JTTC 0002:2003　(Pub. 2003-07-19)

原稿校正標準マーク付け言語(SPML)
Standard Proofreading Markup Language(SPML)

1997-3-28

目次

原稿校正標準マーク付け言語(SPML)
Standard Proofreading Markup Language(SPML)

１．適用範囲　商業出版の原稿校正は通常，著者又は編集者(又は校正係)によって行われ，校正指示を施された原稿は，著者と編集者との間又は編著者を含む複数の著者の間で交換され，必要に応じてさらに重ねて校正指示を施される。企業内又はグループ内の文書作成作業における原稿校正では，権限を異にする幾人ものレビュー担当者の間で原稿が交換され，それぞれの立場での校正指示が施される。これらの分散環境での校正作業を，効率的にかつ正確に実行するため，電子化された原稿に対して，処理系に依存しない標準的な原稿校正マーク付けを行い，それを校正原稿情報として交換することが望まれる。

標準原稿校正マーク付け言語(SPML)は，この作業に用いる標準的な原稿校正マーク付けの規則を規定する。その抽象化モデルを既存の処理系で実現するため，標準一般化マーク付け言語(SGML)を用いて，各機能を記述する。SGMLによって記述されたSPMLは，SGMLの文書型定義(DTD)の要素集合として対象文書の構造記述に組込まれる。

この原稿校正マーク付けの対象となる電子化原稿は，主としてSGMLなどによってマーク付けされた構造化論理文書とする。この規格が規定する原稿校正マーク付けは，非SGML文書に対しても適用できる。アクセス制御及びフォーマティングに対する校正指示は，この規格の適用範囲外とする。

２．適合性

2.1 適合性　次の条件を満たす場合に，この規格に適合する。

(1) 7.2.2に示す文書変更構造を記述できる。

(2) 7.1.3に示す指令を実行できる。

2.2 応用分野　この規格は，文書処理システムが処理する構造化論理文書に適用することができ，一人以上による文書作成工程での校正作業に用いられる。

特に適する応用分野は，次のとおりとする。

(1) 出版のための電子文書（原稿）作成

(2) 出版のための校正

(3) 企業内文書の共同作成

(4) グループ内での共同文書作成

(5) 文書作成工程での変更履歴管理

３．引用文献　この規格は，その規定において次の規格を引用する。

ISO 8879:1986, Information processing - Text and office systems - Standard

Generalized Markup Language (SGML)

備考　JIS X 4151-1992 は，ISO 8879:1986 及び ISO 8879/Amendment 1:1988 の内容に，技術的追加及び編集上の変更を加えたものである。

ISO 9068:1988, Information processing - SGML Support Facilities - SGML Document Interchange Format (SDIF)

備考　JIS X 4171-1996 が，この国際規格に一致している。

JIS Z 8208-1996，印刷校正記号

４．用語の定義　この規格で用いる各用語は，その初出の箇所において定義を与える。

５．記法　通常の科学記法を規定文中で定義する。それに加えてSGMLの記法を使う。

６．基本概念

6.1 抽象テキスト体

6.1.1 抽象テキスト体及び抽象構成素　抽象構成素の並びを抽象テキスト体と定義する。混乱の恐れがなければ，抽象構成素を単に構成素と呼び，抽象テキスト体を単にテキスト体と呼ぶ。同一の抽象構成素が，抽象テキスト体のなかで2回以上出現することもある。個々の抽象構成素の出現を明確に示す必要がある場合には，その出現を構成素出現という。

備考　構成素(constituent atom)は，無定義の概念とする。構成素は文字である場合が多いが，必ずしも文字に対応しないバイト又はビット列であってもよく，文字列であってもよい。章，節などの文書の意味的な部分，又は行，段，ページなどの組版で意味をもつ部分を表すために，構成素を使うこともできる。抽象テキスト体を構成する各構成素が，等質である必要もない。

構成素は，いかなる対象でもよく，実際にはその内部に構造をもってもよい。しかし，抽象テキスト体を考える際には，その内部構造には立ち入らないで，基本要素として考える。構成素及び抽象テキスト体は，解釈の枠組み(見方)であって，絶対的な概念ではない。

空テキスト体は，全く構成素をもたないテキスト体とする。空テキスト体も，特別なテキスト体である。

テキスト体の構成素出現の総数を，そのテキスト体の長さという。

テキスト体を構成する各構成素出現には，その自然な順序に従って番号を振ることができる。最初の構成素出現を1とする番号を構成素出現番号と呼ぶ。構成素出現番号がnである構成素出現を，構成素出現nと表現する。

あるテキスト体Tの部分テキスト体は，Tの連続する構成素の並びを取り出して得られるテキスト体とする。もとのテキスト体T自体及び空テキストも，Tの部分テキスト体とする。

6.1.2 間隙　テキスト体の間隙とは，隣り合う構成素の間を示す。最初の構成素の前及び最後の構成素の後にも間隙が存在する。したがって，n個の構成素からなるテキス

ト T=C1C2...Cn には，(n + 1)個の間隙がある。これらを，次のとおり，0からnまでの番号によって示す。

間隙0 -- C1の直前
間隙1 -- C1とC2の間
間隙2 -- C2とC3の間
...
間隙n -- Cnの直後

空テキスト体は，ただ一つの間隙(間隙0)をもつ。

間隙を示す0から始まる番号を，間隙番号という。間隙番号がiである間隙を，間隙iと表現する。

6.1.3 構成素と間隙との位置関係　構成素出現番号n，mの構成素出現，間隙番号i，jの間隙があるとき，その位置関係について，次の表現を使う。

(1) n < m ---
構成素出現nは，構成素出現mの左にある。
構成素出現mは，構成素出現nの右にある。

(2) i < j ---
間隙iは，間隙jの左にある。
間隙jは，間隙iの右にある。

(3) i < n ---
間隙iは，構成素出現nの左にある。
構成素出現nは，間隙iの右にある。

(4) n <= i ---
構成素出現nは，間隙iの左にある。
間隙iは，構成素出現nの右にある。

6.1.4 範囲　間隙番号i，jの間隙があり，i <= jであるとき，i，jの組(i，j)を範囲という。特に i = j のとき，範囲は単一の間隙を表す。単一の間隙を特殊な場合として含むことを強調するとき，間隙・範囲と呼ぶ。(i，j)が範囲であることを強調するために[i，j]という表記法を用いる。

備考　この表記法は，説明のために導入したもので，規格の一部ではない。

6.1.5 範囲に対応する部分テキスト体　範囲[i，j]に対して，iより右にあり，jより左にあるすべての構成素からなる部分テキスト体が，自然に対応する。これを範囲部分テキスト体という。範囲[i，j]に対応する部分テキスト体を，

    subtext([i，j]，T)

と書く。T = C1C2..Cn なら，

    subtext([i，j]，T) = Ci+1...Cj

i = j のときは，

    subtext([i，j]，T) = 空テキスト体

となる。

6.1.6 テキスト体の結合　テキスト体Tの後にテキスト体Sを並置(concatenation)して得られるテキスト体を，T及びSの結合と呼ぶ。T及びSの結合を

    concat(T，S)

又は

    T + S

と書く(+は，連結を表す中置演算子)。

6.2 テキスト体編集関数

6.2.1 編集関数及び対象テキスト体　一つのテキスト体引数に加えて，その他(任意に)，間隙・範囲，テキスト体を引数として，テキスト体の値を返す関数を，テキスト体編集関数という。混乱の恐れがなければ，単に編集関数と呼ぶ。

編集関数は，一つのテキスト体引数を特別扱いする。このテキスト体引数を編集対象テキスト体と呼ぶ。単に対象テキスト体とも呼ぶ。

次の二つの編集関数を，基本編集関数と呼ぶ。

(1) delete　範囲[i，j]，対象テキストS ---> 結果テキスト

(2) insert　間隙i，挿入テキストX，対象テキストS ---> 結果テキスト

delete及びinsertの定義は，範囲部分テキスト体及び結合を使って，次のとおりに書ける。nは，テキスト体Sの末尾に位置する間隙の間隙番号とする。

```
delete([i, j], S) = subtext([0,i], S) + subtext([j, n], S)
insert(i, X, S) = sub([0, i], S) + X + sub([i, n], S)
```

deleteを削除関数と呼び，insertを挿入関数と呼ぶ。その引数を，削除範囲，挿入位置間隙，挿入テキスト体と呼ぶ。

備考　削除の対象は，範囲に対応する部分テキスト体とする。範囲は，部分テキスト体を表すために使われる。しかし，範囲及び部分テキス体は，習慣的に同一視されるので，“削除範囲”という。

6.2.2 構成素出現及び間隙の追跡　削除関数及び挿入関数を適用すると，結果のテキスト体においては，構成素及び間隙の番号付けが変化する。関数適用の前後における構成素及び間隙の同一性は，次の規則によって判定する。

(1) 挿入されたテキスト体の右に新たな間隙が生じる。この新間隙より右の構成素出現及び間隙には，挿入関数適用前よりkだけ増えた番号が付けられる。kは，挿入されたテキスト体の長さとする。

(2) 削除された範囲部分テキスト体の右の間隙も同時に削除される。削除された部分より右の構成素出現及び間隙には，関数適用前よりkだけ減った番号が付けられる。kは，削除されたテキスト体の長さとする。

構成素出現n，間隙iに対して，この規則に従って新たに付けた番号を，それぞれn'，i'とする。関数の適用の前後において，構成素出現n及びn'，間隙i及びi'は，同一であるとみなす。

複数回の関数の適用においても，この同一性の判定を繰り返せば，構成素及び間隙を追跡できる。構成素及び間隙は，削除によって無くならない限り，その番号が変わっても，テキスト体において同一性を保って存在する。

6.2.3 編集関数の同時適用可能性　二つの基本編集関数を，対象テキスト体に同時に適用することを，順次適用をもとに，次のとおり定義する。

(1) 第二の関数が挿入の場合　第一の関数適用後に，挿入位置間隙を追跡する。挿入位置間隙が削除されて存在しないなら，同時適用は不可能とする。挿入位置間隙が追跡できれば，その挿入位置間隙に対して挿入を実行する。

(2) 第二の関数が削除の場合　第一の関数適用後に，削除範囲の範囲部分テキスト体のすべての構成素を追跡する。一つでも構成素が削除されていれば，同時適用は不可能とする。削除範囲部分テキスト体が追跡できれば，その部分テキスト体を削除する。

複数の基本編集関数に対して，どの二つも同時適用が可能なら，それらの関数の集

合は同時適用可能という。

6.3 テキスト族

6.3.1 テキスト及びテキスト族　名前が付けられたテキスト体をテキストという。テキスト体として同じでも，名前が異なれば違うテキストとみなす。テキストに付けられた名前をラベルという。

備考　“テキストのテキスト体”を，しばしば単に“テキスト”と呼ぶ。例えば，テキストのテキスト体が空であることを，“テキストが空である”と表現する。又は，テキストのテキスト体の部分テキスト体を，“テキストの部分テキスト体”と表現する。

テキストの有限集合をテキスト族と呼ぶ。同一のテキスト族に属するテキストは，すべて違うラベルをもたなければならない。

備考　名前(ラベル)は，文字列で表現される識別子であり，ラベルについては，それ以外の規定は設けない。

6.3.2 テキスト族の操作　テキスト族に，新しく空のテキスト(そのテキスト体が空テキスト体であるテキスト)を追加する操作を，テキスト生成と呼ぶ。追加されたテキストには，新しいラベルが与えられる。

テキスト族に所属するテキストをテキスト族から除外する操作を，テキスト消去と呼ぶ。テキスト族から一つのテキストが消去されても，他のテキストには影響がない。

6.4 テキストリンク

6.4.1 テキストリンクの定義　テキスト中の間隙・範囲又はテキストそのものに，他のテキスト(同一のテキストでもよい)の間隙・範囲又はテキストそのものを対応付けることをテキストリンクと呼ぶ。混乱の恐れがなければ，テキストリンクを単にリンクとも呼ぶ。リンクにより関係付けられる二つのテキストを，リンク元テキスト，リンク先テキストと呼ぶ。

リンクは，次の要素によって決定される。

(1) リンク元テキスト
(2) リンク元テキストの間隙・範囲又はリンク元テキストそのもの(特定の範囲を指定しない。)
(3) リンク先テキストの間隙・範囲又はリンク先テキストそのもの(特定の範囲を指定しない。)
(4) リンク種別
(5) リンクに付けられた名前(ラベル)

参考　ここで定義したリンクは，ハイパテキスト応用を意図していない。単一の線形テキストでは表現しにくい，複雑な文書の構造を表現する手段として，リンクを用いる。

リンクは，名前によって一意に識別される。この名前を，リンクラベルと呼ぶ。リンクラベルが違えば，他の要素がすべて同じでも，異なるリンクとみなす。リンクには，リンクの意味，目的，役割，機能などを示すリンク種別が付随する。

リンク元テキストに対し指定された間隙・範囲又はテキストそのものを，リンク元

という。リンク先テキストに対し指定された間隙・範囲又はテキストそのものを，リンク先という。
　リンクは，同一のテキスト族内で考える。つまり，リンク元テキスト及びリンク先テキストは，同一のテキスト族に所属する。

6.4.2 テキストリンクの操作　テキストリンクの操作として，次の二つがある。
(1) リンクの追加　既に存在する二つのテキスト間に関係を付けることとする。テキスト自体は一切変更しない。
(2) リンクの消去　リンク元とリンク先との関係をなくす。しかし，テキストそれ自体には，全く変更が加えられない。テキストリンクが消去されると，二つのテキストは無関係となるが，同一のテキスト族内に依然として存在する。

6.4.3 間接的なテキストリンク操作　テキスト体編集関数，テキスト族操作によって，間接的にリンクが消去されることがある。例えば，リンク元テキスト又はリンク先テキストが，テキスト族からテキスト消去で存在しなくなった場合，リンクは存続し得ない。テキスト体の編集により，リンク元又はリンク先が変動した場合，そのリンクが存続するか消去するかは，規定しない。応用の要求により，適切な処理を行うものとする。

6.5 対象及び操作　この規格は，ある時点の文書を表現するために，次の対象を用いる。
(1) テキスト族
(2) テキスト族に所属するテキスト
(3) テキスト族に所属するテキスト間のテキストリンク
　テキスト族内では，各テキストはラベルで識別できる。各リンクもまたラベルで識別できる。これらの対象に対して行える操作は，テキスト及びリンクに関する生成・消去，並びに各テキストのテキスト体に対する編集関数とする。
　各操作及びその引数は，次のとおりとする。
(1) テキスト生成 ： 生成された新しいテキストに付けるラベル。
(2) テキスト消去 ： 消去対象のテキストを識別するラベル。
(3) リンク追加 ： リンク元テキストを識別するラベル，間隙・範囲又はテキストそのもの，リンク先テキストを識別するラベル，間隙・範囲又はテキストそのもの，リンク種別，及びリンクのラベル。
(4) リンク消去 ： 消去対象のリンクのラベル。
(5) 挿入編集関数 ： 編集対象のテキストを識別するラベル，挿入位置間隙を識別する間隙番号，及び挿入テキスト。
(6) 削除編集関数 ： 編集対象のテキストを識別するラベル，及び削除範囲を指定する間隙の組。

７．文書変更構造

7.1 変更指令

7.1.1 変更指令の定義　テキスト族に対する一連の操作(の指示)が，変更指令であって，次の操作を含む。
(1) [KIHON]　テキスト族，そのテキスト及びリンクに対する次の基本操作。
- 所属するテキストに対する挿入及び削除の編集関数
- テキスト族におけるテキスト生成及びテキスト消去
- テキスト族におけるリンクの追加及びリンクの消去

(2) [IKI] 取消し指令。取消し指令は，変更指令中の他の部分を無効化する。
(3) [JUNJI] e1，...，eNが，N個の変更指令であるとき，それらの順次実行を示す形式を，変更指令とする。説明のために，{e1，e2，...，eN}によって順次実行を示す。この形式を順次実行指令という。
(4) [DOUJI] e1，...，eNが，N個の変更指令であり，すべてが同時実行可能なとき，それらの同時実行(並列実行)を示す形式を，変更指令とする。説明のため，{e1 & ...& eN}によって同時実行を示す。この形式を同時実行指令という。

参考 つまり，順序を守って順次実行される操作の列が，順次実行指令であり，同時に実行することが可能な，互いに競合しない操作の集合が，同時実行指令である。取消し指令は，“イキ”の定式化である。

項目[KIHON]で定義される指令を基本指令，項目[JUNJI]及び項目[DOUJI]によって定義される指令を複合指令と呼ぶ。複合指令は，いくつかの基本指令，取消し指令，又は他の複合指令をグループ化したのものとする。
変更指令が複合指令であるとき，その複合指令を構成しているそれぞれの変更指令を，指令構成子と呼ぶ。単一の基本操作からなる変更指令の場合には，その基本操作自体を指令構成子とする。

7.1.2 指令構成子の一意識別子及び有効性標識 一つの変更指令において，それを構成するすべての指令構成子は，番号，名前などの一意識別子をもつ。必要があれば，この識別子によって，変更指令中の任意の指令構成子を指し示すことができる。

備考 指令構成子の識別子は，取消し指令のために使用される。実際にすべての指令構成子に識別子を割り当てる必要はない。取消し指令が指す必要がある指令構成子だけに，識別子を割り当ててあればよい。

取消し指令は，その取消し指令が出現した変更指令中の指令構成子を参照する引数をもつ。
変更指令中のすべての指令構成子は，“有効”又は“無効”の値をもつ有効性標識をもつ。無効とマークされた指令構成子は，実行のときに無視される。

7.1.3 変更指令の実行 変更指令の実行は，次の規則に従う。
(1) テキスト族の操作の実行に先立って，取消し指令をすべて解釈する。このとき，後に出現した取消し指令が優先される。したがって，取消し指令の解釈の段階では，最後の指令から最初の指令に向かって走査する。取消し指令が指し示す指令構成子の有効性標識値を“無効”に変更(もともと“無効”であればそのまま)する。無効とされた指令構成子の内部は，それ以上走査しない。
(2) 同時実行指令は，それを同時に実行したかのように実行する。順次実行によって同時実行を模擬するには，次のとおりとする。
- 同時実行指令の指令構成子を任意の順序に並べる。
- 最初の指令構成子を実行する。
- 最初の指令構成子による影響を追跡し，2番目の指令構成子内で使われているテキスト構成素出現，間隙を補正した後で，2番目の指令構成子を実行する。
- この過程を繰り返す。

有効性標識値が“無効”である指令構成子は，無視して実行しない。
(3) 順次実行指令は，その指令構成子を指定された順序で実行する。有効性標識値が“無効”である指令構成子は，無視して実行しない。

(4) 基本操作は，その定義に従って実行する。有効性標識値が“無効”であれば，無視して実行しない。

7.1.4 変更指令の合成　二つの変更指令e1，e2があるとき，その二つから構成される順次実行指令{e1，e2}を，この二つの変更指令の合成ともいう。複数の変更指令の列 e1，e2，... ，eN に関しても同様に，順次実行指令{e1，e2，...，eN}をそれらの合成という。

7.2 版集合及び文書変更構造

7.2.1 版集合　番号又は名前の集合があり，その番号又は名前に(一部分)順序が存在するとき，版集合と呼ぶ。版集合の要素は，版ラベル又は単に版と呼ぶ。版ラベル間の順序を表すために，通常の不等号を使う。しかし，数の順序とは異なる。

参考　版集合は，数学的には半順序集合である。版集合は，文書の版，バージョン，エディションなどを識別するラベルの集合であり，時間，版の分岐などを反映する順序が導入されている。

7.2.2 文書変更構造の定義　版集合のいくつかの版ラベルに対して，テキスト族を対応させ，いくつかの版ラベルの順序のある対に変更指令を対応させたものを文書変更構造と呼ぶ。これは，次の条件を満たさなくてはならない。

(1) 版集合の任意の版ラベルは，次のいずれかとする。
- その版ラベルにテキスト族が対応している。
- 版ラベルの列 r0 < r1 < ... < rN があり，r0にはテキスト族が対応し，(r0，r1)，(r1，2)などには変更指令が対応している。rNは，当該の版ラベルとする。

(2) 版ラベルr0にはテキスト族が対応し，版ラベルの列 r0 < r1 < ... < rN の各(r0，r1)，(r1，r2)には変更指令が対応し，版ラベルrNにはテキスト族が対応しているとき，r0のテキスト族に対し，変更指令の列を順次適用した結果は，rNに対応しているテキスト族に等しい。

(3) 版ラベルxから版ラベルyに向かう二つの列 x = r0 < r1 < ... < rN = y ,x = s0 < r1 < ... < sM = y があるとき，r0，r1，... ，rNの合成と，s0，s1，... ，sMの合成とは，同一の効果をもつ変更指令とする。

参考　これらの条件は，次の意味をもつ。
- すべての版ラベルにテキスト族を対応させることが，可能である。
- テキスト族の対応付け及び変更指令の対応付けは，一貫している。
- テキスト族を求める方法が複数あっても，矛盾しない。

与えられたテキスト族及び変更指令から，変更指令の合成実行により，矛盾無く，すべての版のテキスト族を求めることができる。

備考　標準原稿校正マーク付け言語(SPML)は，ここで定義した文書変更構造を記述する言語である。

7.2.3 変更指令属性及び実行時の検査　変更指令と，そのすべての指令構成素には，いくつかの属性をもたせることができる。これらの属性を変更指令属性と呼ぶ。変更指令一意識別子及び有効性標識は，変更指令属性とする。その他に，次の属性を推奨する。

(1) editor　その変更を行った者又は責任をもつ者

(2) time　その変更を行った時刻
(3) description　その変更に関する記述又は注釈
(4) priority　その変更の重要度，優先順位
　応用の要求により，この他の属性を追加してもよい。変更指令の実行に先立って，これらの変更指令属性の値から，変更の可否又は正当性を検査してもよい。

備考　変更に関連する注釈などは，属性としてではなく，テキスト族へのテキスト及びリンクの追加によっても表現できる。これは，編集・校正時に，メモ又は付箋を添付することに相当する。

８．文書型定義

標準原稿校正マーク付け言語(SPML)を，標準一般化マーク付け言語(SGML)の文書型定義(DTD)の要素集合として規定する。

8.1 レビジョンセット

リビジョンセットの元(のラベル)を列挙するために，revison要素型を用いる。リビジョンセットの一つの元に対し，一つのrevision要素を記述する。order要素によって存在が暗示される元は，省略してもよい。

　リビジョンセットの順序関係を記述するために，order要素型を用いる。元xが元yより真に小さい組(x, y)に対して，<order le=x gr=y>を記述する。

```
<!element revison-set - - (revision-set-desc?,
        (revison|revision-desc)*, (order|order-desc)*)
       -- OrderedSet: -->
<!element revison - O EMPTY>
<!attlist revsion
       label %RevisionLabel; #REQUIRED -- RevesionLabel: --
>

<!element order - O EMPTY>
<!attlist order
       le %RevisionLabel; #REQUIRED -- RevisionLabel1: --
       gr %RevisionLabel; #REQUIRED -- RevisionLabel2: --
>
```

例：

```
<revision-set>
<revision-set-desc>四つのリビジョンからなるセット</>
<revision label=1>
       <revision-desc label=1>初期リビジョン</>
<revision label=1a>
```

```
        <revision-desc label=1a>A氏による変更</>
<revision label=1b>
        <revision-desc label=1b>B氏による修正</>
<revision label=2>
        <revision-desc label=2>完成</>
<order le=1 gr=1a>
<order le=1 gr=1b>
<order le=1a gr=2>
<order le=1b gr=2>
</revision-set>
```

ここで，OrderdSetは，順序集合又はその一部を記述しなくてはならない。RevsionLabelは，前もって決めたラベルの領域に属さなくてはならない。RevisionLabel1及びRevisionLabel2は，異なるラベルでなくてはならない。

8.2 テキストリンク型セット {{テキストリンク}の型}(種別)に付ける名前を宣言する。ただし，テキストリンクの意味が{処理系}，{交換の当事者}に理解されている場合は，省略してもよい。処理・交換に不要な部分は，省略してもよい。特に，対象とする{テキスト族}内にテキストリンクが存在しない場合は，この宣言は不要である。

```
link-type-set
link-type
LinTypeLabel

<!element link-type-set - - (link-type-set-desc?,
        (link-type|link-type-desc)*
>

<!element link-type - O EMPTY>
<!attlist link-type
        label %LinkTypeLabel; #REQUIRED
>
```

8.3 テキスト族セット family-set要素型及びその部分要素型は，テキスト族を記述する。テキストリンクの名前だけを宣言しておいて，実際のリンクの記述を省略し

てもよい。他の方法でリンクを記述できる場合は，link要素を使う必要はない。

この宣言は，対象となるテキスト族のラベル(名前)，それぞれのテキスト族に含まれるテキストのラベル及びリンクのラベルを列挙することを目的とする。処理及び交換に不要な部分の記述は，省略してもよい。

```
family-set
family
text
link
FamilyLabel
TextLabel
LinkLabel

<!element family-set - - (family-set-desc?,
        (family)* >
<!element family - - (family-desc?,
        (text|text-desc)*, (link|link-desc)* >
<!attlist family
        label %FamilyLabel; #REQURED
>

<!element text - O EMPTY>
<!attlist text
        label %TextLabel; #REQUIRED
>

<!element link - O EMPTY>
<!attlist link
        label %LinkLabel; #REQUIRED
        source %TextLabel;
        start  NUMBER
        end    NUMBER
        target %TextLabel;
>
```

8.4 テキスト族とリビジョンとの対応　リビジョン及びテキスト族の対応関係を

記述するために，次の要素型を用いる。

revision-family-map
map

```
<!element revision-family-map - - revision-family-map-desc?,
        (map|map-desc)*>
<!element map 0 EMPTY>
<!attlist map
        revision %RevisonLabel; #REQUIRED
        family   %FamilyLabel; #REQUIRED
>
```

例：

```
<revision-family-map>
<map rev=1  family=initial>
<map rev=1a family=A-reviced>
<map rev=1b family=B-modified>
<map rev=2  family=final>
</revision-family-map>
```

8.5 テキスト内容表現式　テキスト族内のテキストの部分テキストを表すために，次の要素型を用いる。subtextはinsertの内容として用いる。

subtext

```
<!element subtext - 0 EMPTY>
<!attlist subtext
        family %FamilyLabel; #IMPLIED
        text   %TextLabel; #IMPLIED
        start NUMBER #REQUIRED
        end   NUMBER #REQUIRED
>
```

```
<subtext text=1 from=10 to=235>
```

8.6 アクション(テキスト編集)

insert
delete
TextContentModel

```
<!element insert - - %TextContentModel; >
<!attlist insert
        id ID #IMPLIED
        effect (effective|canceled) effective
        family %FamilyLabel;
        text  %TextLabel;
        at    NUMBER
>

<!element delete - 0 EMPTY>
<!attlist delete
        id ID #IMPLIED
        effect (effective|canceled) effective
        family %FamilyLable;
        text   %TextLabel;
        from   NUMBER #REQUIRED
        to     NUMBER #REQUIRED
>
```

例：

```
<insert id=i021
        effect=effective
        family=initial
        text=initial
        at=123>
```

8.7 アクション(テキスト族の操作)

new-text
kill-text

```
<!element new-text - 0 EMPTY>
```

```
<!attlist new-text
        id ID #IMPLIED
        effect (effective|canceled) effective
        family %FamilyLabel; #IMPLIED
        text   %TextLabel;   #IMPLIED
>

<!element kill-text - 0 EMPTY>
<!attlist kill-text
        id ID #IMPLIED
        effect (effective|canceld) effective
        family %FamilyLabel; #IMPLIED
        text   %TextLabel;   #REQUIRED
>
```

8.8 アクション(テキストリンクの操作)

```
new-link
kil-link

<!element new-link - 0 EMPTY>
<!attlist new-link
        id ID #REQUIRED
        effect (effective|canceled) effective
        family %FamilyLabel; #IMPLIED
        label  %LinkLabel; #IMPLIED
        type   %LinkTypeLable: #IMPLIED
        source %TextLabel; #REQUIRED
        start NUMBER #REQUIRED
        end   NUMBER #REQUIRED
        target %TextLabel; #REQUIRED
>

<!element  kill-link - 0 EMPTY>
<!attlist  kill-link
        link %LinkLabel; #REQUIRED
>
```

附属書１.(参考)　SPML文書記述例

簡単ではあるが，完全なSPML文書記述例をここに示す。この例には，二つのリビジョンがあり，それらは1st，2ndでラベル付けられている。つまり，リビジョンセットは，{1st， 2nd}であり，順序は，1st < 2nd である。

```
<revision-set>
  <revision label='1st'>
  <revision label='2nd'>
  <order le='1st' gr='2nd'>
</revision-set>
```

リビジョン1st，2ndに対応する二つのテキスト族は，First，Secondでラベル付けられている。Firstには，二つのテキストhi及びallが含まれ，Secondには，一つのテキストhelloが含まれる。

```
<family-set>
  <family label=First>
    <text label=hi class=string>
    <text label=all class=string >
  </family>
  <family label=Second>
    <text label=hello class=string>
  </family>
</family-set>

<revision-family-map>
  <map revision='1st' family=First>
  <map revision='2nd' family=Second>
</revision-family-map>
```

編集操作を行う人をjiroとする。

```
<editor-set>
  <editor label=jiro key='TAKAHASHI Jiro'>
</editor-set>
```

二つのテキスト族に含まれる三つのテキストは，次の内容であるとする。

hi :こんちには
all:みなさん
hello:こんにちは，皆さん。

テキストhi及びallの間にテキストリンクが存在する。このリンクはcontinueというラベルをもち，nextという種別をもつ。

```
<link-role-set>
  <link-role label=next>
</link-role-set>

<link label=continue
  family=First
  source=hi
  start=0
  end=0
  target=all
  class=next>
```

テキスト族Firstからテキスト族Secondへの変更を，次の手順で行う。

1. テキストhiの“ち”及び“に”を交換する。
2. hi及びallを一つのテキストにまとめる。
3. “みな”を“皆”に置き換える。
4. “，”及び“。”を挿入する。

```
hi : こ ん ち に は
     ^  ^  ^  ^  ^  ^
     0  1  2  3  4  5

all: み な さ ん
     ^  ^  ^  ^  ^
     0  1  2  3  4
```

(1)“に”を挿入。

```
<insert
   text=hi
   addressing=standard
   at=2><subtext text=hi from=3 to=4></insert>

   hi : こ ん に ち に は
        ^  ^  ^  ^  ^  ^  ^
        0  1  2  3  4  5  6
```

(2) 不要な“に”を削除。

```
<delete
  addressing = standard
  text=hi
  from=4 to=5>

   hi : こ ん に ち は
        ^  ^  ^  ^  ^  ^
        0  1  2  3  4  5
```

(3) allのすべてのテキストhiの末尾に挿入。

```
<insert
  text=hi
  at=5><subtext text=all from=0 to=4></insert>

   hi : こ ん に ち は み な さ ん
        ^  ^  ^  ^  ^  ^  ^  ^  ^  ^
        0  1  2  3  4  5  6  7  8  9
```

(4) リンクを消去し，不要なallも消去する。

```
<kill-link link=continue>
```

<kill-text text=all>

(5) hiをhelloにリネームする。

<new-text label=hello>
<insert text=hello at=0><subtext text=hi from=0 to=9></insert>
<kill-text text=hi>

(6)“みな”を“皆”に置き換える。

<delte text=hello from=5 to=7>
<insert text=hello at=5>皆</insert>

(7) 句読点を挿入する。

<insert text=hello at=5>, </insert>
<insert text=hello at=9>。</insert>

```
 hello: こ ん に ち は ,  皆 さ ん 。
        ^  ^  ^  ^  ^  ^  ^  ^  ^  ^
        0  1  2  3  4  5  6  7  8  9
```

これまでの要素をまとめると，次の例となる。

<document-changes>
<document-changes-desc>
例
</document-changes-desc>

<head>
<title>文書変更例</title>
<date>1997 1/20</date>
<creater>HIYAMA Masayuki</creater>
<application-info application=HIYAMA>
No application info
</application-info>

```
</head>

<revision-set>
<revision-set-desc>
二つのリビジョン
</revision-set-desc>
  <revision label='1st'><revision-desc></>
  <revision label='2nd'><revision-desc></>
  <order le='1st' gr='2nd'><order-desc></>
</revision-set>

<link-role-set>
  <link-role label=next>
  <link-class-desc>
二つのテキストの順序を示す。リンク元が先行するテキスト，リンク先が後続するテ
キストを示す。リンク元範囲は，常に 0;0 とする。
</link-class-desc>
</link-role-set>

<editor-set>
  <editor label=jiro key='TAKAHASHI Jiro'
   x-ref = "http://foo.co.jp/pserson/Jiro/"
   >
</editor-set>

<text-calss-set>
  <text-class-decl><text-class label=char-string>
  <text-class-decl>
構成素は文字である。最も一般的なテキストである。
  </text-class-decl>
</text-class-set>

<addressing-set>
 <addressing-decl><addressing label=standard></addressing-decl>
<addressing-set>
```

```
<priority-set>
  <priority-decl><priority label=normal></priority-decl>
  <priority-decl><priority label=important></priority-decl>
  <order le=normal gr=important>
</priority-set>

<priority-range>
 <discrete min=0, max=100, step=1>
</priority-range>

<priority-range>
 <continuous min=0, max=1>
</priority-range>

<date-format label="C asctime">
<date-format-desc>
<date-format-info></>
</date-format>

<family-set>
<family-set-desc>
</family-set-desc>
  <family label=First>
    <text label=hi>
    <text label=all>
  </family>
  <family label=Second>
    <text label=hello>
  </family>
</family-set>

<revision-family-map>
  <map revision='1st' family=First>
  <map revision='2nd' family=Second>
</revision-family-map>
```

```
<link label=continue
  family=First
  source=hi
  start=0
  end=0
  target=all
  role=next>

<!-- REVICE SET -->

<revice-set>
<revice
  source-rev='1st'
  target-rev='2nd'

  editor=jiro
  date='1997 1/20 12:30:11'
  priority=normal
>
<sequence>
<sequence id=collect
  priority=important
>
<!-- 間違いの修正 -->
<insert
   priority=important
   text=hi
   at=2><subtext text=hi from=3 to=4></insert>
<delete
  priority=important
  text=hi
  from=4 to=5>
</sequence>

<sequence id=merge>
```

```
<!-- 一つのファイルに -->
<insert
  text=hi
  at=5><subtext text=all from=0 to=4></insert>
<kill-link link=continue>
<kill-text text=all>
</sequence>

<sequence id=rename>
<!-- helloにリネーム -->
<new-text label=hello>
<insert text=hello at=0><subtext text=hi from=0 to=9></insert>
<kill-text text=hi>

<!-- “みな”を“皆”に -->
<sequence id=tokanji>
<delte text=hello from=5 to=7>
<insert text=hello at=5>皆</insert>
</sequence>

<!-- 句読点を挿入 -->
<sequence id=punct>
<insert text=hello at=5>, </insert>
<insert text=hello at=9>。</insert>
</sequence>
</sequence>
</revice>

</document-changes>
```

附属書２.(参考)　SPML編集システムの指針
従来の校正指示とこの規格の命令関数との関係

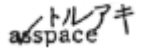
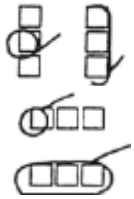
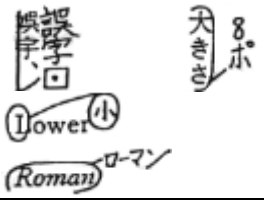
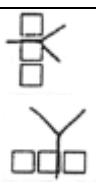
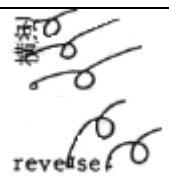
JIS Z 8208の表1に示された校正記号について，この規格で規定している言語で記述した例を附属書2表1に示す。コメント欄の番号は，次の記述に対応する。

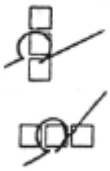
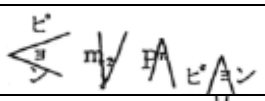
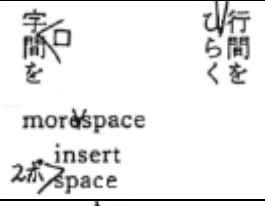
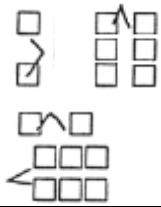
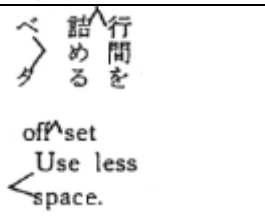
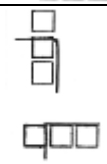
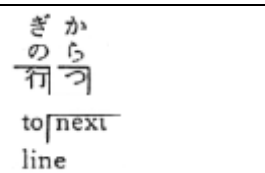
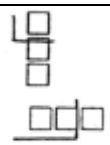
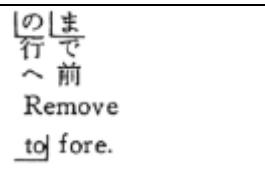
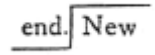
1) この規格が対象とする電子的な文書においては起こり得ない誤まりに対する校正指示なので，対象外とする。

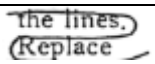
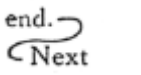
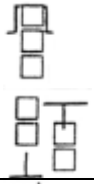
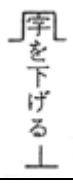
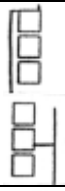
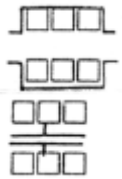
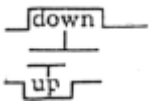
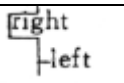
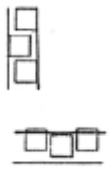
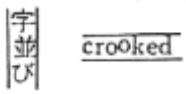
2) フォーマタ若しくはレンダリング系に関する指示，又は出力体裁の誤りを校正している。フォーマタ又はレンダリング系への指示を変更することによって，この規格の範囲内で対応可能であるが，　ここでは扱わない。

3) 本来は体裁に関する校正指示であるが，この例では，文字列データに対する編集として扱う。

附属書 2 表 1　JIS Z 8208 表 1 と本規格との対応

番号	記号	意味	使用例		
			訂正前	訂正後	
1. 1		文字、記号などをかえ、または取り去る。		誤字・	テキストの内容を「娯 <delete from=0 to <insert at=0>誤字
				剰字	テキストの内容を「剰 <delete from=1 to
				a space	テキストの内容を「as <delete from=1 to <insert at=1>␣</ ␣は空白 1 つ分を表
				revive	テキストの内容を「re <sequence id=i1> <delete from=1 to <insert at=1>i </sequence> <cancel ref=i1>
1. 2		書体または大きさなどを変える。		大きさ 誤字・ lower Roman	
1. 3		字間に文字、記号などを入れる。		抜け字	テキストの内容を「抜 <insert at=1>け</
				字・記	テキストの内容を「字 <insert at=1>・</i
				omission	テキストの内容を「om <insert at=4>si</
				comma, or	テキストの内容を「co <insert at=5>,</i
1.4		転倒した文字、記号などを正しくする。		逆横。 reverse ;	

1.5		不良の文字、記号などをかえる。		欠字を broken	
1.6		右付き、上付きまたは下付きにする。		ピョン	テキストの内容を「ピ <delete from=1 to <insert at=1>ョ</i
				ピョン m' Pn ピョン	
1.7		字間、行間などを空ける。		字間を空ける 行間を広らく more space insert space	
1.8		字間、行間などを詰める。		字間を詰める 行間を詰める offset Use less space.	
1.9		つぎの行へ移す。		つぎの行から to next line	
1.10		前の行へ移す。		前の行まで Remove to fore.	
1.11		新しく行を起こす。		別す。	テキストの内容を「す。 <insert at=2>&# 改行は、「&#RE;&#
				end. New	テキストの内容を「en <insert at=5>&# 改行は、「&#RE;&#
1.12		文字、行などを入れ替える。		転倒を	テキストの内容を「倒 <insert at=0><su <delete from=2 to
				入れかえる。	テキストの内容を「え <insert at=0><su

					<delete from=5 to
				transfer	テキストの内容を「tr <insert at=2><su <delete from=4 to
				Replace the lines.	テキストの内容を「th <insert at=0><su <delete from=18
1.13		行をつづける。		る。 行	テキストの内容を「す。 <delete from=2 to 改行は、「&#RE;&
				end. Next	テキストの内容を「en <delete from=5 to 改行は、「&#RE;&
1.14 1.15		指定の位置まで文字、行などを移す。		字を下げる	テキストの内容を「字 <insert at=0> ㄴ ㄴは空白 1 つ分を表
				右へ行を移す	
				down up	
				right left	テキストの内容を「字 <insert at=0>ㄴㄴ ㄴは空白 1 つ分を表
1.16		字並びなどを正しくする。		字並び crooked	
1.17		(欧文)大文字にする。		CAPITAL	テキストの内容を「Ca <delete from=1 to <insert at=1>API

	□□□		capital	Capital	テキストの内容を「ca <delete from=0 to <insert at=0>C</
1.18	□□□□ □□□	(欧文)スモールキャピタル体にする。	small Small	SMALL SMALL	
1.19	□□□□ □□□	(欧文)イタリック体にする。	italic n kW	*italic* *n* kW	
1.20	□□□□	(欧文)ボールド体にする。	bold bold	**bold** bold	

原稿校正標準マーク付け言語(SPML)
解説

１．制定の趣旨　文書作成の重要な工程として校正がある。商業出版における校正作業は通常，著者，編集者及び校正者によって行われ，校正指示は現在も主に紙媒体を中心として，これら複数者間で各々の役割分担で交換されている[1]。商業出版だけでなく，企業内などの一般文書の作成作業においても，権限を異にする幾人ものレビュー担当者の間で原稿が交換され，それぞれの立場で校正指示が施されている。校正は，商業出版だけでなく一般の文書の作成においても行われる作業であり，共通の指示内容と言える。

文書作成の電子化が普及した結果，作成途上の原稿を紙媒体を用いずに電子文書として交換し，校正作業をネットワークを介して行うことへの要求が高まっている。しかし電子文書に対する校正指示の規格はなく，原稿（文書）作成作業の効率化を妨げていた。

校正作業で用いられている主な機能は，次のとおりである。

(1) 文書の内容の挿入，削除，移動，入れ替えを行うテキスト編集
(2) 内容の変更の履歴を管理するバージョン管理
(3) 幾人かのグループでの作業において，著者と編集者との間，上司と部下との間などで編集権限が異なる場合に，その権限を制御するアクセス制御
(4) 特定又は不特定の対象にあててのメモ書き

これらの校正作業を，効率的かつ正確に実行するため，電子化された原稿に対して，処理系に依存しない標準的な原稿校正マーク付けを行い，それを校正原稿情報として交換することが望まれていた。

２．制定の経緯

校正作業の電子化及び校正対象原稿の交換の必要性を認識した通商産業省工業技術院は，校正対象原稿の交換を可能にするための規格の調査研究を，1996年3月に国際大学グローバルコミュニケーションセンターに対して委託した。これを受けて，同センターはSGML文書校正マーク付け言語の提案型国際規格作成調査研究委員会を設立し，1996年4月から調査研究を開始した。

調査研究に際しては，諸外国の専門家との情報交換を行って，開発する規格原案をそのまま国際規格としても利用可能にすることに配慮した。校正原稿標準マーク付け言語(SPML)の規格原案は，委員会報告書に含めて1997年3月に工業技術院に提出された。

３．審議中の主要検討課題

**** TBD ****

４．適用範囲の補足

4.1 SPMLに対する利用者要求　SPMLに対して求められる主な利用者要求は，次のとおりである。

(1) 校正操作として，挿入及び削除の機能をもつ。
(2) 校正対象テキストと挿入テキストとの区別ができる。
(3) 削除範囲及び挿入位置を記述できる。
(4) 複数の校正対象テキストを識別できる。
(5) テキスト中のある範囲又はある位置に他のテキストを対応付けることができる。
(6) 校正操作を複数回適用した場合の変更履歴を記録できる。
(7) 編集指示属性をもつ。

4.2 従来の出版物の校正

4.2.1 校正の機能　出版を前提にした校正は，次の3機能をもっている。
(1) 原稿と組版結果との照合による加除訂正。
(2) 組版レイアウト情報の確認結果による訂正。
(3) 原稿自体の訂正。
(1),(2)は本来の校正機能であり，(2)は，“ 書体 ”“ 文字サイズ ”“ 字間・行間 ” などの指定に対する確認である。一方，たとえば “ 新旧仮名遣いの統一 ”“ 送り仮名の間違い訂正 ”“ 使用漢字の制限 ”“ ルビ又は注の付加 ” など，読者に対する考慮又は出版社のポリシーによって，元の原稿が直されることも多い。(3)の機能は，この場合に用いられ，これも一つの “ 校正 ” である。

(3)の機能が用いられる場合については，以前から原稿自体に起因する問題が多くある。“ 理想的にいうと，原稿の文字は正しく書かれていて，しかも文章にはあいまいなところがなく，意味のよく通っていることが望まれる。しかし，こうした条件を欠くものが多く，校正者を悩ます結果となっている ” という報告がある[2]。

最近の校正はさらに踏み込んだものになりつつあるとの指摘もある。たとえば，直木賞を受賞した “ 凍える牙 ” の作家，乃南アサ氏は，新潮社から戻ってきた初校ゲラをみて，“ 顔から血の気が引いた ” と語る。“ 誤字はもとより，表現不適切さ，物語全体の流れ，描写の過不足，その他ありとあらゆる点の手厳しい意見が大き目の付箋にびっしり書かれ ” ていた（96.10.7付け読売新聞）とのことである。

通常は，(1)及び(2)は，組版作業上のミスとして扱うことのできるものであるが，(1)の場合，校正履歴を残すことはほとんど意味がないのに対して，(2)は必ずしも残さなくてよいとは言えない。(3)のケースでは，履歴を残すことは必須である。校正の仕事という面からみれば，“ 最近の編集者は本の企画作りに追われ，ゲラを読み込む暇がない。その分，仕事が構成者にシフトするようになったようだ ”（講談社校閲局長・中村剛士）[2]との指摘に見られるとおり，仕事の幅が広がっている。

実際に “ 電子校正 ” を考えるに当たって，これらの管理を別にできるようにするか又は一緒にするかについては，重要な検討課題である。

4.2.2 校正の基本　校正は，必ず参照対象との照合という作業形態をとる。この参照対象は，たとえば前述の(1)では“ 原稿 ”，(2)では“ 組版レイアウト指示書 ”，(3)では“ 辞典類 ”となる。閲読では，校閲者の感性を参照することになる。

従来の校正作業は，参照対象と被校正物とを並べて，これらを比較しながら訂正個所を特定し，被校正物の上に，赤で校正指示をする。校正の仕方は，各出版社がそれぞれ規定し，出版社と印刷会社との間にも取り決めをして，それらに基づいて校正作業をする。

校正作業に用いる校正記号は，JIS Z 8208“ 印刷校正記号 ”に規定されているが，これは，活字時代に規定されたものがベースになっていて，必ずしも電子組版環境に相応しいものではない。校正記号については，JISに見られるとおり，必ずしも厳密な指示が規定されていない。ある文字を削除する指示を，その該当文字から引出し線を出して“ トル ”としたところ，該文字をトルという文字列に差し替えた，という笑えない実例がある。そこで，“ トル ”という文字列に替える場合には，その前後の文字も同時に指定する，などの作業上のノウハウも必要になってくる。

これらの校正作業の重要な点は，作業そのものが履歴を残す仕組みになっていることである。たとえば“ イキ ”という校正記号は，“ さっき訂正指示をしたけれども，その指示を止めて，もとの状態を生かす ”という意味で用いる。この“ イキ ”を消して，その上に別の指示をすれば，その指示が最終指示になる。このように，各指示の履歴を確実に残すのがルールであって，決して前回の訂正個所を修正液で消して，訂正のし直しをすることはない。

この校正作業が終了し，“ 初校 ”が組版にまわって組版訂正作業が行われる。“ 初校責了 ”でなければ“ 再校 ”が戻る。これを繰返して“ 校了 ”又は“ 責了 ”となるまで，校正が続けられる。校正のテクニックとしては，再校は“ 初校の赤字ゲラ ”と見比べる。したがって，これを“ 赤字引合わせ ”とも呼ぶ。

4.2.3 校正作業の分担及び協業　校正作業は，その作業の進め方によって“ 単独校正 ”と“ 読合わせ校正 ”とに分類できる。前者を単校，後者を対校などとも称する。単独校正では，その名のとおり一人で校正する。読合わせ校正では，二人がペアを組んで，その一人が原稿（または初校ゲラ）を音読し，他の人がそれを聞きながら校正ゲラ（又は原稿）を黙読して確認していく。

校正対象文書量が少ない場合，校正作業はこのいずれの方法をとっても，一人又は二人で行うことができるが，文書量が多くなると，文書を分割して，複数の担当者が分担する場合もある。校正の基準が明確になっている限り，複数の担当者が分担作業しても特に問題になることはないが，管理者を決めて制御するのが普通である。新聞社の場合は，校閲と整理とに役割分担される。実質的な権限は，整理部長がもつが，最終的な権限は編集局長にある。

4.2.4 校正作業の実際

(1) 原稿整理　この段階で，仮名遣い・送り仮名・漢字の字体などの指示，原稿の誤字の修正，組版指示などがなされる。

(2) ゲラ刷り　原稿が組版作業担当（通常は印刷会社）に渡され，組上がりを刷って出版社に送って校正作業が行われるが，組版を行ったところでは，出版社に送る前に一度，目を通すのが普通である。これを“内校”と呼ぶ。内校では，ざっと目を通す程度である。

(3) 初校　内校後，問題があれば修正の後に出校される。これが初校である。
出版社では，この初校ゲラを受け取って校正作業を行う。その要点は，次のとおりである。

・ 原稿との厳密な照合
・ 組版指定通りに組まれているかどうかのチェック
・ 引用文などの特殊な組み方の範囲の統一性
・ ノンブル，柱などの位置及び内容
・ 図版の位置

(4) 著者校正　出版社の校正が済むと，組版作業担当（以下，印刷会社とする）に戻されるが，その前に編著者に回すのが通例である。

この著者校正で，著者が，表現の変更，内容の追加・入れ替えなどを，ある程度行うことが多い。つまり，著者校正は，出版社の校正の延長ではなく，全く別物と考えられる。

著者校正が終わった段階で，赤字の量を確認し，普通は“要再校”として印刷会社に戻される。

(5) 再校　印刷会社に戻されて初校の修正がなされると，再校として出版社に戻る。この場合，初校の赤ゲラとともに出校される。

(6) 赤字引合わせ　校正者は初校の赤ゲラと再校とを照合する。とくに，この時点で重要なことは“直すべき個所”以外に手が入っていないかどうかのチェックである。

(7) 素読み校正　赤字引き合わせが終了したものは，素読みを行うとよい。この時点では参照対象としての原稿（初校）との対比，という見方から離れ，客観的に対することができ，新たな問題点の発見が期待できる。

もし再校の赤が多ければ，要三校として印刷会社に戻し，三校が帰る。こうして四校，五校‥‥と続くわけが，かなり複雑なものでも四校までで，五校まで必要とすることはきわめて少ない。

(8) 責了　赤が少なくなった時点で，その赤字を印刷会社が責任をもって正すことを前提に校了とすることがよく行われる。これを責任校了又は責了と呼ぶ。

(9) 校了　責了を含め，校正作業の終了を意味する。校了の一歩手前で，特定の個所

だけ赤字が多い場合，この部分（ページ単位）だけをだして校正することもある。これを“念校”と呼ぶ。

4.2.5 電子文書の校正作業　これまでの手順は，従来の活版時代から踏襲された校正作業であり，次の特徴をもつ。

・ 参照対象も被校正物も“紙”である。
・ 校正作業（指示）は，“紙”の上に手書きすることによってなされる。

電子文書の校正作業については，ハードコピーをとって従来と同様の校正を行うことは可能であるが，“電子文書処理”の利点が活かされない。利点を活かすためには，画面上での操作が必要になる。通常の操作は，次のとおりである。

・ 被校正物は“画面”に見えているだけである。
・ 校正は，訂正指示を入力するのではなく，キーボードから“訂正作業”の形で実際にデータを修正する。

この場合には，履歴が残らない（Undoが効くことと履歴が残るということは，同義ではない)。annotation能力のあるアプリケーションであれば，従来型の校正も可能である。

これに対して，校正タグ付けを行う方法がある。この方法の特徴は，次のとおりである。

(1) 利点
・ 校正指示と校正作業とが完全に分離できる(もっとも，訂正作業はほとんどコンピュータによって実行される)。
・ 校正の分散作業が可能になる。

(2) 欠点
・ 校正時に訂正後の文書が可視化されない。

この校正作業をグループで行う場合，照合というレベルであれば，ほとんど問題はない。しかし，文脈を含めた手の入れ方を認めた途端に，訂正権限の問題がでてくる。

手を入れては困る部分と入れても良い部分を，著者が区分することも容易ではない。どちらかを指定して，それ以外はその逆としても問題は残る。著者にとって，あまりに当たり前のことに関しては，いずれの場合も指定から漏れやすい。ダヴィッド社の“校正ハンドブック”には，“主人公が一人称からいつのまにか三人称になっていたり，一つの文の中で，時制が150年も前に溯ったり，という‘誤植’があったが，実は小説の中の話し手の意識の混乱を暗示するための表現上のテクニックであった”という事例が紹介されている。著者は，このような基本的な部分に校正者が手を入れようとするとは考えない。手を入れても良い部分を指定することを考える場合には，固有名詞，数字などが，これらの対象外になる。しかし，本当はこのような個所の間違いもかなり多い。結局，一々著者に確認せざるをえないことになりかねない。

新聞などでは，“原文のまま”という但し書き付きで署名入りの文章が載る。それを読むと，“何々え”“私わ”などという表記にぶつかる。著者が“ぜったいに手を入れるな”という以上，最終的には編集者と言えども従わざるをえない。そこで構成者にも，原文訂正を禁ずる指示が出される。しかし明らかに日本語表記上の誤りがあり，この責任の所在を明確にする（はっきり言えば，編集・構成者側に非があるのではないことを公言する）ために，“原文のまま”という1行を入れるのである。

文書処理に限らず，電子化処理に移行すると“人の温もり”が希薄になることが多い。校正作業は，著作者と読者とを結ぶ橋渡し役であり（表向きは，著者と組版担当者との間の橋渡し役にみえる），著者・組版担当者を含めたチームプレーのプレーヤでもある。その意味での“校正コミュニケーション”とでも言うべきものが，電子校正においてどう変貌を遂げるのか。もう少し積極的に言うならば“どう変化させるべきなのか”を，今後検討する必要がある。

５．懸案事項

**** TBD ****

６．その他の解説事項

6.1 関連技術　テキスト編集操作の記述には，型付きλ式のコンセプトを用いて定式化している。λ計算，形式文法，モデル理論，LISPなどついては，それぞれ適当な文献を参照されたい。

6.2 参考文献

[1] 小林一博，本づくり必携，にっかん書房，1883-05.
[2] 藤森善貢，出版技術入門，日本印刷新聞社，1965.

７．原案作成委員会　この規格の原案を審議し作成したSGML文書校正マーク付け言語の提案型国際規格作成調査研究委員会の構成員を次に示す。

SGML文書校正マーク付け言語の提案型国際規格作成調査研究委員会　構成表

	氏名	所属
(委員長)	小町　祐史	松下電送株式会社
(幹事)	前沢　克俊	大日本印刷株式会社
	永松　荘一	通商産業省機械情報産業局電子機器課
	兼谷　明男	通商産業省工業技術院標準部
	佐々木　好洋	社団法人日本事務機械工業会

	小笠原　康直	財団法人日本規格協会
	檜山　正幸	檜山オフィス
	沢田　要	川崎製鉄株式会社
	長村　玄	ダイナラブ・ジャパン株式会社
	高柳　由美子	凸版印刷株式会社
	今郷　詔	株式会社リコー
	古瀬　幸広	国際大学グローバルコミュニケーションセンター
(事務局)	森光　裕行	国際大学グローバルコミュニケーションセンター