社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

Firefoxにおけるマルチバイト文字に起因するバグの調査

市川　直人†　　柗本　真佑†　　楠本　真二†

† 大阪大学大学院情報科学研究科
E-mail: †{n-itikaw,shinsuke,kusumoto}@ist.osaka-u.ac.jp

あらまし　ソフトウェア開発において，ソフトウェアに含まれるバグを理解することは重要である．本稿ではバグの一種として，マルチバイト文字に起因するバグに着目する．マルチバイトバグは，マルチバイト文字を扱うあらゆるソフトウェアで発生する共通問題であり，UI/UX 等のユーザの使用感への影響に限らず，様々な症状を引き起こす可能性がある．本研究では，マルチバイトバグについての理解を目的として，Firefox のバグ管理システム Bugzilla のバグ報告をもとに，マルチバイトバグを，報告内容/修正過程/原因と結果の 3 つの観点から調査した．報告内容では，修正期間と修正優先度について調査を行った．また，修正過程では，修正期間，放置期間，コメント数，修正パッチの提案回数を調査し，原因と結果では，個々のマルチバイトバグに対してコメントと修正パッチを目視調査し，その原因と結果を調査，分類した．

キーワード　マルチバイト文字，バグ報告，バグ管理システム，Bugzilla，Firefox

1. はじめに

ソフトウェアに含まれるバグ(注1)の理解を目的として，過去に発生したバグに関する分析，調査が多数行われている [1] [2] [3] [4]．具体的な調査の例としては，プログラムの並列動作に起因するコンカレンシーバグの分析 [1] や，非機能バグの中でも影響の大きいセキュリティバグとパフォーマンスバグの分析と比較 [2]，OSS を対象としたバグ全体の大規模な傾向の分析 [3] [4] などが挙げられる．また，バグ自体ではなくバグの修正活動に着目した調査も広く行われている [5] [6]．例えば，OSS におけるバグ報告内容の品質の分析 [5] や，同一バグに対する重複バグ報告の実態調査 [6] などである．

本稿ではバグの一種として，マルチバイト文字に起因するバグ（以降，MB バグ）に着目する．マルチバイト文字とは，1 バイトで 1 文字を表現する ASCII 等のシングルバイト文字とは異なり，複数バイトで 1 文字を表す文字，あるいは符号化体系を意味する．日本におけるマルチバイト文字としては，UTF-8 や Shift-JIS が広く用いられている．MB バグはこれらマルチバイト文字を起因として発生するバグであり，シングルバイト文字の利用のみを前提としたソフトウェア上で発生しやすい．MB バグの代表例としては文字化けが挙げられる．他にも MB バグの発生原因としては，エンコードの誤りや文字長の計算誤り，制御文字の不適切な処理など，その理由は多岐に渡る．

MB バグは，マルチバイト文字を扱うあらゆるソフトウェアで発生する共通問題であるといえる．近年では，特定のソフトウェアを多様な言語や文化に適合させる，ソフトウェアの国際化（internationalization，i18n）や地域化（localization，l10n）と呼ばれるプロセスが広く採用されるようになった．このプロセスにおいては，多種多様な言語への適応，すなわちマルチバイト文字の適切な処理が欠かせない．MB バグの性質の理解と，その傾向の把握は，この i18n や l10n などの共通プロセスの改善に繋がる可能性がある．さらに MB バグの発生原因や修正方法を具体的なソースコードの粒度で整理し体系化できれば，様々なソフトウェアで横断的に活用可能な一種のパターンを獲得できる可能性もある．

さらに，バグ自体が引き起こす影響という側面で考えると，MB バグは UI/UX 等のユーザの使用感への影響に限らず，様々な症状を引き起こす可能性がある．まずユーザの使用感は，ソフトウェアが提供する本質的な機能外の，いわゆる非機能に属する特性ではあるものの，ソフトウェアの品質を考える上では欠かせない特性である．加えて，MB バグはソフトウェアの機能面に悪影響を及ぼすこともある．具体的な事例としては，2018 年に発生した iOS の Unicode 処理に起因する「文字爆弾」の事例が挙げられる（CVE-2018-4124）．この事例では，iOS のアプリ上でテルグ文字(注2)を表示するだけで，そのアプリのクラッシュや OS の再起動ループが発生するため，Twitter を初めとする様々なメッセージングアプリで悪用された．

本稿では，MB バグの理解を目的として，Firefox プロジェクトに含まれる MB バグの定量的な分析を行う．調査では，修正済みのバグを母集団とし，MB バグとそれ以外の通常バグの 2 種類を抽出して比較を行う．比較を行った内容は，重複報告数，修正優先度，修正期間，放置期間，コメント数，修正パッチの提

(注1)：本稿ではソフトウェア内の欠陥や不具合の意味でバグという言葉を用いる．

(注2)：南インドの一部で利用されるテルグ語を表記する文字．

案回数である．また，31 件のバグ報告のコメントと修正パッチを目視確認し，個々の MB バグの原因と影響を読み解き，分類した．

2. 準　　備

2.1 文字コード

文字コードとは，文字をコンピュータ上で扱うために，文字とビット列を対応付ける規則である．文字コードは，1 文字を 1 バイトで符号化するシングルバイト文字と，1 文字を 2 バイト以上（可変長も含む）で符号化するマルチバイト文字に大別できる．シングルバイト文字は，1 文字を 1 バイトのビット列に対応させることから，高々 256 種類の文字しか扱うことができない．そのため，使用する文字の種類が少ない英語系の言語圏ではシングルバイト文字が多用される一方で，日本語，中国語，韓国語などの使用する文字の種類が多い言語圏では，マルチバイト文字が使われる事が多い．代表的なシングルバイト文字としては ASCII，ISO/IEC 646 があり，マルチバイト文字は UTF-8，UTF-16，Shift_JIS などが有名である [7]．

2.2 マルチバイトバグ

本稿では，マルチバイト文字に起因して発生するバグをマルチバイトバグ（MB バグ）と定義する．最も代表的な MB バグは文字化けである．現在使われている文字コードのほとんどは ASCII を基本にしており，ASCII には，アルファベット，数字，基本的な記号の符号化が定義されている．異なるマルチバイト文字はそれぞれ同じ文字であっても異なる符号化を行うが，アルファベットや数字に関しては，ASCII と互換性を持たせるために ASCII と同じ符号を割り当てる事が多い．そのため，意図しない文字コードでビット列を変換しても，ASCII に含まれる文字に関しては正しく変換されることが多いが，ひらがなや漢字といった ASCII に含まれない文字は意図しない文字として変換されてしまう．

また，シングルバイト文字で変換された文字列の文字数を数える場合，1 バイトを 1 文字とカウントすればよいが，このような方法では，マルチバイト文字で変換された文字列の文字数を正しく数えることができない．このように，マルチバイト文字は様々なバグを引き起こす．

2.3 Mozilla におけるバグ管理システム

bugzilla.mozilla.org(注3)（BMO）は Mozilla プロジェクトで用いられるバグ管理システムであり，Firefox を含む Mozilla のプロジェクトに関するバグ報告や機能要望を行う．図 1 に BMO のスクリーンショットを示す．BMO では，1 つのスレッドを 1 つのバグとして管理しており，各スレッドは，そのバグの概要を表すメタ情報と，その修正について議論するコメント部分の 2 つから構成される．スレッドの情報部分の要素のうち，本研究で扱った要素を表 1 に示す．

BMO において，報告された個々のバグは，図 2 に示すライフサイクルに従って遷移する．報告されたバグは未確認（UNCONFIRMED）という Status が割り当てられ，権限を持った

(注3)：https://bugzilla.mozilla.org/home

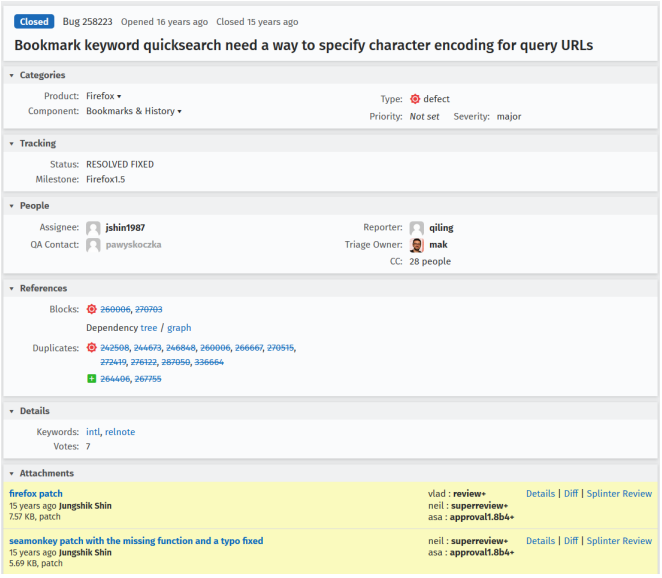


図 1　BMO のスクリーンショット

開発者によってバグの存在が確認されると新規のバグ（NEW）に遷移する．その後，そのバグに担当者が割り当てられ，担当者割り当て済み（ASSIGNED）に遷移する．UNCONFIRMED から ASSIGNED までの Status は開かれた状態であり，スレッドで議論が交わされ，修正パッチの提出やそのレビューが行われる．バグに何らかの決着がつくと完了（RESOLVED）に遷移し，バグは閉じられる．この時，どのような理由で解決したかに応じて，修正済み（FIXED）や重複（DUPLICATE）などの情報が Resolution として設定される．なお，FIXED はバグが修正された場合に設定され，DUPLICATE は既存のバグ報告と重複した内容だった場合に設定される．この時，議論は既存のバグのスレッドで引き続き行われる．RESOLVED のバグは，報告者や権限を持ったアカウントにバグの修正を確認されると検証済み（VERIFIED）に遷移する．何らかの理由で再発したバグは再発生（REOPENED）として再び開かれる．

3. Research Questions

本研究では，MB バグに関する様々な特徴の理解を目的として，3 つの Research Question（RQ）を設定し調査を行う．RQ

表 1　バグの情報

情報部分の要素	説明
ID	バグを一意に表現するための番号
Product	どの製品のバグなのかを示す
Status	バグの状態を示す．図 2 に従って遷移する
Resolution	バグがどのように解決されたかを示す
Opened	そのバグが報告された日時
Closed	そのバグ報告が閉じられた日時
Priority	そのバグを修正する優先度を示す．
Assignee	そのバグを修正する責任者のアカウント
Reporter	そのバグ報告を行ったアカウント
Duplicates	そのバグ報告以降に報告された重複バグ ID の一覧
Attachments	そのバグのコメントに添付された画像や修正パッチ

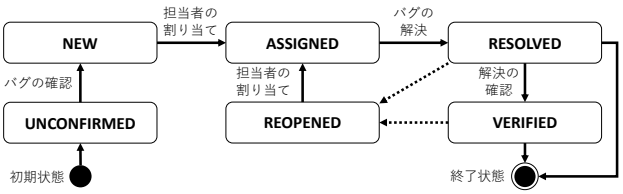


図 2　BMO のバグライフサイクル

の構築に当たって，まずバグ報告の投稿から解決までに至るプロセスを，報告と修正過程の 2 つに分類して考える．ここでの報告とは，バグ報告が投稿されてから担当者割り当て状態に遷移するまでのプロセスに該当しており，実際のバグの修正作業が開始されるまでの工程となる．さらに修正過程とは，担当者割り当て状態から解決状態までのプロセスであり，実際のバグ修正作業に該当する．この 2 つのプロセスそれぞれについて，以下 RQ を設定した．

RQ1: MB バグの報告内容にはどのような特徴があるか

RQ2: MB バグの修正過程にはどのような特徴があるか

さらにバグ修正の過程ではなくバグそのものの性質として，バグの発生原因と影響，及び修正内容についても以下 RQ を設定し調査する．

RQ3: MB バグの原因と影響には何があるか

さらに上記 3 つの RQ について，より具体的なサブ RQ を考える．まず RQ1（報告内容）については，他のバグ報告と同一の報告が行われる重複バグ報告について調べる．BMO をはじめとするバグ管理システムにおいて，重複バグ報告の存在は解決すべき課題となっている [8] [9]．重複バグの多さは，重複確認に要する開発者のエフォートの増加に繋がるためである．さらに，修正の優先度という属性もバグ修正プロセスにおいて重要な性質である．この優先度は，プロジェクト内の開発者による修正作業の優先順位を表しており，バグの重大さとは独立して設定される．もし，MB バグがそれ以外の通常バグと比較して優先度が高い傾向にあれば，MB バグが積極的に修正べき事項だと捉えられていると見なすこともできる．これら重複と優先度を確認するために，2 つのサブ RQ を設定する．

RQ1a: MB バグの重複報告数は通常バグと比べて異なるか

RQ1b: MB バグの優先度は通常バグと比べて異なるか

RQ2（修正過程）については，まず修正に要する期間に着目する．ここでは修正期間と放置期間それぞれについて考える．修正期間とは，バグの報告からそのバグが解決される（RESOLVED）までの期間である．放置期間とは，バグの報告から Reporter 以外によるコメントが投稿されるまでの期間である．つまり，当該バグの修正が開始するまでにどの程度の期間を要したかを意味しており，これが長いほど開発者の関心が低いバグであると見なすことができる．さらに修正過程の特徴として，バグ報告スレッド内でのコメント数と修正パッチの提案回数を調べる．コメント数の多さは開発者や報告者による議論の多さを表しており，多くの場合，コメント数の多さはバグ修正の難易度の高さと正の相関を持つと考えられる．修正パッチの提案回数の多さは，修正方法がレビューによって何度も差し戻しされた，というケースで多く発生するため，修正の難易度を表すと考えられる．上記 4 つの指標それぞれに対応して以下サブ RQ を設定する．

RQ2a: MB バグの修正期間は通常バグと比べて異なるか

RQ2b: MB バグの放置期間は通常バグと比べて異なるか

RQ2c: MB バグのコメント数は通常バグと比べて異なるか

RQ2d: MB バグの修正パッチ提案回数は通常バグと比べて異なるか

最後に，RQ3（発生原因と影響）について述べる．RQ1 と RQ2 はバグの修正プロセスに着目した内容であるが，RQ3 ではバグそのものの性質について考える．まずバグ自体を考えるに当たって，ソースコードのような具体的な粒度での発生原因の整理と体系化は重要な課題である．もしバグ発生原因のパターンを発見することができれば，静的解析ツール [10] などへの組み込みが可能となりバグの早期発見が実現できる．また影響はバグ修正の優先度やバグ自体の重大さに強く関連する性質である．MB バグは，直感的には文字化けのバグのように UI への影響が多いと推測できるが，クラッシュ等の致命的な影響を及ぼすことも考えられる．これらバグ自体の性質について以下 2 つのサブ RQ を設定する．

RQ3a: MB バグの発生原因は何か

RQ3b: MB バグはどのような影響を与えるか

4. 調 査 対 象

調査の題材として，Mozilla プロジェクトで開発されている Web ブラウザ，Firefox を用いる．Web ブラウザを題材とする理由は，Web ブラウザは WWW の様々な言語のリソースを閲覧するという性質上，徹底して国際化・地域化が施されているためである．当然，Firefox も国際化・地域化が施されており，様々な言語環境において様々な言語のリソースを表示，閲覧することが可能である．そのため，マルチバイトの適切な処理は Web ブラウザの必須要件となっており，同時に多数の MB バグが発生，修正されていると考えられる．

MB バグ，及び比較対象となるその他のバグ（以降，通常バグ）の抽出には，BMO の詳細検索機能を用いて，表 2 に示す条件で検索を行った．RQ3 においてバグ修正の内容を調査する都合上，Status が完了（RESOLVED）か確認済み（VERIFIED）でかつ，Resolution が修正済み（FIXED）のバグ報告のみに限定した．言い換えれば，現在修正中のバグや不適切と認定されたバグ報告は除外されている．加えて，MB バグの検索条件には Resolution が重複（DUPLICATE）というクエリを追加し，この検索に該当した重複報告の重複先となる FIXED なバグを調査集団に加えた．これは MB バグの絶対的な母数が少なく，可能な限り多数の MB バグを洗い出すためである．上記の条件を統一した上で，MB バグのみを抽出するためにスレッドのコメント欄に対して，MB バグに関連するキーワードを指定した．英語表現としての揺れを吸収するために，`multi` の直後のスペースやハイフンの有無を許容した．さらにマルチバイトを表す別の表現として `non-ascii␣character` というクエリを追加した．

これら検索条件に該当した件数は表 2 の下部に示す通りである．通常バグに関しては，検索に該当したバグの内 10,000 件を

そのまま調査の母集団として用いる．MB バグは 78 件が検索条件に該当した．この 78 件には MB バグではない，偶然検索に該当してしまったバグが含まれている．これらを取り除くために，実際に MB バグであるかどうかをスレッドの中身を目視により精査し選定した．現在，31 件については目視調査を完了しており，その中から MB バグであることを確認した 15 件を続く RQ の回答に用いる．なお，目視調査が完了していない 47 件については，現在調査を継続中である．

5. RQ1: 報告内容

5.1 RQ1 の調査結果

RQ1a(重複報告数) の調査結果として，MB バグと通常のバグそれぞれの重複報告の割合を図 3 に示す．重複報告件数を 0 件，1 件以上 5 件以下，6 件以上 10 件以下，11 件以上に 4 分割し，各区間に属すバグの割合を棒グラフで表している．MB バグも通常のバグも，重複報告が 0 件であるバグが大半であり，重複報告が 1 件以上 5 件以下のバグの割合はあまり変わらない．しかし，MB バグは通常のバグに比べて重複報告件数が 11 件以上のバグの割合が高いことが確認できる．また，調査した MB バグ 15 件に対する総重複報告数は 30 件（200.0%）であった．通常のバグ 10,000 件に対する総報告重複件数は 3,056 件（30.6%）であり，MB バグの重複報告は非常に多いことがわかる．

RQ1b(修正優先度) の調査結果として，各 Priority の全体に占める割合を表す棒グラフを図 4 に示す．Priority は，P1 が最も修正優先度が高いことを表し，P2，P3，P4，P5 の順に優先度は低くなる．Priority は Assignee によって設定されるが必須の項目ではないため，設定されず，No set となるバグも存在する．グラフからは，P1〜P5 のすべてについて MB バグは通常のバグより割合が低く，Priority が設定されていないバグの割合が高いことが確認できる．このことから，MB バグの修正優先度は低い傾向にあると分かる．

5.2 RQ1 の考察

RQ1a で述べた結果から，MB バグの重複報告は通常のバグよりも多い傾向にあることが分かった．これは，MB バグはバグ報告の際に，重複する既存のバグの有無を確認できていないためである．その理由として，MB バグの報告者の母国語に起因する可能性が考えられる．MB バグはその性質上，アルファベット以外の文字を利用する地域で確認される傾向にあり．実際，本研究で扱った 15 件の MB バグの Reporter のユーザーネームを確認したところ，4 件が日本人で，4 件が中国人であった．BMO では英語でバグが管理されているため，英語を母国語としないユーザが既存のバグとの重複を確認せず，または，確認できずに重複した報告を行ってしまうケースが多いと推測できる．そのため，重複報告を検出する仕組みが求められる．

また，RQ1b から MB バグは修正優先度が低い傾向にあることが分かった．これは，MB バグがその性質上，一部の地域のユーザにしか影響を与えない場合が多いことに由来すると考えられる．また，後述する 7.1 節で述べるが，MB バグはソフトウェアの機能面に影響を与えないバグが多いことも理由の一つだと考えられる．

表 2　検索条件と条件該当件数

	MB バグ	通常のバグ
Product	Firefox	
Status	RESOLVED or VERIFIED	
Resolution	FIXED	

Comment	`multiple␣byte`
	`multi␣byte`
	`multi-byte`
	`multibyte`
	`non-ascii␣character`
		—

クエリ該当件数	78 件	10,000 件
目視確認済み件数	31 件	—
目視確認の結果	15 件	—

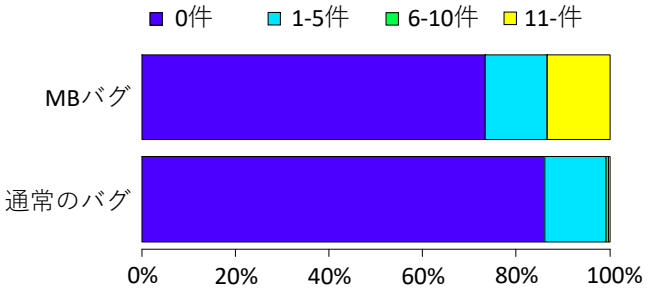


図 3　RQ1a: 重複報告件数

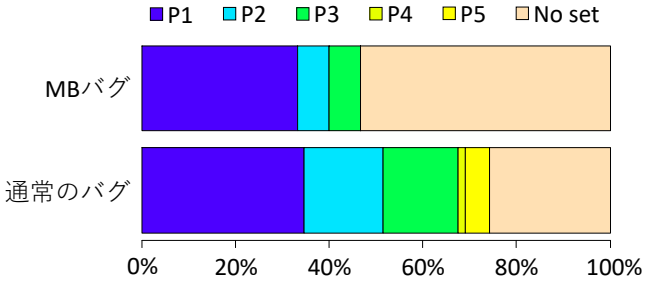


図 4　RQ1b: 修正優先度

RQ1 の結論

MB バグは重複報告が多い．また，MB バグの修正優先度は低い傾向にある．

6. RQ2: 修正過程

6.1 RQ2 の調査結果

本研究では 2 群間の差の検定として，Wilcoxon 順位和検定を用いる．図 5 は，RQ2 で扱うそれぞれの指標についての箱ひげ図である．視認性確保のため，外れ値の一部を表示していない場合があることに注意されたい．

RQ2a(修正期間) の調査結果として，MB バグと通常のバグの修正期間を図 5a に示す．検定の結果，有意差は確認できず，MB バグと通常のバグに大きな差は見られなかった．

RQ2b(放置期間) の調査結果として，MB バグと通常のバグの放置期間を図 5b に示す．検定の結果，有意水準 5%で有意差が認められた．MB バグは通常のバグよりも放置期間が短いことが確認でき，修正対応が早いことが分かる．

RQ2c(コメント数) の調査結果として，MB バグと通常のバグのコメント数を図 5c に示す．検定の結果，有意水準 1%で有意差が認められた．MB バグは通常のバグよりもコメントが多い傾向にあることが確認でき，バグ修正のために多くの議論を要することが分かる．

RQ2b(放置期間) の調査結果として，MB バグと通常のバグの修正パッチ提案回数を図 5d に示す．検定の結果，有意水準 1%で有意差が認められた．MB バグは通常のバグよりも修正パッチの提案回数が多いことが分かる．

6.2 RQ2 の考察

RQ2a の結果から，MB バグの修正期間の長さは通常のバグと差がみられなかった．それに対して，RQ2b の結果は MB バグは報告から修正対応までの期間が短いことを示している．このことから，MB バグは修正対応が行われてから修正が完了するまでの期間が長いといえる．また，RQ2c が示すように，MB バグはコメント数が多い傾向にあることから，MB バグを修正するためには多くの期間と議論を要すると考えられる．そのため，MB バグの修正は難易度が高いと推測する．RQ2d が示すように，MB バグの修正で提案されるパッチの数が通常のバグよりも多いことも，MB バグの修正は難易度が高いと考える根拠である．

RQ2 の結論

MB バグの修正期間は通常のバグと差が見られなかった．しかし，放置期間については MB バグのほうが短い．また，MB バグは議論が活発に行われ，多くの修正パッチが提案される．このことから MB バグは修正の難易度が高いと推測する．

7. RQ3: 原因と結果

7.1 RQ3 の調査結果

表 3 では，目視調査を行った 15 件の MB バグの原因を分類している．エンコード処理において適切なマルチバイト文字を定義していないことが原因として最も多く，全体の半分を占めた．文字数を正しくカウントできていないバグは，文字数をカウントすべき処理で，バイト数をカウントしていることが原因のバグである．

表 4 では，目視調査を行った 15 件の MB バグの影響を分類している．MB バグを，ソフトウェアの機能に影響を与えるバグと，UI/UX といった機能面以外に影響を与えるバグの 2 つに大別し，そのうえでそれぞれに対し，バグの症状での分類を行った．11 件のバグが，文字の表示が意図とは異なるという内容であり，全体の 7 割を占めた．文字の認識が正しく行われないバグは Web ブラウザに限らず，文字を入力として扱うすべてのソフトウェアで発生しうるが，データ通信のエラーを引き起こすバグは Web ブラウザ特有のバグであった．

表 3　RQ3a: MB バグの原因

原因	件数
文字コードの定義不足	10 件
文字数を正しくカウントしていない	2 件
MB 文字の制御文字のエスケープ	2 件
UTF-8 における置換文字の無視	1 件

7.2 RQ3 の考察

RQ3a では，MB バグの原因は適切な文字コードを定義していないという単純な原因が大半を占めることが分かった．これは，欧米に代表されるラテンアルファベット使用言語圏においては，コンピュータ環境でマルチバイト文字を使用することがないため，マルチバイト文字に対する関心が薄いことが理由として考えられる．

また，RQ3b では，MB バグはソフトウェアの機能面ではなく UI/UX に影響を与えるバグが多くを占めることが分かった．しかし，15 件のうち 4 件はソフトウェアの機能に影響を及ぼすバグであり，MB バグは軽視できないことがわかる．

RQ3 の結論

MB バグは文字コードの定義不足という単純な原因が大半を占めた．MB バグは非機能面への影響を及ぼす傾向にあるが，ソフトウェアの機能面に影響を及ぼすバグも存在する．

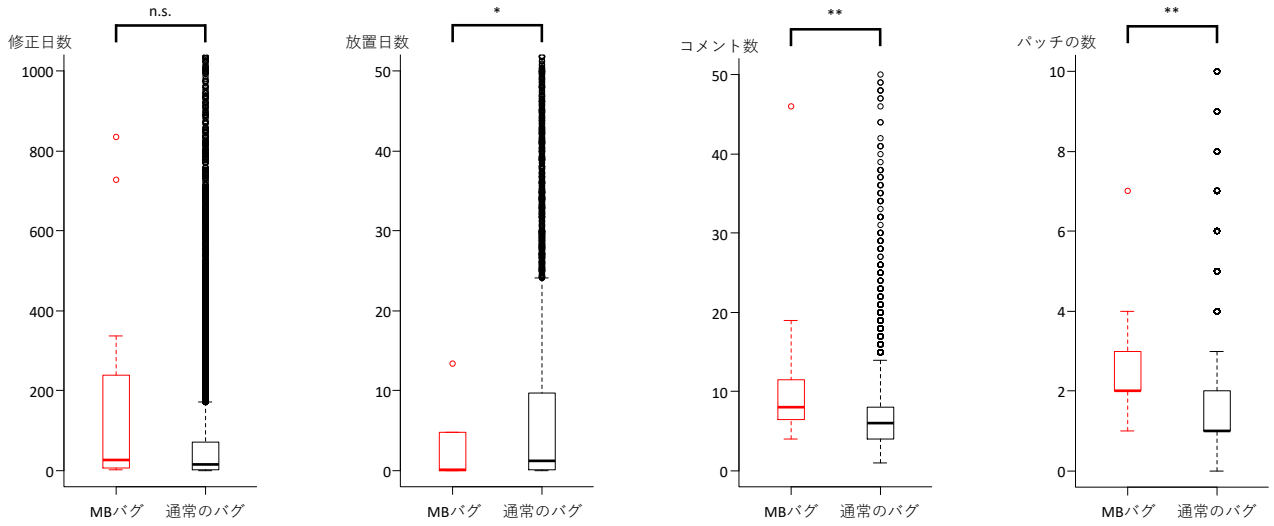
8. 妥当性の脅威

MB バグを収集するにあたって，検索条件を満たした 78 件中 31 件を目視調査したが，目視調査に至らなかった 47 件についても調査を行った場合，異なる結果が得られた可能性がある．また，マルチバイト文字に起因するバグであっても，今回用いた検索クエリがコメントで使用されていないバグの存在が考えられる．“UTF-8” や “korean” などで検索したところ，本研究における検索条件での検索結果には含まれない MB バグがいくつか見つかった．しかし，特定の言語や特定の文字コードをクエリとして検索を行った場合，調査対象に偏りが生じる可能性を考慮して今回はクエリとして採用しなかった．

MB バグと比較した 10,000 件の通常のバグは，BMO の検索結果として表示できるバグの件数の上限が 10,000 件であったことからこれらを通常のバグとして扱ったが，実際に検索条件を満たすバグは 10,000 件以上存在する．そのすべてを通常のバグとして調査した場合，異なる結果が得られていた可能性がある．

表 4　RQ3b: MB バグの影響

機能への影響	症状	件数
なし	正しい表示が行われない	11 件
あり	入力した文字が正確に認識されない	2 件
	データ通信におけるエラー	2 件



(a) RQ2a: 修正期間　(b) RQ2b: 放置期間　(c) RQ2c: コメント数　(d) RQ2d: 修正パッチの提案回数

図 5　RQ2 の結果(注4)

9. おわりに

本研究では，Firefox における MB バグの特徴を複数のメトリクスを用いて調査した．その結果，MB バグは重複報告が多いこと，修正の優先度が低い傾向にあること，修正の難易度が高いことを示した．また，目視調査により，バグの原因と影響の分類を行うことで，MB バグの原因の大半は文字コードの定義不足という単純な原因であることを示し，MB バグの影響は非機能面へ及ぶ傾向にあるが，ソフトウェアの機能面に影響を及ぼすバグも存在すると示した．

今後の課題としては，目視調査が完了していない 47 件のバグについて引き続き調査を進めること，Firefox 以外のソフトウェアにおいても MB バグに同様の傾向が見られるかを検証することが考えられる．また，開発者の所在地を取得することで，各地域におけるマルチバイト文字への意識の違いを検証することも考えられる．

謝辞　本研究の一部は，日本学術振興会科学研究費補助金基盤研究（B）（課題番号：18H03222）の助成を得て行われた．

文　献

[1] S. Lu, S. Park, E. Seo, and Y. Zhou, “Learning from Mistakes: A Comprehensive Study on Real World Concurrency Bug Characteristics,” In Proc. International Conference on Architectural Support for Programming Languages and Operating Systems, pp.329–339, Association for Computing Machinery, 2008.

[2] S. Zaman, B. Adams, and A.E. Hassan, “Security versus performance bugs: A case study on firefox,” In Proc. Working Conference on Mining Software Repositories, pp.93–102, 2011.

[3] Z. Li, L. Tan, X. Wang, S. Lu, Y. Zhou, and C. Zhai, “Have things changed now? an empirical study of bug characteristics in modern open source software,” In Proc. Workshop on Architectural and System Support for Improving Software Dependability, pp.25–33, Association for Computing Machinery, 2006.

[4] L. Tan, C. Liu, Z. Li, X. Wang, Y. Zhou, and C. Zhai, “Bug characteristics in open source software,” Empirical Software Engineering, vol.19, no.6, pp.1665–1705, 2014.

[5] N. Bettenburg, S. Just, A. Schröter, C. Weiundefined, R. Premraj, and T. Zimmermann, “Quality of bug reports in eclipse,” In Proc. Workshop on Eclipse Technology EXchange, pp.21–25, Association for Computing Machinery, 2007.

[6] N. Bettenburg, R. Premraj, T. Zimmermann, and S. Kim, “Duplicate bug reports considered harmful ... really?,” In Proc. International Conference on Software Maintenance, pp.337–345, 2008.

[7] 矢野啓介，プログラマのための文字コード技術入門，WEB+DB PRESS plus シリーズ，技術評論社，2019.

[8] Y. Tian, C. Sun, and D. Lo, “Improved duplicate bug report identification,” In Proc. European Conference on Software Maintenance and Reengineering, pp.385–390, IEEE Computer Society, 2012.

[9] A.T. Nguyen, T.T. Nguyen, T.N. Nguyen, D. Lo, and C. Sun, “Duplicate bug report detection with a combination of information retrieval and topic modeling,” In Proc. IEEE/ACM International Conference on Automated Software Engineering, pp.70–79, Association for Computing Machinery, 2012.

[10] N. Ayewah, W. Pugh, D. Hovemeyer, J.D. Morgenthaler, and J. Penix, “Using static analysis to find bugs,” IEEE Software, vol.25, no.5, pp.22–29, 2008.

(注4)：n.s. は有意差が認められないことを表す．*は有意水準 5%で，**は有意水準 1%で有意差が認められたことを表す．