in for a Well-being Check!, Nous n’avons pas eu de vos nouvelles depuis un certain temps. Connectez-vous pour obtenir un Bilan bien-être!).
* If the user still does not complete a Survey B submission, we will stop sending the user notifications and not schedule the next round of 4 days. This respects a decision by OPMIC, SM and CPRD based on respondent burden practices and current collection reminder strategies.
* The user can receive notifications for a maximum of 30 days from the day they complete Survey A  
  (survey A + 29 days).
* The Notification will be forecasted for 4 days. Always 4 days of Notification on the queue.
* If the user skips a day and doesn't complete a survey, that day will be included in the 30 days.

Following is an explain of the notification strategy:

We just think the default setting, 2 times notification per day.

1) After Survey A, the app will setup schedule:

Day 1:   x notification    the x=2-1 (the 1 time survey B in survey A) , the x could be 0 if there is not enough time to arrange a notification depends on current time

Day 2:   2 notifications

Day 3:   2 notifications

Day 4:   2 notifications

Next day: 1 last chance notification

2) During Day1 if the user changes the setting before any survey B done, The schedule list will be re-created.

3) Still in Day1, if the user changes the notification times in setting after one time surveyB done, for example change notification time from 2 to 4,

The old notification schedule list will be removed and a new notification list will be setup.

Day 1:   x notification   The x= 4- the count of passed notifications of day1 in old list, we only count the times in old notification list, there is no way to count that un-prompted surveyBs. the x could be 0 if there is not enough time to arrange a notification depends on current time

Day 2:   4 notifications

Day 3:   4 notifications

Day 4:   4 notifications

Next day:1 last chance notification

We continue using the default setting

4) During Day2, after first time survey B(no matter it is prompted or not), a new day schedule will be appended.

Day 2:   2 notifications

Day 3:   2 notifications

Day 4:   2 notifications

Day 5:   2 notifications

Next day:1 last chance notification  (the old last chance notification was removed, a new one was setup)

5) Still during Day2, after a survey B which is not first time of that day, nothing changes.

6) Starting from Day2, if user changes setting, the schedule will be immediately changed. for example change notification time from 2 to 4,

The old notification schedule list will be removed and a new notification list will be setup.

Day 2:   x notification   The x= 4- the count of passed notifications of day2 in old list, we only count the times in old notification list, there is no way to count that un-prompted surveyBs. the x could be 0 if there is not enough time to arrange a notification depends on current time

Day 3:   4 notifications

Day 4:   4 notifications

Day 5:   4 notifications

Next day:1 last chance notification

I agree with this logic.

7) Same things to day3, day4(reference to first list)

We are still back the default setting

Agreed.

8) During day 5(reference to first list)  , no matter the user received the last chance notification or not, as long as the user finish a survey B, a new list will be setup

Day 5:   x notification   The x= 2-1, the 1 is the survey B we just finished. the x could be 0 if there is not enough time to arrange a notification depends on current time

Day 6:   2 notifications

Day 7:   2 notifications

Day 8:   2 notifications

Next day:1 last chance notification

9)   In Day5 or after Day 5(all the scheduled notification are passed, the new schedule is not setup yet due to the user didn’t come back  yet), if the user changes setting, Nothing changes, because the scheduler is not triggered

 10) In Day5 or after Day 5(all the scheduled notification are passed, the new schedule is not setup yet due to the user didn’t come back  yet), if the user changes setting and later this user finishes a survey B, new 4 days notification schedule will be setup( actually it is mentioned in #9) along with a next day last chance notification.  The 4 is max days, it may be reduced depends on how many days left to the 30 days last day limitation.

About the unexpected EQ:

Actually we don’t have the concept of unprompted EQ, during a day,

1. User can do any times of survey during a day  
   2) After each of the survey, the scheduler will check if we need to append new schedule to that list
2. If needed(Normally after the first time of that day) , a new schedule list will be appended
3. If No, do nothing about the schedule
4. The app has no idea which EQ is associated to which schedule