Vol.2015-DPS-164 No.10
Vol.2015-EIP-69 No.10
2015/9/11

# モバイル端末の省電力制御を考慮したメッセージ配信遅延低減方式

大西健夫[†1]　城島貴弘[†1]

スマートフォンの普及に伴い，クラウドから端末上のアプリケーションに対してメッセージを配信するプッシュ配信の利用が拡大している．プッシュ配信はリアルタイム性の要求される用途にも利用されるため，メッセージの配信遅延はプッシュ配信の品質を示す重要な指標の 1 つとなる．メッセージ配信遅延を増加させる主な要因として，間歇受信と呼ばれるモバイル端末における消費電力制御があげられる．また，配信サーバが多数のメッセージを送信する場合は，処理遅延によってさらに配信遅延が増大することとなる．本稿では，メッセージ配信遅延が特に増大する多数のメッセージを送信する場合に，端末の間歇受信を考慮してメッセージ配信処理のスケジューリングを実施することで，メッセージ配信遅延を低減する方式を提案する．

# A Delay Reduction Method for Push Notifications Considering Power-saving Control of Mobile Terminal

TAKEO ONISHI[†1]　TAKAHIRO SHIROSHIMA[†1]

A push notification service delivers messages in real time from services on the cloud to applications on mobile terminals. Many application providers and users use the push notification service. In push notification, the latency is a key factor of the quality of service. A power saving control of mobile terminals, discontinuous reception, and a load of the push server increase the latency. In this paper, we propose a method to reduce the latency of push notification.

## 1. はじめに

近年，スマートフォンの普及に伴い，クラウド上のサーバからスマートフォンのアプリケーションに対して，任意のタイミングでメッセージを配信するプッシュ配信の利用が広まっている[1][2]．プッシュ配信により，新規に発生した情報をリアルタイムにアプリケーションやユーザに届けることで，様々なサービスを実現することが可能となっている．たとえば，プッシュ配信は，SNS などの更新通知やマルチメディア通信の着信，災害情報の通知などに活用されている．

プッシュ配信において，メッセージ配信の遅延時間はサービス品質を表す重要な指標の一つである．なぜならば，プッシュ配信の用途には，マルチメディア通信の着信や災害情報の通知など,リアルタイム性が求められる用途があり，低遅延でメッセージを配信する必要があるためである．

モバイル端末へのメッセージ配信遅延の発生原因を考えた場合，主に通信遅延とサーバ処理負荷の 2 つが考えられる．

まず，1 つ目の通信遅延に関しては，モバイル端末の省電力制御によって大きな影響を受ける．連続稼働時間が重視されるモバイル端末では，電力消費量を抑えるために，一定時間通信が発生しなかった場合は，無線モジュールを休止させる休止状態に遷移する制御を行っている.ただし，常時無線モジュールを休止していると，ネットワーク側から送信されたパケットを受信できないため，休止状態中の端末は一定周期で無線モジュールを短時間復帰させ自端末宛てのパケットの到着の有無を確認する，間歇受信(Discontinuous Reception : DRX)[3]と呼ばれる制御を実施している．間歇受信によって自端末宛てのパケットが確認された場合は，休止状態からアクティブ状態に復帰してパケットを受信する．一方で，自端末宛てのパケットが存在しない場合は，再度無線モジュールを休止させる．このように，休止状態中の端末は間歇受信のタイミングのみでしかサーバから送信されたメッセージを受信できないため，アクティブ状態の端末に比べて大幅に通信遅延が増大する．

次に，2 つ目の遅延発生原因のサーバ処理負荷に関しては，多数の端末に対して同時にメッセージを配信する際にサーバが高負荷になることによって発生する．災害情報の通知などは，配信端末が多数に渡ると考えられるため，サーバ処理負荷による遅延が発生しやすい．

以上で挙げた 2 つの遅延要因が重なった場合に，特にメッセージ配信の遅延が大きくなる．そこで我々は，モバイル端末の省電力制御(DRX)を考慮して，高負荷時のサーバにおけるメッセージ配信処理を適切にスケジューリングすることでメッセージ配信遅延を低減する方式を提案する．

## 2. 従来研究

本章では，プッシュ配信およびプッシュ配信におけるメッセージ配信遅延低減方式に関する従来研究について説明する．

†1 日本電気株式会社(株)
NEC Corporation.

Vol.2015-DPS-164 No.10
Vol.2015-EIP-69 No.10
2015/9/11

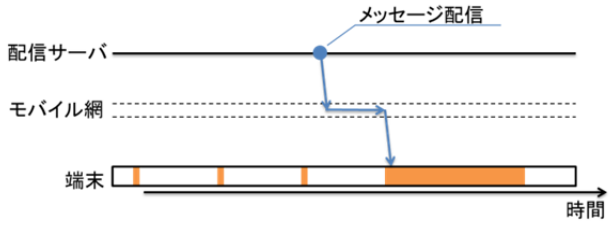


図 1　DRX とメッセージ受信タイミング

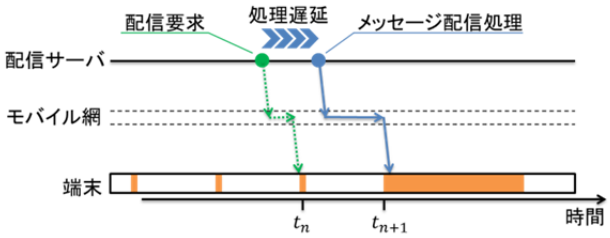


図 2　サーバの処理遅延により増加する
メッセージの配信遅延

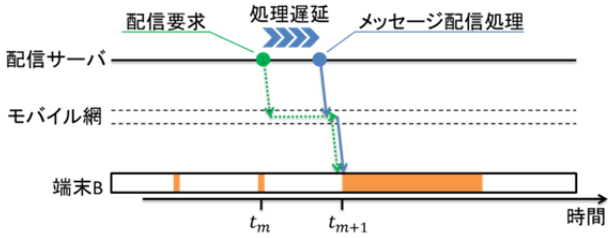


図 3　処理遅延が発生してもメッセージの到着時刻は変化しない場合の模式図

プッシュ配信は，現在，様々なプラットフォーム上で提供されている. 例えば, Android における GCM(Google Cloud Messaging)[1], iOS における APNS(Apple Push Notification Service)[2], FireFox OS における SimplePush[4]などがあり，また，W3C において PushAPI[5]の検討もなされている．これらを統合するプッシュ配信方式の研究[6]もなされており，プッシュ配信に関する需要の高さを示している．

A. Adya ら[7]は，配信サーバにおける CPU 負荷やメッセージ配信処理に要する時間などの評価を行っている．メッセージ配信処理の要因の分析を行い，システム内部の処理を見直すことで遅延低減が期待できるとしているが，モバイル網を介した通信による遅延を考慮したメッセージ配信遅延の評価は行っていない．

また，我々は休止状態の端末と，アクティブ状態の端末の通信遅延の差に着目し，アクティブ状態の端末に対するメッセージ配信処理を優先的に実施することで，配信遅延を低減する方式を提案してきた[8][9]．しかしながら，休止状態の端末間での配信処理の優先度付けは実施していなかった．そこで本稿では，休止状態中の端末が実施する DRX に着目し，休止状態の端末に対する配信処理間の優先度制御を実施することで, 配信遅延を低減する方式を提案する．

## 3. 提案方式

本章では，モバイル端末の省電力制御(DRX)を考慮して，高負荷時のサーバにおけるメッセージ配信処理をスケジューリングすることで，メッセージ配信遅延を低減する方式を提案する．まず，DRX による遅延とサーバ処理負荷による遅延が重なった場合のメッセージ配信遅延の特徴について考察を行い，それに基づき方式を提案する．

### 3.1 メッセージ配信遅延の発生要因

休止状態中の端末に対してメッセージを配信した時に発生する通信遅延の概念図を図 1 に示す. 図 1 の下部の四角は，端末の無線モジュールの状態を示しており，色のついている部分は無線モジュールが起動している状態，色のついていない部分は無線モジュールが休止している状態を示している．図の横軸は時間を示しており，図の左側において端末は休止状態となっている． DRX を実施しているため，周期的に短時間無線モジュールを起動させている．配信サーバが休止状態の端末にメッセージを配信すると，モバイル網内までメッセージが到着するが，端末は無線モジュールを休止させているため，端末にはメッセージが到着せずにモバイル網内に滞留する．その後，端末が DRX によってメッセージの到着を検知すると，休止状態から復帰しメッセージを受信する．このように，休止状態の端末は DRX のタイミングでしかメッセージを配信サーバから受信できないため，サーバがメッセージ配信を実施した時刻から見て，次の DRX のタイミングまでメッセージの到着が遅延する．

次に，サーバの処理遅延が加わった場合に発生するメッセージ配信遅延を図 2 に示す. 配信サーバにメッセージの配信要求が到着するが，メッセージ配信処理の実行は，処理遅延によって遅れている． もし，処理遅延がなければ配信要求を受けた時間から見て次の DRX の時間($t_n$)にメッセージが到着すると期待されるが，処理遅延によってメッセージ配信処理が実行された時刻は$t_n$を過ぎた時刻であるため，メッセージが端末に到着するのはさらに次の DRX の時間($t_{n+1}$)まで遅延している．このように，サーバの処理遅延が発生することで，さらにメッセージの到着遅延が増大することになる．

### 3.2 配信処理スケジューリング方式

前述の端末の DRX タイミングに配信処理が間に合わないために発生する遅延を低減するために， DRX までの残り時間が短い端末に対するメッセージ配信処理を優先する配信処理スケジューリング方式を提案する．提案方式のように配信処理をスケジューリングすることで，図 2 のよう

Vol.2015-DPS-164 No.10
Vol.2015-EIP-69 No.10
2015/9/11

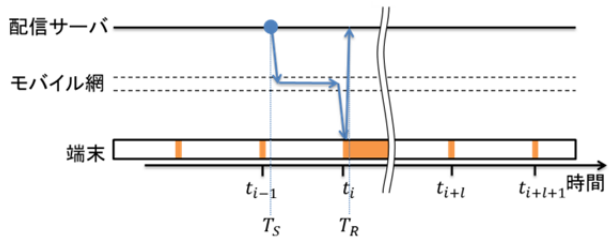


図 4　DRX タイミングの測定法

な状況の場合にメッセージ配信遅延の低減が期待される．すなわち，DRX までの残り時間が短い場合に，配信サーバが配信要求を受けた直後にメッセージ配信処理が実行されることで，メッセージが端末に到着する時間が$t_{n+1}$から$t_n$に早まる．一方で，DRX までの残り時間が長い端末に対するメッセージ配信処理は低優先とする．これは，DRX までの残り時間が長い場合，図 3 のように，端末に対するメッセージ配信処理を優先してもしなくても，メッセージが端末 B に到着するのは同じ時間($t_{m+1}$)になるためである．

ここで問題となるのは，配信サーバ側で端末の DRX の時間を推定する方法である．提案方式では，図 4 のように配信サーバから端末に対して DRX 推定用パケットを往復させることで推定する．

DRX は周期的に実行されるため，DRX の時間$t_i(i = 1,2,\dots)$は，

$$t_i = Li + C \tag{1}$$

で表される．$L$は DRX の周期，$C$は定数である．なお，$L$は通信事業者が設定する値であり，ここでは既知の値とする．従って，DRX の時間を推定するためには，$C$を算出すればよい(LTE における DRX の時間の決定方法は[10]を参照)．

ここで，配信サーバが時間$T_S$に DRX 推定用パケットを休止状態の端末に対して送信したとする．端末は DRX を実施しているため，DRX 推定用パケットが端末に到着するのは，時間$T_S$から見て，次の DRX のタイミング（ここでは$t_k$）まで遅延する．DRX 推定用パケットを受信した端末は即座に返信パケットを配信サーバに送信し，配信サーバは時間$T_R$に返信パケットを受信したとする．アクティブ状態の端末とサーバの RTT を用いて，RTT/2 によって端末からサーバへの通信遅延を近似すると，

$$T_R = t_k + \frac{\mathrm{RTT}}{2} \tag{2}$$

が成立する．式(1)から DRX の時間$t_i$の$L$の剰余は$C$であることから，

$$C = t_k \% L = \left(T_R - \frac{\mathrm{RTT}}{2}\right) \% L \tag{3}$$

により，$C$は算出される．端末は DRX 推定用パケットを受信した時にアクティブ状態に遷移するが，再度休止状態に遷移した場合は，再び式(1)に従い DRX を実施するため[10]，式(3)により算出した$C$を用いて，配信サーバにおいて端末の DRX の時間を推定することができる．

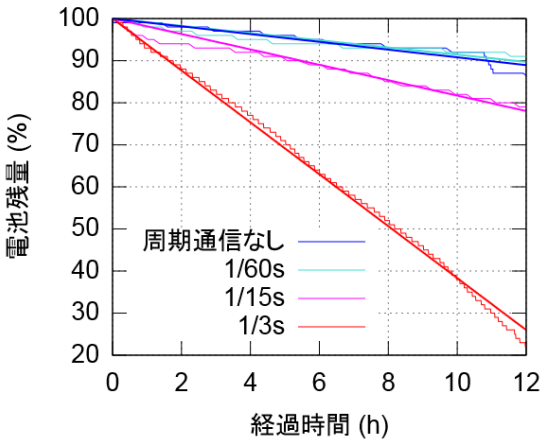


図 5 周期通信による電池消費量

端末の DRX の時間を決定するオフセット値$C$は，ハンドオーバなどの要因によって変化する場合があるため，配信サーバは定期的に DRX 推定用パケットを端末に送信して，オフセット値$C$の変化を監視する必要がある．DRX 推定用パケットの送信頻度を高くすると，オフセット値$C$の変化を早期に検知することができるため，端末の DRX 時間の推定誤差を減らすことができる．一方，頻繁に通信することになるため，端末の電池消費を増大させる副作用がある．このため，DRX 推定用パケットの送信頻度は端末の電池消費量への影響を考慮して，影響が少ない範囲でなるべく高頻度にすることが望ましい．

以上，まとめると，配信サーバは定期的に端末に対し DRX 推定用パケットを送信し，DRX の時間を推定するために必要なオフセット値$C$を算出する．配信サーバはメッセージを配信する時に，メッセージ配信対象となっている端末 M(M=1,2,3,…)の現在時刻から見て次の DRX までの残り時間($R_M$とする)を式(1)を用いて算出する．配信サーバは，メッセージ配信対象の端末の内，$R_M$が最小となる端末からメッセージ配信処理を実施する．

## 4. 評価

本章では，提案方式の評価について述べる．評価は 2 つの観点で実施した．1 つ目は，提案方式を実現する上で必要となる DRX タイミングの推定が端末の電池消費に与える影響の評価である．2 つ目は提案方式によるメッセージ配信遅延低減効果である．以下，詳細を述べる．

### 4.1 DRX タイミング推定による電池消費

前述したとおり，提案方式では定期的に配信サーバと端末間でパケットを往復させ，端末の DRX タイミングを推定するためのオフセット値$C$を算出する必要がある．頻繁

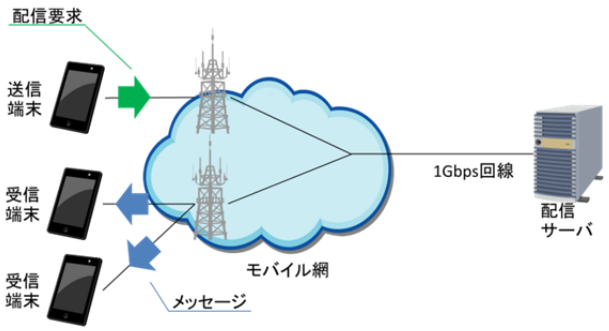


図 6　評価システムの構成

表 1　配信サーバの諸元

| 項目 | 値 |
|---|---|
| OS | Ubuntu 14.04.2 LTS |
| CPU | Core i7-2637 (1.70GHz) |
| メモリ | 8GB |
| ネットワーク | 1Gbps(有線 LAN) |

な通信は端末の電池消費に影響を与える可能性があるため，定期的にパケットを往復させることによる電池消費量への影響を評価する必要がある．

評価では，定期的に配信サーバから端末にパケットを往復させ，電池残量の時間変化を観測した．評価に用いた端末は Galaxy S5(Android 4.4.2)で，パケットの送信周期は，3 秒に 1 回，15 秒に 1 回，60 秒に 1 回とした．また，比較用に定期的なパケットを送信しない場合についても電池残量の時間変化を測定した．なお，各測定において，電池残量 100%の状態から計測を開始した．端末に対する操作は，電池消費量に影響するため，測定中は操作をしないようにした．また，画面のバックライトも電池消費量への影響が大きい事ことから，測定中は常時画面を消灯した状態とした．

図 5 に評価結果を示す．図中細いステップ状の線は観測された電池残量，太い直線は観測された電池残量を線形回帰した結果を表している．3 秒に 1 回の場合は電池消費量に与える影響は大きく，周期通信を発生させない場合に比べて 7 倍程度の速度で電池残量が減少することが分かる．一方で，60 秒に 1 回の場合は，周期通信を発生しない場合とほぼ同一の速度で電池残量が減少している．このことから，数分に 1 度程度のパケット往復であれば，ほとんど電池消費量に影響を与えることなく端末の DRX タイミングを推定することができることが分かる．

### 4.2 メッセージ配信遅延

次に，提案方式によるメッセージ配信遅延の低減効果について述べる．

#### 4.2.1 評価方法

評価システムの構成を図 6 に示す．配信サーバは，1Gbps の光回線でネットワークに接続されており，メッセージ配信要求の受信し，宛先となる端末に対するメッセージ配信処理を実施する．配信サーバの諸元は表 1 に記したとおりである．送信端末は，メッセージ配信要求を配信サーバに送信する．配信要求はランダムな時間間隔で送信する．受信端末は配信サーバからメッセージを受信し，メッセージの到達遅延を測定する．送信端末および受信端末は LTE 回線を経由して，配信サーバと通信する．送信端末は 1 台，受信端末は 2 台とした．端末は Galaxy S5(Android 4.4.2)を利用した．

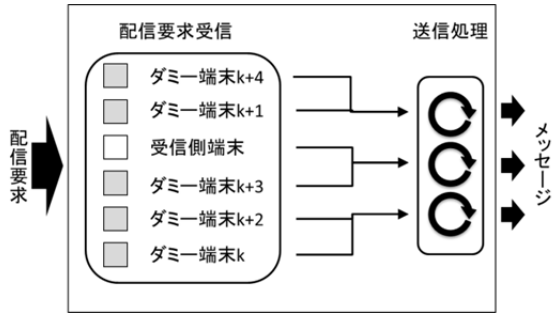


図 7　従来方式における配信サーバの処理

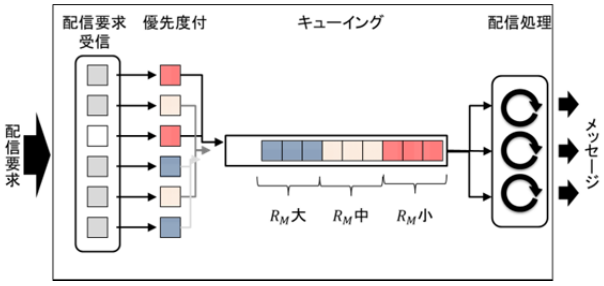


図 8　提案方式における配信サーバの処理

配信サーバにおける処理を図 7 と図 8 に示す．図 7 は従来方式(提案方式なし)の処理，図 8 は提案方式の処理を示している．配信サーバは送信側端末から配信要求を受信する．配信要求には配信メッセージと$N$個の宛先が記載されている．なお，本評価では，$N = 30000$とした．$N$端末分の受信端末を用意することができなかったため，$N$個の宛先の内，2 つは受信端末を示す宛先となっており，残りはダミーの宛先（ダミー端末宛て）となっている．配信サーバでは，メッセージの送信処理を行うスレッドが複数存在し，図 7 に示した従来方式の処理では，各スレッドが順次配信要求に記載された宛先へのメッセージ送信処理を実施していく．配信処理の負荷を評価するため，ダミー端末宛てのメッセージも送信するが，受信側端末に到着するのは，受信側端末宛てのメッセージのみである．

提案方式の場合は，配信要求を受け付けた後，各端末に対するメッセージ送信処理の優先度付を実施する．優先度は，DRX までの残り時間$R_M$(ただし，$M$=1,2,…,$N$)が小さい

もの程，高優先度とする．$R_M$を算出する時に必要となる式(1)の$C$については，受信側端末の$C$は推定用パケットを往復させた結果に基づき算出し，ダミー端末の$C$は乱数で生成した．その後，各端末に対する送信処理を優先度が高いものから先にキューに積み，送信処理スレッドはキューからFIFO で送信処理を実施していく．このようにすることで，優先度の高いと判定された端末に対する配信処理程，先に実行されることとなる．

表 2 に測定に用いた設定をまとめた．前述したとおり，受信端末数は 2 台，ダミー端末数は 29998 台で，配信サーバは計 30000 台に対するメッセージ配信処理を 1 度の配信要求で実施する．端末の DRX 周期$L$は実測値に基づき 1.28 秒とした．送信端末はメッセージ配信要求を約 20 秒間に 1 回，配信サーバに送信した．受信端末はメッセージを受信した時にアクティブ状態に遷移するが，約 20 秒間の間隔を取ることによって，次のメッセージ受信時は再度休止状態に遷移した状態となる．なお，メッセージ配信要求の送信間隔は，厳密に 20 秒とはせず，乱数を用いて，18.72 秒から 21.28 秒の間に均一に分布するようにした．これは，送信要求が受信端末の DRX のタイミングに対して均一に発生するように，つまり，$R_M$が 0～$L$の間に均一の確率で分布するようにするためである．

#### 4.2.2 評価結果

送信端末がメッセージ配信要求を送信してから受信端末がメッセージを受け取るまでの時間（配信遅延）の累積度数分布を図 9 に示す．図中，青の点線で示した線が従来方式の場合，赤の実線で表した線が提案方式の場合の配信遅延となっている．図を見て分かるように，提案方式の方が低遅延でメッセージを配信できていることが分かる．従来方式の配信遅延の平均値は 1.02 秒，提案方式の配信遅延の平均値は 0.56 秒となっており，約 45%の遅延低減効果が確認できた．

従来方式の配信遅延を見ると，約 0.5 秒のところで傾きが変化していることが分かる．これは送信端末が配信要求を要求してから，配信サーバにおいて最後に配信処理されたメッセージがモバイル網に到着するまでの時間に対応すると考えられる．配信遅延が最大となると考えられるのは，配信サーバにおいて最後に送信処理が実施され，かつ，モバイル網において DRX の周期$L$滞留する場合である．従来方式の配信遅延の最大値は約 1.8 秒であるため，$1.8 - L \approx 0.5$であり，計算と一致する．従来方式では，配信処理遅延によって，図 2 のように，次の DRX のタイミングに間に合わず，次々回の DRX のタイミングまでメッセージの到着が遅れている状況が発生しているため，配信遅延が$L$より大きな値となる場合が多くなり，今回の評価では約 30%を占めている．

一方で，提案方式は配信要求を送信後，1.28 秒(DRX 周期：$L$)後にはほとんど受信端末にメッセージが到着している．つまり，配信要求を送信した時間から見て次の DRX のタイミングで受信端末にメッセージが到着しており，意図したとおり優先制御により配信遅延が低減されていることが分かる．

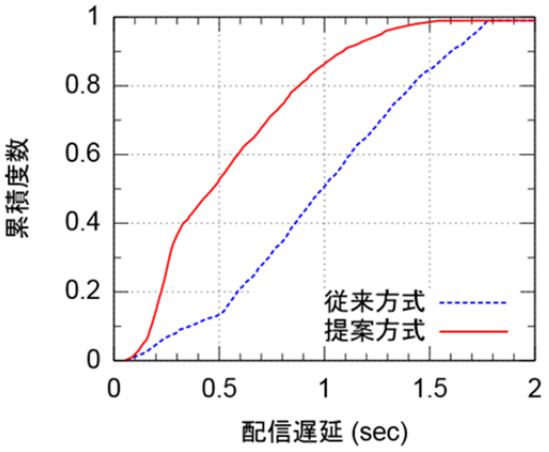


図 9 メッセージ配信遅延の評価結果

表 2 配信遅延の評価条件

| 項目 | 設定値 |
|---|---|
| 受信端末数 | 2 台 |
| ダミー端末数 | 29998 台 |
| DRX 周期($L$) | 1.28 秒 |
| 配信要求発生間隔 | 18.72～21.28 秒 |
| 配信要求数 | 約 1600 回 |
| DRX 推定パケット送信間隔 | 110 秒 |

## 5. おわりに

本稿では，モバイル端末の省電力制御(DRX)を考慮して，高負荷時のサーバにおけるメッセージ配信処理をスケジューリングすることでメッセージ配信遅延を低減する方式を提案した．休止状態中の端末は，配信サーバから送信されたメッセージを DRX のタイミングでしか受信できない．配信サーバが高負荷状態にある場合，メッセージ送信処理が次回の DRX タイミングに間に合わず，メッセージの到着が次々回の DRX タイミングまで遅延するという課題が発生する．提案方式では，DRX タイミングまでの残り時間が短い端末に対する送信処理を優先することで，前述の遅延発生を低減する．評価の結果，約 45%の配信遅延低減効果を確認することができた．今後の課題としては，メッセージを受信する端末が移動する場合の評価や，メッセージ配信数を変更した場合の評価などがあげられる．

## 参考文献

1) Google: Google Cloud Messaging for Android, Android Developers (online), available from <https://developers.google.com/cloud-

messaging> (accessed 2015-7-31).
2) Apple: Local and Push Notification Programming Guide: Apple Push Notification Service, iOS Developer Library(online), available form < https://developer.apple.com/library/ios/documentation/NetworkingInternet/Conceptual/RemoteNotificationsPG/Chapters/ApplePushService.html > (accessed 2015-7-31).
3) 3GPP TS 36.321, Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) protocol specification, V8.0.0(2007).
4) Mozilla: WebAPI/SimplePush, MozillaWiki(online), available from <https://wiki.mozilla.org/WebAPI/SimplePush> (accessed 2015-8-3).
5) Sullivan, B. and Fullea, E.: Push API, W3C Working Draft(online), available from < http://www.w3.org/TR/push-api/ > (access 2015-8-3).
6) Lee, Y., Oh, J. and Lee, B.G. : Logical push framework for real-time SNS processing, *Proc. 2012 Fourth International Conference on Computational Aspects of Social Networks* (*CASoN*), pp.47-51, IEEE(2012).
7) Adya, A., Cooper, G. Myers, D, et al.: Thialfi: A Client Notification Service for Internet-Scale Applications, *Proc. 23rd ACM Symposium on Operating Systems Principles* (*SOSP*), pp.129-142, ACM(2011).
8) 大西健夫, 城島貴弘, 中島一彰: サーバプッシュにおけるモバイル端末の RRC 状態を考慮したメッセージ配信遅延抑制方式, 情報処理学会研究報告, 2013-DPS-154(39), pp.1-6, 2013.
9) 城島貴弘, 大西健夫: モバイル端末における RRC 状態遷移に伴う通信遅延を考慮したメッセージング方式の提案, 信学技報, vol. 112, no. 463, NS2012-225, pp. 349-354, 2013.
10) 3GPP TS 36.304, Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment(UE) procedures in idle mode, V9.2.0(2010).