社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－DBS－141 (17)
2007－GN－ 62 (17)
2007／1／26

Webにおける構造化エディタを用いたコラボレーション環境の実現

岡田 尚希† 伊藤 一成† Martin J.Dürst†

† 青山学院大学 理工学部
〒229–8558 神奈川県相模原市淵野辺 5–10–1
E-mail: onaoki@sw.it.aoyama.ac.jp, {kaz, duerst}@it.aoyama.ac.jp

あらまし 近年，専門のウェブクリエータはスタイルシートを併用することで，構造化された（X)HTMLを作成するようになった．一方，一般ユーザはblogやwikiなどを用いて多くのコンテンツを生成しているが，アドホックな方式であることは否めない．そこで我々は高度に構造化された妥当な(X)HTMLコンテンツが容易に生成できるWebベースのエディタを実装した．本稿では，エディタの詳細およびそれを用いた情報共有，コラボレーションの展望について述べる．

キーワード HTMLエディタ, 情報共有, ドキュメントの構造化

Supporting Collaboration using a Web-based Structured Editor

Naoki OKADA†, Kazunari ITO†, and Martin J. DÜRST†

† College of Science and Engineering, Aoyama Gakuin University
Fuchinobe 5–10–1, Sagamihara, Kanagawa, 229–8558 Japan
E-mail: onaoki@sw.it.aoyama.ac.jp, {kaz, duerst}@it.aoyama.ac.jp

Abstract Recently, professional Web creators are moving to structured (X)HTML, in combination with CSS. On the other hand, general users are more and more able to create content via blogs or wikis, but only in an ad-hoc way. We have created a Web-based configurable editor that allows the easy creation of highly structured and valid (X)HTML content. In this paper, we describe this editor and discuss its use in the context of information sharing and collaboration.

Key words HTML Editor, Knowledge Sharing, Structurized Documents

1. はじめに

近年，専門家や技術者の間でHTMLの情報構造としての役割が再認識されている．HTMLは誕生当初から，情報構造を記述するものであるとし，コンテンツとプレゼンテーションの分離は，Webの原理の一つになっている[1]．実際に企業やデザイナー，技術者の現場でも，仕事の明確な分離，再利用，効率化のため，情報構造はHTML，デザインはCSSと使い分けされるのが当たり前になってきた．一般ソフトウェアではAmaya[2]のような，構造化を重視したエディタの開発が行われている．しかし，Webページ内で使用できるHTMLエディタはZoho Writer(注1)，Think Free(注2)などのように，体裁（レイアウト）の編集を主とするワープロ感覚のものがほとんどを占め，本格的な情報構造の編集を行えるエディタはない．

また，気軽に情報を生成，発信できる新しいサービスや仕組みが誕生した．不特定多数のユーザから自然発生的に発信される膨大なコンテンツが，暗黙的に構造化され計算機可読な情報となれば，サーチエンジンなどWeb

(注1)：http://writer.zoho.com/
(注2)：http://www.thinkfree.com/

上におけるサービスに加え，比較的小規模なグループにおける協調的な作業においても飛躍的な質の向上が期待できる．そこで，直感的に情報構造のある XHTML 文書またはその一部を Web 上で生成可能なエディタを実装した．

本稿では初めに構造化エディタの情報共有・活用などコラボレーション環境への役割を説明し，3 章以降で，そのコラボレーション環境を実現する本エディタの説明を行う．

2. 情報共有・活用のための構造化エディタの要件

2.1 様々なサービスへの柔軟な対応

エディタは Web メール，掲示板，情報共有のアプリケーションなど，様々な用途で使用されるので，求められる機能は多様である．例えば，情報共有のアプリケーションでは，テーブルやリストなど，詳細な情報構造の記述が求められる．一方，大多数のユーザが書き込みを行う掲示板などでは，最低限の意味づけのみが求められる．つまり，多機能であるだけでなく，各機能を状況にあわせて設定できる柔軟なエディタが望ましいと考えられる．

2.2 ユビキタスなコラボレーション環境への対応

近年，ユビキタスという言葉が注目されている．情報家電，携帯電話などあらゆる情報機器が情報の発信源として機能を有するようになった．しかし，ユビキタス環境におけるコラボレーション環境の実現において専用アプリケーションによる利用では，OS の問題やバージョンの相違による問題が存在する．その観点から，ブラウザがある環境であれば，OS に左右されることなく，どこでも実行可能な Ajax という考えが広まってきた．

2.3 XHTML による情報共有・交換の促進

独自タグの使用可能などの機能を備えた情報記述言語 XHTML が勧告 [3] されている．XHTML は HTML に比べルールが厳密であるため，処理にかかる負荷が軽減される．そのため，データ処理が高速化され，アプリケーションによる安定動作が促進される．また，XML ベースなので，情報の意味構造を計算機が分析可能であり，必要な情報だけをアプリケーションを通して抽出できる．また，データ抽出や部分利用したり，XSLT による文書変換が可能である．これは，データの再利用性を高め，情報の共有化を促進する．

さらに，名前空間の利用により，SVG，RDF，RSS，FOAF などの XML アプリケーションを埋め込むことができ，将来登場する XML アプリケーションを利用する場合も XHTML の仕様自体は変更する必要がない [4].

2.4 パーソナライゼーション

現在のインターネットサービスは，ユーザがコンテンツを直接アクセスする形態である．しかし，インターネットの爆発的普及が進むにつれ，従来より言われてきた情報洪水の問題がますます顕在化してきており，自分の欲する情報を手軽に獲得するのが困難になっている．文書にメタデータが付加される事で，文書の内容を計算機が判読でき，閲覧ユーザの嗜好にあわせて提供可能である．さらに，人々のトレンドをリサーチすることも可能である．

このように，今後ユーザがいろいろな場所からネット上の様々なサービスを連携して利用する形態が増えてくると予想されるため，ユーザ，コンテンツ，サービスを XHTML 技術でメタデータで統合して，これらを動的に融合して管理・活用し，ネットワーク上の資源をユーザが好む形式で提供できる枠組みが求められている．

2.5 様々なメタデータ仕様のサポート

lightweight なメタデータ仕様として，XHTML にメタデータを埋め込んで，コンテンツの情報を，より詳細に構造化しようとする microformats が注目を浴びている．[5]．例えば文書を microformat の代表的な仕様の一つである hCard のフォーマットに基づいて構造化することで，アドレス帳の標準的なファイル形式である vcf 形式に変換可能である．hReview で記述されたレビュー記事も一覧表示，抜き出しが可能である．

また，hCalendar で構造化されているイベントや予定の情報も，Greasemonkey(注3) と googlehcalendar(注4) を用いて google Calendar に登録することが可能である．

リンクへの意味づけを行う VoteLinks や rel-enclosure に対応したソフトウェアはまだ少数だが，VoteLinks の値を集計しレビューや意見が記述された記事に対する評価情報を集計するといった活用方法が考えられる．またこれらのリンクは CSS や JavaScript 等を用いて，他のリンクと視覚的に差別化して表示可能であり，ユーザはリンク先の情報を容易に判別できる．

ユーザが仕様にのっとった記述を容易にかつ暗黙的に生成することで，他のアプリケーションとのデータ連携を推進し，文書をより有効に用いるための仕組みが確立されていくであろう．

3. エディタの実装

3.1 概　　要

JavaScript で実装し，複数の Web ブラウザで実行可

(注3) : http://greasemonkey.mozdev.org/
(注4) : http://makedatamakesense.com/

能にし，ユビキタスなコラボレーション環境を実現した．エディタ単体で使用できるだけでなく，任意の iframe 要素を配置するだけで，既存のページに自由かつ簡単に組み込み可能である．また，多くのサービスに組み込まれ使用されるよう，エディタの機能をカスタマイズすることも可能である．

本エディタは，計算機可読の XHTML を常に well-formed に出力可能である．それゆえ，作成された情報を計算機を通してデータ交換・共有を可能にし，コラボレーション環境を実現する．また，独自要素・属性の使用も設定により追加可能のため，より文書の意味に近い高度な情報を valid に付加することが可能である．さらに，高度な情報構造の記述を大多数のユーザが直感的に行えるインターフェースにし，コラボレーション環境を促進する．

3.2 マークアップ方法

本エディタでは，XHTML の仕様に基づいた編集が可能である．XHTML における要素タイプは，大きくブロックレベル要素とインライン要素に分類され，それぞれマークアップ方法が異なる．

3.2.1 ソースコードの出力

`<p>` の内部には `<div>` を含むことはできないなど，要素には包含関係が存在する．また，各要素ごとに取りうる属性が決められている．HTML の処理エンジンは，とりあえず表示することを最優先にしているので，これらのルールはあいまいである．例えば終了タグの省略など well-formed でないソースコードも勝手に解釈されてしまう．このあいまいさは，厳密さを必要とするデータ交換や共有の目的を考える上で，大きな弊害となっている．XHTML は XML の実装であるため，データ交換・再利用などを図ることができ，また，いままで HTML のあいまいさを解釈する必要があったエンジンのロジックを簡素化できる．そのような利点を持つため，今後 HTML から XHTML に移行していくと思われる．

そこで，要素および属性の定義を，XHTML 1.0 [6] の定義を参考に外部ファイルに記述し，それに従い要素編集が行われることで，常に XHTML の仕様に沿った well-formed な出力だけではなく， valid なソースコードの書き出しを可能にした．また，この外部ファイルの定義は編集できるので，エディタでの使用可能要素・属性の追加・削除ができ，より自由な情報構造の記述が可能である．このように，高度な情報構造を持つ XHTML を作成可能なので，本エディタにより作成された情報をアプリケーションを通し交換・再利用可能にし，コラボレーション環境を実現する．

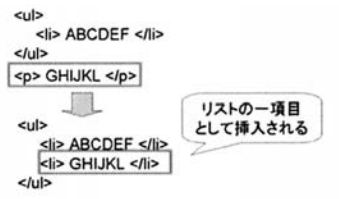

図 1　直前にリスト要素がある要素を `<li>` に置換した場合

図 2　直前にリスト要素がない場合

3.2.2 ブロックレベル要素

ブロックレベル要素は見出し，段落など文書を構成する基本要素となるものである．ブラウザでの表示に際しても，ひとつのブロック（通常改行を伴う表示上のまとまり）として扱われる．そのため，構造化の促進において，要素の入れ子など，詳細に要素編集が行えることが重要である．この詳細な要素編集実現のため，現在選択している要素に対して，次の 5 通りの操作を可能にした．

（1） 指定要素の内部を新たに要素で加工
（2） 指定要素を別の要素に置換
（3） 指定要素を新たな要素で包含
（4） 指定要素の前に新要素を挿入
（5） 指定要素の後ろに新要素を挿入

さらに，`<li>`，`<dt>`，`<dd>` 要素の編集は，前後の要素に応じて処理が変化する．例えば 図 1 のように，直前にリスト要素がある `<p> GHIJKL </p>` を `<li>` で包含しようとする場合，ユーザは，直前のリストの一項目として追加したい場合が多い．それゆえ，新しく生成された `<li>` は，直前のリスト要素に挿入される．直前のリストとは別のリストにしたい場合は，この要素を `<ul>` もしくは `<ol>` で処理される．

しかし 図 2 のように，直前にリスト本体の要素が存在しない場合に `<li>` による処理を要求するユーザは，新規リストの一項目として追加したい場合が多い．なのでこの場合，新規にリストを作成し，その新しいリストの一項目として挿入する．

このように，要素の編集を柔軟に変化させることで，より直感的に要素編集を行うことが可能である．

3.2.3 インラインレベル

フレーズに役割を与えたりするインライン要素の編

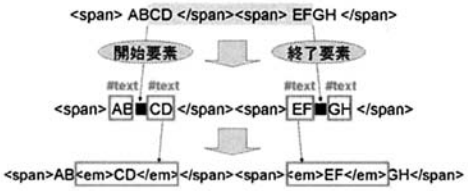

図 3　複数の要素を跨いだ時のマークアップ処理

集は，テキストカーソルで選択した範囲に対して行う．JavaScript には要素の編集を簡単に実現できる命令が存在する．しかし，使用できる要素があらかじめ決まっており， 要素や <dfn> などを使用することのできない要素がある．

XHTML では，ユーザの独自要素の使用が認められており，より多くの情報を付加することが可能である．

そこで，幅広い要素の編集を可能にするため，3 章 1 節で説明した，要素・属性を定義した外部ファイルを元に，要素を編集する独自関数を作成した．以下に，この独自関数の説明を記す．

JavaScript では，選択範囲は Range (InternetExplorer では TextRange) で扱うことができ，選択範囲の編集も可能である．しかし，この Range はブラウザによって名前が違うように，機能も大きく異なる．そのため，Range は，マークアップ処理を行う前に，選択範囲の終始を表す要素の挿入に用い，マークアップ処理自体は Range を用いず，文書の一部である要素を取得，変更，削除するためのメソッドとプロパティを有する DOM API を用いた操作にすることで，他ブラウザへの容易な対応を可能にする．

マークアップは，選択範囲に含まれる各要素の入れ子の末端であるテキストノード（#text）を指定要素で包含することで，複数の要素を跨いでいた場合のマークアップを可能にする．

例えば 図 3 に で囲われている ABCD と EFGH を，C から F へ要素を跨いでマークアップをした場合を示す．まず，選択範囲の前後に選択範囲の開始と終了を表す要素を挿入する．すると AB，CD，EF，GH の四つの #text 要素が作られる．そして，開始要素から CD，EF，終了要素へと DOM 操作で隣へ移動しながら #text 要素をマークアップしていく．これにより，要素間を跨いでいたとしても CDEF がマークアップされる．

しかし，選択範囲の #text 要素へのマークアップだけでは，複数の要素を覆ってマークアップを実行した場合，複数の要素に分割されてしまう．例えば 図 4 に示すように ABCDEF という単語を強調しようとした場合，選択範

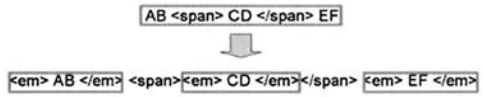

図 4　#text 要素へのマークアップで，単語 ABCDEFGHI を で強調

囲の #text 要素をマークアップした結果，ブラウザ表示上は ABCDEF の強調に見えるが，内部では三つの単語 AB，CD，EF の強調になってしまっている．これは，HTML としては valid である．しかし，これはユーザの意図したマークアップではないだろう．

そこで，要素の整形を行う関数を作成し，ユーザの意図したマークアップを可能にすることで，構造化エディタの質の向上を実現した．図 4 の場合，二つ目の は， の子要素であり，この の子要素は だけである．このような場合 CD は CD としても問題はない．すると， 要素の三つは隣接する．

インライン要素が，同要素名で意図的に個別のものとしてマークアップされる場合はほとんどない．そのため，同要素名でインライン要素が隣接している場合は結合する．その結果，三つの隣接した は一つに結合し，ABCDEF というように，一つの単語として強調が実現できる．

#text 要素に行うマークアップと，要素の整形処理，この二つの処理により，複数の要素を跨いでいても，マークアップの処理は実行され，かつ，ユーザの意図する通りの要素の編集を実現する．

3.3　テーブル

テーブルは，企業の決算情報，見積書，在庫一覧などで使用され，テーブルが自由に，かつ常に正しく操作できることは，データ交換において非常に有効である．

しかし，Web ページ内で使用できるエディタでは，結合ができなかったり，テーブル内に結合セルがあった場合，範囲選択が思い通りに行われなかったりと，テーブルを完全に扱うことが出来るものはない．

そこで，行・列の挿入・削除はもちろん，選択セルの結合と，結合セルが存在した場合の正しい範囲選択を実現する独自関数を作成した．これにより，詳細な表が思い通り作成可能である．

例えば，図 5 に，6 行 6 列の 2 行 2 列から 4 行 4 列までが結合されているテーブルを，5 行 3 列から 3 行 5 列まで選択した場合の他エディタと本エディタの違いを示す．この場合，選択範囲は結合セルにかぶってしまっている．このような場合でも，図 5 に示すように，本エ

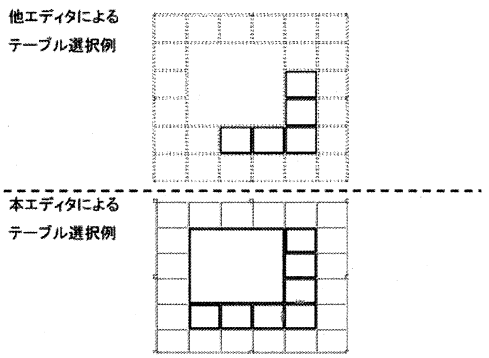

図 5　5 行 3 列目から 3 行 5 列目までを選択した場合の比較

ディタでは，表計算ソフトのように選択範囲を広げ，正しくセルの範囲選択が行われる．

3.4　インタフェース

本エディタでは，高度な情報構造の記述を目指すとともに，グループウェアとして，また，blog，wiki などのサービスの組み込みとして，幅広い場面で大多数のユーザに使用されることを目指す．そのため，誰もが直感的に情報構造の記述が可能なインターフェースを作成した．

ソフトウェアのような使い慣れたインターフェースは，ユーザの新しいアプリケーションに対する抵抗感を少なくし，高い好感度をもつ．

例として，書籍紹介ページ作成中のスクリーンショットを図 6 に示す．エディタ内部では，図 7，図 8 のような valid なソースコードが生成される．マークアップや，サブウィンドウの表示など，エディタにおける様々な要求は，ブラウザ表示上の最上部にあるドロップダウンメニュー（図 6 の　）より行われる．現在アクセスしている要素の属性情報や，文書のタイトルなどの情報は，サブウィンドウ（図 6 の　）で適宜参照し，編集できる．現在アクセスしている要素は，アウトラインで強調される（図 6 の　）．最下部では `<body>` 要素から現在アクセスしている要素までの入れ子状況がノードパスバーというもので表示される（図 6 の　）．

3.4.1　メニュー

要素のマークアップや操作のボタンを並べて表示するのは，多くの領域を占有してしまう問題と共に複雑度が増し，使いやすさが低下する．それは，編集の手違いを起こしやすくし，編集の共同作業など，コラボレーション環境において大きな障害となる．

そのため，ドロップダウンメニューを作成し，インライン要素，ブロック要素などカテゴリに準じて配置した．

図 6　書籍紹介ページの作成画面

図 7　エディタ内部で保持されたソースコード例

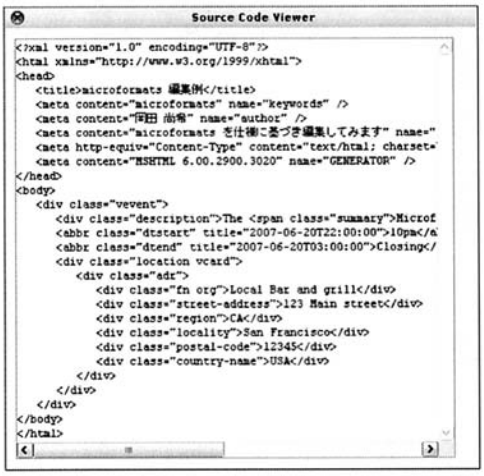

図 8　microformats 形式で保持されたソースコード例

3.4.2　要素の属性情報，文書情報の入力

要素の属性は，各要素の性質を定義する場合に使用される．同じ要素名でも属性の違いによりさまざまな働きをする．そのため，データ交換にも使用できる情報構造

を記述する上で，要素とともに属性の内容を把握・編集できることは重要なことである．

そのため，本エディタでは，要素に対して，属性を編集可能にした．属性はリストで表示され，それぞれを個別に設定できるため，一つの属性編集の誤りが，他の部分に影響を与えるということを起こさない．

また，HTML および XHTML では，タイトルの入力が必須である．そのため，タイトルおよび，文字コードやキーワードなどの文書情報を編集可能にした．

しかし，これら要素の属性や，タイトルなど文書情報は頻繁に編集する訳ではない．そこで，必要な時だけサブウィンドウとして表示し，参照できるようにした．

3.4.3 要素のアウトライン表示

 や などの要素は，表示が変わるわけではないため，マークアップの範囲が認識できない．そのため，重要な要素を消去してしまうなど，共同作業において失敗を引き起こす可能性がある．

そこで，テキストカーソルがある位置の要素をリアルタイムでアウトラインにより強調する機能を実現した．これは本エディタの，常に valid な要素が書き出される性質があって初めて有効に実現される．

これにより，ソースコードを確認することなく，テキストカーソルがある要素の範囲が把握できる．

3.4.4 ノードパスバー

同範囲に複数の要素がマークアップされている場合，要素のアウトライン表示だけでは現在どの要素にアクセスしているかが認識できない．

そこで，現在どの要素にアクセスしているかが一目で分かるよう，最下部に <body> 要素から現在アクセスしている要素までのノードパスを表示するバーを設置した．また，そのバーに表示されている要素名をクリックすることで，直接指定要素にアクセスできる．これにより，任意の要素へアクセスしてアウトラインで内容を確認し，属性を編集する，といった行為が GUI で行え，直感的な編集が可能である．

4. まとめと今後の課題

一般ユーザでも情報構造を重視した文書を直感的に作成可能にし，構造化の促進を実現する充実したエディタを実装した．本エディタは近日中に公開予定である．これによりコラボレーションツール作成の一端を担うことができれば本望である．

謝　辞

本研究の一部は科学研究費補助金（課題番号: 18700095）の支援によるものです．ここに記して謝意を表します．

文　献

[1] Ian Jacobs and Norman Walsh. Architecture of the World Wide Web, Volume One. http://www.w3.org/TR/webarch/, 12 2004.
[2] Irène Vatton. Amaya Overview. http://www.w3.org/Amaya/Amaya.html, 01 2006.
[3] W3C. Xhtml 1.0 the extensible hypertext markup language (second edition). http://www.w3.org/TR/xhtml1/, 2000.
[4] 益子貴寛. Web 標準の教科書. 秀和システム.
[5] 小川浩, 後藤康成. *Web2.0BOOK*. インプレス, 2006.
[6] 神崎正英. ユニバーサル HTML ／ XHTML. 毎日コミュニケーションズ.