共有エディタの編集制御系設計†

池井 寧†† 都築功兒†† 大川善邦††

複数の人間による共同作業を計算機で支援するシステムにおいては，作業者同士が実時間で対話的に情報交換を行える機能が重要である．作業者が情報交換を行える場として設計された従来の共有の作業空間，あるいは共有ウインドウでは，作業の実時間性を損なう排他制御が実際には行われており，提示されたテキストを厳密に同時に編集することはできない．本論文では，複数の作業者が全く同時に編集操作を行うことが可能な共有エディタの制御系の基本的な設計を示す．まず最初に，2サイトで同時に編集を行った場合について，テキストの文字列と作業者のキャレットが妥当な配列となる形式を整理する．これにより，編集前の文字間に入力された文字列がサイトごとに隣接して置かれる配列と，2通りのキャレット配置法が選択される．この結果に基づき，共有テキストの構造として，編集前の各文字間にサイト別に入力位置を有する2点入力位置モデルが構成される．このモデルは，まず単純な入力形式の場合，すなわち当該サイトと共有者サイトの入力が交互せずにまとまって到着する場合について，その入力位置の定義が与えられる．さらに，より一般的な入力形式，つまり当該サイトと共有者サイトの入力が入り交じって到着する場合に拡張された後，3サイト以上が共有する場合についても一般化される．

1. はじめに

ワークステーションの普及と通信技術の著しい進展に伴い，複数の人間の共同作業を計算機で支援するシステムの研究が積極的に進められつつある．

現在，計算機による支援が考えられている共同作業としては，会議や打ち合せ，あるいは共同執筆，ソフトウェアの共同開発などがその対象に挙げられている．これらの共同作業においては，量的質的な差はあるにせよ，複数の作業者が相互に情報交換を行うことが最低限の必要条件である．したがって，これらの共同作業を支援するシステムの設計においては，複数の作業者がリアルタイムで対話的に利用できる「共有の作業空間」をどのように計算機で提示するかが中心的な問題の1つとなる．

共有の作業空間の設計においてまず重要なことは，実現された作業空間が，対象とする作業の内容とその効率的な実行に適した構造を持つことであり，このためには厳密には個々の作業の詳細な分析に基づいた個別的な設計が必要となるであろう．しかしながら，それらに共通する部分が全くないわけではない．それは，恐らくすべての共同作業が，少なくともある局面においては共有の作業空間を介しての直接的な対話を作業者に要求するということである．そして，この直接的な対話を支援するためには，作業者の実時間的な使用要求に対して，十分な余裕を持って応えられる制御構造の設計が必要となる．例えば，会議形式の共同作業では，そのほとんどすべての局面において，作業者同士が会話的な情報交換を実時間で，かつ円滑に行えることが何よりも本質的であり，これは映像や音声だけの問題でなく，共有空間内の作業についても同様である．一方，共同執筆やソフトウェアの共同開発などでは，より広義の共同作業環境，特にその基礎となる作業のモデル化が重要であり，それに基づいて作業空間を適切に設計し，局所的な情報を共有化したり，作業対象の内部的な参照拘束関係を各作業者に提示したりすることが要求される．しかしながらそれと同時に，お互いが関連する部分についての実時間的な調整作業も少なからず必要であって，これを詳細にかつ効率的に行うためには，上述の対話的な共有の作業空間も不可欠となる．

前者の会議形式の作業を対象とするシステムとしては，会議室に大型の計算機スクリーン（電子的黒板）と複数の個人用ワークステーションを配置して，これらによって共有の作業空間および個人的な作業空間を提示し，対話的な議論の場を設定する実験的環境が試作されている[1),10)]．また，これらのワークステーションは，音声や画像の通信回線を併用すれば遠隔地に分散されていても良いわけで，広域のネットワークを利用するシステムも開発されつつある[2)-5)]．一方，後者のシステムとしては，ハイパーテキストの考え方を応用して作業対象を構造化する例が多く見られる．そこでは，共同開発する文書への注釈付けの管理や共著者への連絡メッセージの送信管理を高度化したり，ソフ

† The Design of Editing Control System for a Shared Editor by YASUSHI IKEI, KOJI TSUZUKI and YOSHIKUNI OKAWA (Department of Mechanical Engineering for Computer-Controlled Machinery, Faculty of Engineering, Osaka University).

†† 大阪大学工学部電子制御機械工学科

トウェアの開発に係わる討論の進行を状態モデルで構造化する試みなどが行われている[6)-9)].

作業者が実時間で対話的に編集することを中心的には考えていない従来の後者のシステムでは，文書の管理単位となっている要素を全く同時に編集することは一般にはできない．つまり，文章を入力，添削するための各作業者のエディタは，同時に同一の編集対象要素を持つことはできない．しかも現在は，その要素が文書の中でも比較的小さな領域，すなわち段落や文のような単位とされているのでもない．しかしながら，1つの文書を共同で開発する以上，同一部分を見ながら作業者間で協議する局面は必ず存在し，もちろんそれに伴って文書の変更が行われるはずである．協議や変更を最も効率的に行うためには，対面して直接原稿に書き込むのと同様に文書ファイルを変更できれば良い．そうでなければ，協議の結果を文書に反映させるための手続きが別に必要になる．

協議と並行して文書を添削，改訂する作業では，会話しながらポインタで文書の細部をお互いに指示するだけでなく，そこに実際に文字や記号などの具体的な表示を与えることが必要である．しかも，それは議論の焦点となっている部分において，対話者間で全く独自に任意のタイミングになされること，つまり，決して1本の鉛筆を取り合うような制限を受けることなく行えることが理想であろう．これを実現するためには，編集作業を管理するエディタが，文書に変更を加える編集操作の最小の1単位から並行動作を許容するように共有化されていなければならない．ここでの並行動作とは具体的には，各作業者がそれぞれ文書中に注目点（フォーカス）を持ち，これに対して任意の編集操作を並行して行えるということである．この共有形のエディタは，もちろん，会議形のシステムにおける共有空間の編集管理にも重要な機能である．つまり，参加者が共有のスクリーンあるいは個々のワークステーションの共有ウインドウに表示された議論の対象を前にして，銘々が並行して変更や注釈付けの操作を行えるために必要である．討論それ自身が中心の会議では，各自のワークステーション相互間で共有空間の内容に少々一貫性が欠如したとしても問題とならない場合もあるが，議論の結果を特に正確な記録として残さねばならないときには，一貫性を厳密に保証するエディタが不可欠となる．

以上の議論から，複数の作業者がエディタを共有する時に要求される条件が幾つか導かれるであろうが，その中でも基本的なものとして，各作業者にとっての編集位置の任意性と十分な応答性とを考えることができる．この2つの条件を同時に満たすことは必ずしも容易ではない．というのは，即応性，特に広域のネットワーク環境においても作業者各自の編集操作に対する応答性を損ねないことを考えると，文書のコピーをそれぞれのワークステーションに分散することが必要となるが，そうすると同時に同じ位置を複数の作業者が編集した場合の結果の同一性が問題となる．つまり，もし文書を編集しているエディタが中央に1つ用意され，これが作業者からの編集入力を逐次的に受けて編集結果を全員に送り返すことにすれば，エディタの設計は極めて容易であるが，各作業者に対する応答性を保証するためには非常に高速のネットワークとワークステーションを用いなければならず，この条件の充足は作業者間の距離や作業者数が増加すると次第に困難となる．したがって，エディタのプロセスと編集する文書のコピーを各ワークステーションに分散して個々の作業者への応答性を確保しながら，しかも並行に編集される文書の同一性を保つことが必要となる．

ここで考えている仮定とそこから導かれる問題を言い替えれば，即応性を保証するため各作業者が入力した編集操作は直ちにそのワークステーションのエディタプロセスで実行されるが，他のプロセスへはネットワークの通信時間の遅れを伴って到着するということであり，ゆえに各操作者が入力した編集入力を伝達するメッセージ（自分自身のエディタプロセスへもメッセージによって伝達されると考える）の到着順が，それぞれのエディタプロセスによって異なる可能性があるということである．編集操作を指示するメッセージの到着順が異なれば，編集結果を同一にすることは一般にはできない．これと類似する問題としては分散データベースの書き換えの問題がある[12)]．この問題の解決方法としては，まずすべてのワークステーションにおいて受信する伝達メッセージの到着順が等しくなるようにメッセージを順序付ける方法がある[11)]が，先に述べた仮定では応答性を優先するためにメッセージの到着順が異なることを積極的に認めているのでこの方法を用いることはできない．順序づけをするためには順序の比較に相当する何らかの方法[11)]が必要であり，ネットワーク上では相当のオーバヘッドが生じることを避けられないからである．さらに，通常用いられるロック制御では，対象となる共有空間の領域をあらかじめ分割し，それぞれの領域に対するロックを得

た作業者だけがその領域を変更することが許される．しかしロックを得るためにはロックの授受が必要であり，これも編集入力の処理に際して，前方法と同様のオーバヘッドを生じることになる．複数の作業者に編集位置の任意性を認めるとすれば，文書の最小単位，例えば文字単位にロックを掛けることになり，ロックの授受は極めて頻繁に起こるので，オーバヘッドは極端に増加するはずである．

従来の会議形システム，例えば Xerox の Colab[1] では，このオーバヘッドを避けるために，編集メッセージを単純にブロードキャストして着信側でビジーの表示をさせるにとどめ，同一の部分への競合的な入力が発生することを許容している（ただし，入力の衝突は検出される）．また，Capture Lab[10] では共有スクリーンへの編集操作は，一人の作業者のみに許可されているだけである．いずれも，作業者への制限がより少なく，かつ同一性を保証できる制御法の必要性について言及している．

著者らは，文字だけからなる1つの文書を複数の作業者が全く同時に編集できる共有エディタの制御系を設計した．本エディタは，作業者が使用するすべてのワークステーションに分散されたエディタプロセスから構成され，入力された編集操作をメッセージ通信で相互に伝達する形態を有する．通常のエディタと同様の応答性を損なうことなく任意の時点で編集可能で，かつ編集可能領域も制限されない操作性を得るために，本エディタでは編集する文書テキストの構造化を行った．また，構造化に先立ち，共有テキストを編集する際のエディタの動作の基本的な性質を調べている．以下では，まずこの性質について述べ，さらにそこから導かれる共有エディタ制御系の基本的な設計を示す．

2. 共有エディタの構成要件と設計条件

複数の作業者が同時に編集操作を行うことができる共有エディタの基本的な構成要件として，ここでは次の各点を考える．

①各サイトの編集操作は，通常の（非共有）エディタと同様に時間遅れなく処理されること．このためには，エディタがサイトごとに処理系を持ち，入力された編集要求を各サイトで直ちに処理しなければならない．

②その結果，複数サイトでほぼ同時に入力が行われた場合，各サイトのテキストは一時的に相違するが，入力終了後一定時間以内にこれらは再び一致すること．

③エディタを共有する各作業者は，各々編集位置指示用のキャレットを有し，それらはすべて共有エディタの画面内に表示され得ること．これはキャレットが作業者間の意志疎通の手段となると同時に，他の共有者の入力位置を明示することによって，テキストが予期せぬ位置で変化することをなくすために必要である．

本論文ではこれらの要件を満たすエディタの設計を与えるが，問題を簡単化するために次の仮定を置く．

①編集対象のテキストは，1次元の有限長文字列とする．

②編集操作は，1文字単位の挿入と削除，および編集位置を示すキャレットの前後1文字分移動の4種類だけとする．

③編集対象の各文字は一意に識別できる．

2次元的なブロックを単位とした編集も，基本的には1文字単位の上記操作に分解できるので，以下で述べる設計を適用可能である．ここでは，理解を容易にするために①，②の前提の下で考える．さらに，③はテキストバッファの1文字当たりの容量を増加することで容易に実現できる．

3. 共有テキスト編集の基本的性質

3.1 使用語彙と記号の定義

以下に本論文に用いる語彙および記号と，エディタの動作環境に関する仮定を述べる．

[定義]

サイト S_s…並行に編集を行う操作者と計算機が置かれた1つのサイトであって，そのアドレス（識別子）が s ($s \geq 1$) であるもの．

表示文字…サイトの計算機画面に表示可能な英数字，漢字，記号などの通常の文字．

制御文字…一文字削除，およびキャレットの高位・低位方向各1文字の移動，の合計3種類の編集制御要求に相当する特殊文字．（テキストには含まれない）

テキスト…一意に識別可能な表示文字からなる1次元の文字列．

文字の順位（数）…テキストの先頭を1として，1文字単位に末尾に向かって付けた番号で，先頭方向を低位の方向，末尾方向を高位の方向と呼ぶとする．

キャレット…テキストを画面に表示したとき，現在注目している位置を示すもので，表示文字間に置いた

特別な記号（後述）である*．テキストには含まれない．あるサイトについて見たとき，そのサイトのキャレットを**自己キャレット**，また同時に編集中の他のサイトのそれを**共有者キャレット**と呼ぶ．

入力文字 c_j^i…サイト S_i においてキーボードから入力された文字の中で j 番目（$j \geqq 1$）の文字であり，表示文字あるいは制御文字のいずれかの文字．

挿入文字 Ins(c_j)…ここで仮定するエディタは入力モードを持たず，入力文字 c_j^i が表示文字の時はテキストへ挿入される文字と見なされる．この文字を挿入文字と呼び，Ins(c_j^i) と表示する．挿入文字は，キャレットが示す文字の間の位置に挿入される．

削除制御文字（BS 文字）…キャレットの低位側の1文字を削除することを指示する制御文字で，BS 文字と略記する．

低位方向移動文字（←文字）…キャレットを低位方向へ1文字分移動させる左向きカーソルキーに対応する制御文字で，←文字と略記する．

高位方向移動文字（→文字）…キャレットを高位方向へ1文字分移動する右向きカーソルキーに対応する制御文字で，→文字と略記する．

基準文字 D_i…編集開始前に全サイトにおいて一致しているテキストの文字を基準文字と呼び，その中で順位が i ($i \geqq 1$) である文字を D_i と記す．編集前のテキストの文字数が全部で t 文字である時，テキストは "$D_1D_2 \cdots D_iD_{i+1} \cdots D_t$" で表される．

バッファ文字 B_i…編集開始の前後に関わらずそのサイトのテキストとして表示されている表示文字をバッファ文字と呼び，その中で順位が i ($i \geqq 1$) である文字を B_i と記す．編集開始前ならば．$B_i = D_i$ である．特に，B_i の添え字 i は任意の時点におけるテキストの先頭からの順位であることに注意する．

フォーカス（B_iB_{i+1}）…キャレットによって示される位置，すなわちテキストの現在の編集対象位置をフォーカスと呼び，表示されているテキスト中の隣接する2文字，例えば B_i，B_{i+1} の組 (B_iB_{i+1}) でこれを表す．この B_i，B_{i+1} を各々低位フォーカス，高位フォーカスと呼ぶ．

自己フォーカス…当該サイト（自己サイト）のフォーカスである．

共有者フォーカス…当該サイトに表示されている共有者キャレットが表している（名目上の）共有者サイトのフォーカスである．

エディタプロセス…同一のテキストを同時に編集するために各サイトの計算機において並行に実行される編集処理プロセス．特定のサイトの計算機にとっては，このプロセスは単一のプログラムであって，テキストに対する編集命令は逐次的に実行される．またテキストは，編集開始前に同一のコピーが全サイトのエディタプロセスに与えられているとする．各サイトのキーボードからの入力文字は，そのサイトのエディタプロセスにまず読み込まれる．エディタプロセスでは，その第1段階として，文字が入力された時点の編集対象位置であるフォーカスを得る．（フォーカスの生成）フォーカスは，現在表示されている文字，すなわちバッファ文字の中から得られる．次に入力された文字とこのフォーカスとを併せて解釈することで，テキストに対して加えるべき編集操作が決定され編集命令が作られる．次にエディタプロセスは，第2段階として，その編集命令を実行しテキストを変更するとともに，編集命令伝達メッセージによってその編集命令を他のサイトへ送信する．エディタプロセスの第1段階および第2段階は，文字の入力後，直ちにかつ連続して行われる．また，エディタプロセスにとっての送信作業は，そのサイトで並行に動作している通信管理プロセスに対して編集命令伝達メッセージを渡すだけであり，通信のための同期待ちなどはない．

編集命令…エディタプロセスの第1段階で決定された編集操作を記述した指示．以下に定義される4種類の命令がある．

挿入編集命令…挿入文字に対応する編集命令であり，対象位置を示すフォーカス，例えば (B_iB_{i+1}) と，挿入文字，例えば Ins (c_j^i) とからなる．

削除編集命令…BS 文字に対する編集命令であり，削除対象文字が B_i の時，Del (B_i) と表す．フォーカスの更新のために，削除後のフォーカス ($B_{i-1}B_{i+1}$)** も付加しておく．

移動編集命令…←文字，→文字に対応する編集命令であり，キャレットの移動先のフォーカスが (B_iB_{i+1}) の時，それぞれ←(B_iB_{i+1})，→(B_iB_{i+1}) と表す．

編集命令伝達メッセージ…各サイトでの入力文字から導かれた編集命令を，共有者のエディタプロセスに伝達するためのメッセージ．内容は，挿入編集命令・削除編集命令・移動編集命令のいずれかである．編集

* ここでの使用目的は一般的な "カーソル" と等しいが，表示位置が文字間であることが異なる．

** ここでの表記に限り，B_{i-1} および B_{i+1} は削除実行前の順位 $i-1$ および $i+1$ の文字．

命令伝達メッセージは，計算機ネットワークですべての共有者サイトに伝達されるが，この際の仮定として，あるサイトから送信されたメッセージは，そのサイトのメッセージについては送信された順序で共有者サイトに到着するということだけを与える☆.

一括入力…全サイトが同一のテキストを有する初期状態において当該サイトが編集入力を開始し，その入力列とそれらに対する同サイトのエディタプロセスでの編集の実行がすべて終了した後に，共有者サイトからの編集命令が到着するという単純化された場合の入力形式である．(次節で詳しく述べるように，一括入力は交代入力を考える際の入力群の単位となる.)

交代入力…当該サイトの編集命令と共有者サイトの編集命令が交互に入り交じってエディタプロセスに到着する，より一般的な場合の入力形式である． ■

3.2 2サイト一括入力時のテキストとフォーカスの更新

まず，最も単純な2サイトの一括入力において，テキストとフォーカスを更新する場合について考える．問題となる点は，最初に述べたように，両サイトの編集命令が各々のエディタプロセス（の第2段階）に到着する順序が逆転することである．しかし，その場合でも，もし両サイトの編集位置（フォーカス）が十分に離れているならば，たとえ両者の編集命令の到着順が逆転しても，フォーカスで指定された位置をそれぞれに編集するかぎり同一の結果が得られ問題はない．また，両者が移動命令だけを行うならば結果は自明である．したがって問題となるのは，自己フォーカスが共有者フォーカスに接近していて，当該サイトの編集命令が共有者フォーカス，および将来の共有者フォーカスを変更したり削除したりして，両者の編集命令がお互いに干渉する場合である．

以後，当該サイトでn文字分の入力を編集した後に共有者サイトのm文字分の編集命令が到着する場合をn/m 次入力と呼ぶ．2サイト一括 n/m 次入力において，編集命令が両サイトのエディタプロセス（の第2段階）に到着するタイミングを**図1**に示す．図では縦の下向き方向が時間の経過であり，太線は各エディタプロセスに対してn文字，またはm文字相当の編集命令が継続的に到着することを表している．(エディタプロセスの第1段階は，編集命令到着に先立って文字が入力されたサイトで行われているとする.) また，

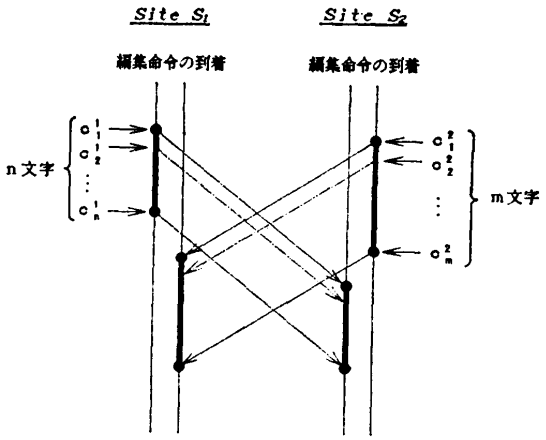


図1 2サイト一括 n/m 次入力時の編集命令到着のタイミング
Fig. 1 Edit command arrival timing of two site n/m order packaged input.

交差する矢印は編集命令伝達メッセージが通信時間の遅れを伴って共有者サイトに到着することを表している．

さらに以下では，エディタ画面中で自己フォーカスを表すキャレットを|，共有者フォーカスを表すキャレットを⋮で表示する．

(a) 1/0 次入力時の更新 当該サイトで1文字だけを入力し，共有者サイトは何もしない場合であるが，結果は必ずしも自明ではない．干渉が起こる場合，つまり当該サイトの1次入力が現在の共有者フォーカス (D_iD_{i+1}) をも変更するのは，同じフォーカス (D_iD_{i+1}) へ挿入する時，および D_i または D_{i+1} を削除する時である．

まず，フォーカス文字が削除される場合では，被削除文字の両隣文字が新しい共有者フォーカスとなり，これは一意に決まる．なお，削除編集命令の実行方法について補足しておくと，仮定にあるように各文字は一意に識別可能としているので，削除編集命令の実行は指定された文字をテキストの中から検索してこれを削除することによる．3.2(b)節以降で複数の入力がある場合に両サイトで同一文字を削除することになった場合は，テキスト中にそれがなければ削除は既に実行されたとして完了すれば良い．

次に，当該サイト1が文字 c_1^i を挿入した場合では，**表1**のように新しい共有者フォーカスを2通り考えることができる．つまり，共有者フォーカスが ($D_ic_1^i$) となる場合①と，($c_1^iD_{i+1}$) となる場合②である．表では両者のフォーカスが一致している初期状態と2通りの挿入後状態について，挿入を行うサイト1から見た

☆ 実現上の方法，例えばサイトの受信バッファで整列してからエディタプロセスに渡すなど，については任意である．

表 1 共有者フォーカスの2つの更新可能位置
Table 1 Two possible renewal positions of co-owner's focus.

	挿入サイトでの表現	共有者サイトでの表現
初期状態	$D_i \| D_{i+1}$	$D_i \| D_{i+1}$
挿入後①	$D_i \| c_i^1 \| D_{i+1}$	$D_i \| c_i^1 \| D_{i+1}$
挿入後②	$D_i c_i^1 \| D_{i+1}$	$D_i c_i^1 \| D_{i+1}$

表 2 1/1 次挿入文字入力時の2つの可能な結果
Table 2 Two possible results of 1/1 order insert character input.

	低位サイトでの表現	高位サイトでの表現
初期状態	$D_i \| D_{i+1}$	$D_i \| D_{i+1}$
挿入後ⓐ	$D_i c_i^1 \| c_i^2 \| D_{i+1}$	$D_i c_i^1 \| c_i^2 \| D_{i+1}$
挿入後ⓑ	$D_i c_i^1 c_i^2 \| D_{i+1}$	$D_i c_i^1 c_i^2 \| D_{i+1}$

表現と共有者サイトから見た表現の両方を記した．両者の相違は，フォーカス位置を示すキャレットの表現が入れ替わっている点である*．なお，挿入後の自己フォーカスについては何れの場合も必ず $(c_i^1 D_{i+1})$ でなければならない．すなわち，挿入を行った後のキャレットは挿入文字より必ず高位側に置かなければならない．これは，その後引き続いて連続的に文字を入力する時，文字列が高位方向に向かって整列されるために必要な条件である（昇順入力条件）．

（b） 1/1 次入力時の更新 1/1 次入力では当該サイトの入力文字に続いて共有者サイトの編集命令がエディタプロセスに到着する．まず，両サイトの入力文字が共に挿入文字の場合について調べる．

干渉が起こるのは，共有者サイトの挿入編集命令が到着した時，挿入位置を指示するフォーカスが現テキストにおいては既に隣接文字の組ではなくなっている場合であり，これは，両者の初期フォーカスが一致していた時に起こる．この場合の結果を考える上で仮定する基本的な方針は，共通のテキストに対して両サイトで同時**に入力された文字は，それが対象位置に挿入されることにおいて同等の権利を持つとすることである．つまり，両サイトから同じフォーカス $(D_i D_{i+1})$ に対して各々 c_i^1, c_i^2 を同時に挿入したならば，それらの文字は，少なくとも指定された基準文字 D_i と D_{i+1} に挟まれた領域の中に置かれるとすることである．ただし，挿入された結果を両サイトで同一とするためには，c_i^1 と c_i^2 の順位は決定されなければならない．この決定方法として，一方のサイトの文字を常に低位側，他方を高位側とするサイト固定順位形と，挿入された文字 c_i^1, c_i^2 の種類に依存して順位を決定する文字依存順位形の2つの方法が考えられる．しかし後者は，文字比較の作業が必要になるだけでなく，後に述べる多文字数入力の場合に両サイトの挿入文字列を不規則に混合する結果となり利点が少ない．したがって，ここでは前者だけについて考える．

サイト固定順位形で挿入文字が低位側となるサイトを**低位サイト**，高位側となるサイトを**高位サイト**と呼び，低位サイトのアドレスを1，高位サイトのアドレスを2とすると，両サイトの入力文字は c_1^1, c_1^2 で表される．この時，昇順入力条件を考慮すると可能な結果は**表2**に示す2通り（ⓐ, ⓑ）となる．つまり，挿入後の低位サイトのフォーカスの取り方に2通りの選択があり，高位サイトの入力文字 c_1^2 を高位フォーカスとする場合ⓐと，低位フォーカスとする場合ⓑとが可能である．高位サイトのフォーカスについては，自身の入力文字 c_1^2 を低位フォーカスとしなければならないので，他の選択肢はない．

この1/1次挿入入力の結果は，自己の挿入に続いて共有者サイトの編集命令が到着して得られるから，自己の挿入だけを考えた1/0次入力の結果が，その中間段階となっていなければならない．両フォーカスが一致した初期状態から1/1次の結果への変化の系列を，キャレットが不自然な移動をしないように組み合わせると，**図2**となる．この4通り* 以外は，共有者キャレットの無意味な移動を伴うことになり，エディタの制御法として好ましくない．エディタの制御において

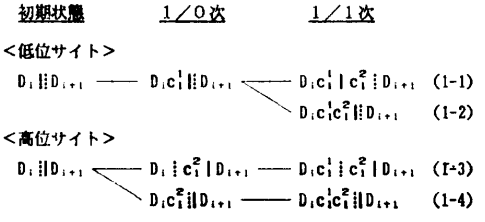


図 2 1/1 次挿入入力の文字列構成系列
Fig. 2 String forming series of 1/1 order insert input.

* 同一文字間の2つのキャレットの順序を区別する必要は実質的にはないが，ここでは分かりやすいように順序があるとして記述する．

** 同時とは，ここでは共有者サイトで入力された文字が到着するより以前に，自己サイトの文字が入力されたことを意味する．同時の関係は，4章で交代入力をより詳細に定義する中で正確に理解される．

* ただし，(1-4)の場合については，低位サイトの挿入文字 c_1^1 がそのキャレット ‖ に隣接しない位置に挿入されており，やや不自然であるが，直後に述べられている保存形でこれに対応する(1-2)では問題はない．ここでは，保存形の場合も残して議論を進める．

は，速い応答速度と共に，操作者にとって不自然でない一貫性のある動作が必要である．ここでもし，低位サイトにおいて，1/0 次の状態で共有者フォーカスを $(D_i c_i^1)$ としたならば，$D_i \vdots c_1^1 | D_{i+1}$ となるが，その後の 1/1 次の(1-1)，(1-2)に至る過程では，共有者の挿入文字 c_1^2 がその時点の見かけのフォーカス $(D_i c_i^1)$ に挿入されないばかりでなく，c_1^1 の高位隣側に挿入された c_1^2 にくっついて共有者キャレット⋮が飛ぶことになり，動作が明らかに不自然となる．

これら4つの中で(1-1)と(1-3)は同じ結果を導くために両サイトで対応する1つの系列であり，(1-2)と(1-4)もまた同様である．前者は，高位サイトの文字が入力された時点でフォーカスが分離する制御であり，これを以後**分離形制御**と呼ぶ．一方，後者はフォーカスが同じ位置のまま保存される制御であり，

表 3 1/1 次入力における両サイト挿入以外の編集命令の干渉パタンと結果の文字列
Table 3 Interference patterns of edit command and their result strings in 1/1 order input except for the both site insertion.

		初期状態	編集命令	1／1次分離形	1／1次保存形
1	LS	$D_{i-1}D_i \| D_{i+1}$	$Ins(c_i^1)$	$D_{i-1}c_i^1 \| D_{i+1}$	←
	HS	$D_{i-1}D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1}c_i^1 \| D_{i+1}$	←
2	LS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^1)$	$D_{i-1}c_i^1 \| D_{i+1}$	←
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1}c_i^1 \| D_{i+1}$	←
3	LS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^1)$	$D_{i-1}c_i^1 \| * \| D_i$	$D_{i-1}c_i^1 * \| D_i$
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$\leftarrow(D_{i-1}D_i)$	$D_{i-1}c_i^1 \| * \| D_i$	$D_{i-1}c_i^1 * \| D_i$
4	LS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^1)$	$D_i c_i^1 \| * \| D_{i+1}$	$D_i c_i^1 * \| D_{i+1}$
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$\rightarrow(D_iD_{i+1})$	$D_i c_i^1 \| * \| D_{i+1}$	$D_i c_i^1 * \| D_{i+1}$
5	LS	$D_{i-1}D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1} \| c_i^2 \| D_{i+1}$	$D_{i-1}c_i^2 \| D_{i+1}$
	HS	$D_{i-1}D_i \| D_{i+1}$	$Ins(c_i^2)$	$D_{i-1} \| c_i^2 \| D_{i+1}$	$D_{i-1}c_i^2 \| D_{i+1}$
6	LS	$D_{i-1} \| D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1}c_i^2 \| D_{i+1}$	←
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^2)$	$D_{i-1}c_i^2 \| D_{i+1}$	←
7	LS	$D_{i-1}D_i \| D_{i+1} \|$	$Del(D_i)$	$D_{i-1} \| * \| D_{i+1}$	$D_{i-1} * \| D_{i+1}$
	HS	$D_{i-1}D_i \| D_{i+1} \|$	$\leftarrow(D_iD_{i+1})$	$D_{i-1} \| * \| D_{i+1}$	$D_{i-1} * \| D_{i+1}$
8	LS	$D_{i-1}D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1} * \| D_{i+1}$	←
	HS	$D_{i-1}D_i \| D_{i+1}$	$\leftarrow(D_{i-1}D_i)$	$D_{i-1} * \| D_{i+1}$	←
9	LS	$D_{i-1} \| D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1} \| * \| D_{i+1}$	$D_{i-1} * \| D_{i+1}$
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$\rightarrow(D_iD_{i+1})$	$D_{i-1} \| * \| D_{i+1}$	$D_{i-1} * \| D_{i+1}$
10	LS	$D_{i-1} \| D_iD_{i+1} \|$	$Del(D_{i+1})$	$D_{i-1}D_i * \|$	←
	HS	$D_{i-1} \| D_iD_{i+1} \|$	$\rightarrow(D_iD_{i+1})$	$D_{i-1}D_i * \|$	←
11	LS	$D_{i-1} \| D_i \| D_{i+1}$	$\leftarrow(D_{i-1}D_i)$	$D_{i-1} * \| c_i^2 \| D_i$	$D_{i-1} * c_i^2 \| D_i$
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^2)$	$D_{i-1} * \| c_i^2 \| D_i$	$D_{i-1} * c_i^2 \| D_i$
12	LS	$D_{i-1}D_i \| D_{i+1} \|$	$\leftarrow(D_iD_{i+1})$	$D_{i-1} * \| D_{i+1}$	←
	HS	$D_{i-1}D_i \| D_{i+1} \|$	$Del(D_i)$	$D_{i-1} * \| D_{i+1}$	←
13	LS	$D_{i-1}D_i \| D_{i+1}$	$\leftarrow(D_{i-1}D_i)$	$D_{i-1} * \| D_{i+1}$	←
	HS	$D_{i-1}D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1} * \| D_{i+1}$	←
14	LS	$D_{i-1} \| D_i \| D_{i+1}$	$\rightarrow(D_iD_{i+1})$	$D_i * \| c_i^2 \| D_{i+1}$	$D_i * c_i^2 \| D_{i+1}$
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$Ins(c_i^2)$	$D_i * \| c_i^2 \| D_{i+1}$	$D_i * c_i^2 \| D_{i+1}$
15	LS	$D_{i-1} \| D_i \| D_{i+1}$	$\rightarrow(D_iD_{i+1})$	$D_{i-1} * \| D_{i+1}$	←
	HS	$D_{i-1} \| D_i \| D_{i+1}$	$Del(D_i)$	$D_{i-1} * \| D_{i+1}$	←
16	LS	$D_{i-1} \| D_iD_{i+1} \|$	$\rightarrow(D_iD_{i+1})$	$D_{i-1}D_i * \|$	←
	HS	$D_{i-1} \| D_iD_{i+1} \|$	$Del(D_{i+1})$	$D_{i-1}D_i * \|$	←

ただし，LS…低位サイトでの表現，HS…高位サイトでの表現，
＊…移動先フォーカスへの仮想的挿入文字

同様に**保存形制御**とする．つまり，1/1 次挿入入力に対するサイト固定順位形制御では，分離形と保存形の2種類のフォーカス制御が，キャレット移動の面から妥当であることが分かる．

両サイトの入力が共に挿入の場合以外で，かつ干渉がある時，つまり編集命令の到着時に指定フォーカスが一部を欠いたり，隣接文字でなくなる入力の組合せは全部で 16 通りある．これらを**表3**に示す．低位サイト (LS) と高位サイト (HS) の両方について，初期状態とそれぞれの編集命令，そして分離形，保存形の結果である．ここで分離形，保存形の区別が必要なのは，移動編集命令をそれが指定するフォーカスへの仮想文字＊の挿入編集と見なして結果を導いているからである．こうすることにより，移動先のフォーカスが非隣接状態であったり，削除されていた場合でも，挿入の場合と同様に妥当な結果を統一的に得ることができる．もちろんこの仮想文字は，表示あるいは保存される必要のない便宜的な存在である．なお，この表で分離形と保存形でそれぞれ導かれた結果が実質的に異なるのは，5, 11, 14 の場合だけである．

(c) 2/2 次以上の挿入文字入力時の更新　3.2 (b)節と同様に考えると，2/2 次で挿入文字だけからなる編集命令が与えられた場合の結果は，1/0 次，2/0 次，2/1 次，2/2 次の順序の挿入系列において，キャレットが不自然な移動をせずに導かれることが必要である．まず，3.2(b)節で導入した低位サイト，高位サイトの区別をせずに，つまりサイトの相違に基づいた挿入文字の順序付けをせずに，この系列を列挙すれば，**図3**となる．ただし，ここで当該サイトのアドレス s を1とする．

図3において 1/0 次は前節と同様である．2/0 次については，文字の順位とフォーカスの更新位置に関して可能な場合をすべて挙げてある．昇順入力条件を考慮すると，共有者フォーカスの位置だけが異なる3つの場合がある．この中で *1) の場合は，1/0 次の①，②のどちらの状態から至るとしても共有者フォーカスの制御に一貫性を欠いている．つまり，1/0 次の①は挿入サイトを高位サイトと見た場合 (図2の(1-3)参照) であるが，*1) で再び共有者キャレット⋮を移動させる必然性がない．c_2^1 はサイト1によって c_1^1 と D_{i+1} の間に挿入されるのであり，この時 D_i と c_1^1 の間にある共有者キャレット⋮は，本来全く関係ないと考えられるからである．また②から *1) に至る場合では，2文字目の挿入においてキャレットが分離するこ

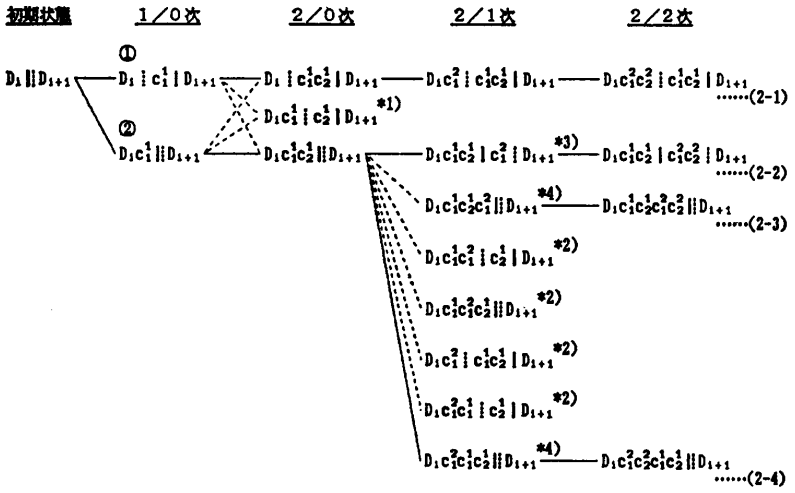


図 3 2/2 次挿入入力の文字列構成系列
Fig. 3 String forming series of 2/2 order insert input.

とになるが，これを認めれば n 文字目で分離する方式も考えなければならない．これらの方式では，分離する場合としない場合の区別が必要となり，制御が複雑化するだけである．したがって，*1) に至る系列は不適当であり，それに続く 2/1 次以降については省略した．また，破線で表された残りの組合せも，この議論と同様の理由から不適当となる．

次に 2/1 次では，昇順入力条件を考慮すると 7 通りの結果が考えられる．2/0 次の $D_i c_1^1 c_2^1 \| D_{i+1}$ に続く 2/1 次の結果にそのすべてを記載した．この中で *2) の 4 つの結果に至る場合は，挿入文字とキャレットの位置に関して上述の議論と同様，一貫性のない制御を必要とし不適当である．これに対し *3) は，3.2(b)節で検討した高位サイトの挿入時にフォーカスが分かれる分離形，*4) は保存形であると見ることができる．2/0 次のもう 1 つの有効な結果，$D_i \vdots c_1^1 c_2^1 \mid D_{i+1}$ に続く系列は，7 通りのうち 1 通りだけが適当である．

さらに，2/2 次では全部で 14 通りの結果が考えられる．表 4 にこれらを示す．この中で同様の選択をすると，2/1 次で妥当と考えられる各々に対して，丁度 1 つずつが対応する．これらの最終結果の挿入文字の順位は，サイトごとに挿入文字が隣接している点で共通している．入力サイト別に見た文字列が，高位側と低位側とで入れ替わった 2 種類があるのは，3.2(b)節の低位サイト，高位サイトの区別がこの場合も可能であることを意味している．つまり，(2-2)はサイト 1 が低位サイトであって，これが $c_1^1 c_2^1$ を挿入し，サイト 2 が高位サイトとして $c_1^2 c_2^2$ を挿入した結果であると考えられる．これは 3.2(b)節と同じ割り当てをした場合に相当する．これに対して(2-1)は逆の割り当て，すなわちサイト 2 が低位サイト，サイト 1 が高位サイトに割り当てられた場合の結果である．さらにこれら 2 つの系列では，高位サイトの挿入でフォーカスが分かれる分離形のフォーカス制御が行われたと見ることができる．残り 2 つの系列(2-3)，(2-4)は，上記 2 種類のサイトの割り当てそれぞれに対して，保存形のフォーカス制御を行った結果であると見ることができる．

以上より，2/2 次より大きい次数の挿入文字入力に対しても同一サイトの挿入文字を隣接させた配列が妥当な結果であり，それは 1/1 次のサイト固定順位形からの拡張であること，さらにフォーカスの制御では分

表 4 2/2 次挿入文字入力の昇順入力条件を満たす文字列
Table 4 Result strings of 2/2 order insert character input which fulfill the ascending order input condition.

	挿入文字列	キャレットも含む可能な結果
1	$c_1^1 c_2^1 c_1^2 c_2^2$	$c_1^1 c_2^1 \mid c_1^2 c_2^2 \mid$
2		$c_2^1 c_2^2 \mid c_1^2 \mid$
3		$c_1^1 c_2^1 c_1^2 c_2^2 \|\|$
4	$c_1^1 c_1^2 c_2^1 c_2^2$	$c_1^1 c_1^2 c_2^1 \mid c_2^2 \mid$
5		$c_1^1 c_1^2 c_2^1 c_2^2 \|\|$
6	$c_1^2 c_1^1 c_2^1 c_2^2$	$c_1^2 c_1^1 c_2^1 \mid c_2^2 \mid$
7		$c_1^2 c_1^1 c_2^1 c_2^2 \|\|$
8	$c_1^1 c_1^2 c_2^2 c_2^1$	$c_1^1 c_1^2 c_2^2 \mid c_2^1 \mid$
9		$c_1^1 c_1^2 c_2^2 c_2^1 \|\|$
10	$c_1^2 c_1^1 c_2^2 c_2^1$	$c_1^2 c_1^1 c_2^2 \mid c_2^1 \mid$
11		$c_1^2 c_1^1 c_2^2 c_2^1 \|\|$
12	$c_1^2 c_2^2 c_1^1 c_2^1$	$c_1^2 c_2^2 \mid c_1^1 c_2^1 \mid$
13		$c_1^2 c_2^2 c_1^1 \mid c_2^1 \mid$
14		$c_1^2 c_2^2 c_1^1 c_2^1 \|\|$

表 5 n/m 次挿入文字入力の結果
Table 5 The results of n/m order insert character input.

	低位サイトでの表現	高位サイトでの表現
分離形	$D_i c_1^1 \cdots c_n^1 \mid c_1^2 \cdots c_m^2 \mid D_{i+1}$	$D_i c_1^1 \cdots c_n^1 \mid c_1^2 \cdots c_m^2 \mid D_{i+1}$
保存形	$D_i c_1^1 \cdots c_n^1 c_1^2 \cdots c_m^2 \,\|\|\, D_{i+1}$	$D_i c_1^1 \cdots c_n^1 c_1^2 \cdots c_m^2 \,\|\|\, D_{i+1}$

離形，保存形の2つが1/1次と同様に存在することが容易に推察できる．したがって，n/m 次サイト固定順位形挿入文字入力において，低位サイト1で c_n^1 まで，高位サイト2で c_m^2 まで挿入した場合は，**表5**に示す結果になると考えられる．

入力が2/2次以上の場合で，かつそれらが挿入文字だけではない，しかも干渉があるという場合については，そのすべての場合を挙げることはできない．次章では，これまでの結果に基づいて，編集の過程にあるテキストを適切に構造化することによって，並行に行われた編集の結果を一意に定義することを考える．

4. 共有テキストの構造化

4.1 2サイト一括入力時の共有テキスト

ここでは，挿入だけでなく削除およびフォーカス移動が含まれる編集命令が与えられた場合について考える．2/2次以上で，かつ削除，移動を含む編集後にもテキストが互いに等しくなるためには，挿入位置やフォーカスの移動位置が編集命令の到着順に依存せずに一意に決定できるようにテキストを構造化すれば良い．前章の結果を考慮すれば，編集前のテキストの各文字の間に，挿入文字の配列されるべき領域が，2つのサイトについて分離して存在する**2点入力位置モデル**によって，その構造の1つが与えられる．**図4**に2点入力位置モデルの概念図を示す．編集前のテキスト，すなわち基準文字列を $D_1 \cdots D_i D_{i+1} \cdots D_l$ とする時，任意のフォーカス $(D_i D_{i+1})$ が示す文字 D_i，D_{i+1} の間は，2つの領域に分かれていると考える．両サイトがフォーカス $(D_i D_{i+1})$ に対して挿入した場合，低位サイトの挿入文字であれば，2つの領域のうち，低位側すなわち D_i に近い側の**領域1**に置き，高位サイトの文字であれば D_{i+1} の側の**領域2**に置く．以降，低位サイトのアドレスを1，高位サイトのそれを2として，領域番号と一致させておいても一般性に差し支えない．前章の最後で考えた場合，つまり低位サイトが n 文字，高位サイトが m 文字連続に同一の初期フォーカス $(D_i D_{i+1})$ に挿入した場合の結果が図4である．ただし，これは低位サイトから見た表示であり，低位サイトのキャレットは，分離形制御なら①，保存形制御なら②の位置に置かれていると考える．高位サイトのキャレットは③の位置である．編集制御のために内部的な構造として設けられた基準文字の間のこの2つの領域を**レベル1の領域**と呼び，低位側，高位側のそれぞれを**レベル1の領域1**，**レベル1の領域2**と呼ぶ．このモデルの**特徴**は，編集前のテキストのすべての基準文字の間を，基本的には対等公平に2つのサイトに分け与えていることであり，両サイトの入力文字は，1つの基準文字間について見れば低位側あるいは高位側のどちらかの領域に位置づけられるという違いはあるにせよ，挿入目標位置として当初考えた基準文字間には，ともかくも挿入されるということである．これは1/1次入力からの仮定であるが，4.2節の交代入力においても前提となる考え方である．

挿入文字以外の制御文字が入力されると，このレベル1の領域内部の順位が問題となるので，各領域に属する文字を一意に表現する表示法をここで定義しておく．

[定義]

レベル1の領域の文字 $x_{i,r,j}$…エディタプロセスが管理する2点入力位置モデルのレベル1の領域に存在する文字 x は，それがどの基準文字の間の領域に属するかを示す最初の添え字として，低位側の基準文字の順位 i を継承して持つ．さらに，2番目の添え字としてそれが領域1，2のどちらに属するかを示す領域の番号 r を持つ．これは，上述のとおりサイト固定順位形の場合，サイトのアドレスに等しい．さらに，3番目の添え字として，(任意の時点において) その時点でのその領域の先頭からの順位 j を持つ．この表記法によれば，図4の状態は**図5**で表される．

ルートフォーカス…レベル1の領域は，隣接する任意の2つの基準文字の間に設けられる．あるレベル1の領域，あるいはその領域に属する文字について，そ

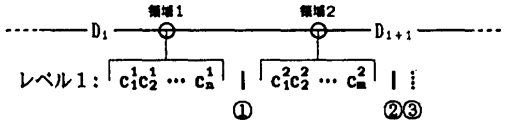


図4 2サイト一括挿入入力後の2点入力位置モデル (低位サイトでの表示)

Fig. 4 Two input position model after two site packaged insert input. (Described at the low order site.)

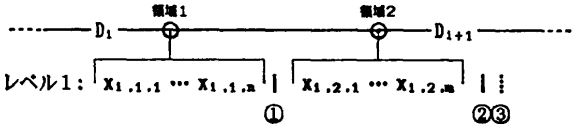


図5 2点入力位置モデルにおけるレベル1の領域の文字の表現 (低位サイトでの表示)

Fig. 5 Character description in the level 1 region of two input position model. (Described at the low order site.)

の領域の両端となっている基準文字の組を，その領域，あるいはその領域の文字のルートフォーカスと呼ぶ．また，それらが参照するルートフォーカスと呼ぶ．図5では，(D_iD_{i+1}) がルートフォーカスである．ルートフォーカスを構成する基準文字の中で，低位の文字を**低位ルートフォーカス**，高位の文字を**高位ルートフォーカス**と呼ぶ．また，ルートフォーカス (D_iD_{i+1}) とその間に設けられたレベル1の領域，およびその領域の文字を総称して，(D_iD_{i+1}) をルートフォーカスと（して参照）するレベル1の2点入力位置モデルと呼ぶ． ■

このモデルに基づいて管理されるテキストを計算機の画面に表示するときは，基準文字と共にレベル1の領域の文字を低位側から順に並べれば良い．すなわち，図5のテキストは分離形のフォーカス制御を用いた場合，エディタ画面上で次のようになる．

[低位サイトの画面]

$D_1\cdots D_i x_{i,1,1}\cdots x_{i,1,n}|x_{i,2,1}\cdots x_{i,2,m}\,\vdots\, D_{i+1}\cdots D_t$

[高位サイトの画面]

$D_1\cdots D_i x_{i,1,1}\cdots x_{i,1,n}\,\vdots\, x_{i,2,1}\cdots x_{i,2,m}|D_{i+1}\cdots D_t$

保存形制御を用いれば，次のとおりである．

[低位サイトの画面]

$D_1\cdots D_i x_{i,1,1}\cdots x_{i,1,n}x_{i,2,1}\cdots x_{i,2,m}|\,\vdots\, D_{i+1}\cdots D_t$

[高位サイトの画面]

$D_1\cdots D_i x_{i,1,1}\cdots x_{i,1,n}x_{i,2,1}\cdots x_{i,2,m}\,\vdots\,|D_{i+1}\cdots D_t$

もちろんこれは，一括入力がすべて終了した後の画面である．両サイトの入力途中，例えばそれぞれ3文字まで挿入したところでは，画面表示は次のようになる．分離形フォーカス制御ならば，

[低位サイトの画面]

$D_1\cdots D_i x_{i,1,1}x_{i,1,2}x_{i,1,3}|\,\vdots\, D_{i+1}\cdots D_t$

[高位サイトの画面]

$D_1\cdots D_i\,\vdots\, x_{i,2,1}x_{i,2,2}x_{i,2,3}|D_{i+1}\cdots D_t$

である．保存形フォーカス制御ならば，

[低位サイトの画面]

$D_1\cdots D_i x_{i,1,1}x_{i,1,2}x_{i,1,3}|\,\vdots\, D_{i+1}\cdots D_t$

[高位サイトの画面]

$D_1\cdots D_i x_{i,2,1}x_{i,2,2}x_{i,2,3}\,\vdots\,|D_{i+1}\cdots D_t$

である．因に，画面に表示されるこれら文字からキャレットを除いた文字が，バッファ文字である．

さて，ここで各サイトにおける一括入力が，挿入，削除，移動の何れの文字をも含み得る場合を考える．これらの入力文字が解釈され編集命令となったときに生成されるフォーカスは，**表6**のように整理することができる．生成フォーカスに対応する挿入位置は，挿入後の時点でのその文字の表記で表されている．なお，ここでは一括入力を考えているので，各サイトで生成されるフォーカスは相手の入力文字を含まないことに注意する．表6は，一括入力の中で基準文字の削除が行われたか否かの場合に分けて記述されている．それら各々の場合について考える前に，まず削除の実行方法について述べておかねばならない．

表 6 一括入力時の生成フォーカスと挿入位置
Table 6 Generated foci during a package input and their corresponding insert positions.

基準文字の削除なし		基準文字の削除(D_{i+1}〜D_j)あり	
生成フォーカス	挿入位置	生成フォーカス (i<j)	挿入位置
$D_i\ D_{i+1}$	$x_{i,u,1}$	$D_i\ D_{j+1}$	$x_{i,u,1}$
$D_i\ x_{i,1,s,1}$	$x_{i,u,1}$	$D_i\ x_{k,s,u,1}$	$x_{i,u,1}$
$x_{i,1,s,k}\ x_{i,1,s,k+1}$	$x_{i,1,s,k+1}$	$x_{e,s,u}\ x_{h,s,1}$	$x_{e,s,u}$
$x_{i,1,s,u}\ D_{i+1}$	$x_{i,1,s,u}$	$x_{k,s,u}\ D_{j+1}$	$x_{k,s,u}$

ただし，$i<k<j$, $i\leqq e<h\leqq j$. u はその時点の領域 s の文字の中の最大順位．挿入位置は，挿入後の時点での表記である．

削除の編集命令の実行については，それが基準文字を削除する場合と，自身が入力したレベル1の領域の文字を削除する場合とを別々に考える必要がある．基準文字は，共有者サイトからも参照されているが，レベル1の領域の文字は，共有者からの編集命令が指定するフォーカスとなることは決してないからである．まず，基準文字を削除する編集命令を実行するときは，2点入力位置モデルの中ではその基準文字に“削除”を意味するラベルを付けるにとどめる．**図6**に基準文字の D_{i+1} と D_{i+2} が削除された場合のモデルを示す．このように2点入力位置モデルの中では，削除された基準文字をすぐに除去してモデルの構成を変えるのではなく，削除が行われたことだけを記録しておく．モデルの構成を変えてしまうと，共有者からの編集命令が被削除文字をフォーカスとしていた場合に実行できなくなるからである．しかしもちろん，画面に

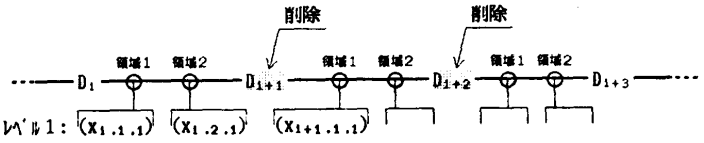


図 6 2点入力位置モデルで D_{i+1} と D_{i+2} が削除された場合
Fig. 6 Two input position model after deletion of D_{i+1} and D_{i+2}.

表示されるバッファ文字列の中からは消去し，削除の制御文字の入力と同時に画面からは消えるように制御しなければならない．次に，そのサイトで自身が入力したレベル1の領域の文字を，その後の入力で削除する場合には，その領域から実際に除去してしまってかまわない．それは，共有者サイトにその削除の編集命令が到着した時点でも，自己のサイトでの削除と何等変わるところなく行えるからである．最後に，削除後のフォーカス位置であるが，これが被削除文字の両隣の文字となることは最初に削除を実行したサイトではもちろん自明である．しかし，この削除編集命令が共有者サイトに伝達され実行されたときに，被削除文字の両隣が最初に実行したサイトの場合と同一となる保証はない．したがって，削除編集命令を共有者サイトから受信したサイトでは，削除を実行した後に，命令に付随して与えられている削除後のフォーカス位置へキャレットを移動することが必要となる．これは，次に述べるフォーカス移動の命令の解釈と同様に行えば良い．

削除に関する制御が明らかになったので，先ほどの議論に戻り，挿入と移動の編集入力の実行について考える．まず，基準文字を削除する入力が含まれなかった場合であるが，これは表6の左半分に示されている．この場合に挿入，移動の命令に伴って生成されるフォーカスは，基準文字とそれ以前に挿入した文字の組み合わせからなり，4通りがある．挿入位置（移動の場合は仮想文字が置かれる位置）は，上述のレベル1の領域の文字の定義にしたがって示されている．挿入文字の第2の添え字 s は，それを挿入したサイトのアドレスであり，$s=1$ または $s=2$ である．また第3の添え字として現れる u は，その時点のその領域の文字列中の最高順位を示すとする．それぞれの挿入位置については説明を要しないであろう．

次に，基準文字の削除を含む場合が，表6の右半分に示されている．これは，フォーカスが生成された時点以前に当該サイトが基準文字を既に削除していた場合である．この時は，生成フォーカスが，削除された基準文字の位置を挟み，複数の挿入可能位置ができる場合がある．例えば，図6では D_{i+1} と D_{i+2} が削除されているが，このとき (D_iD_{i+3}) をフォーカスとして* サイト1が挿入するならば，$x_{i,1,1}$，$x_{i+1,1,1}$，$x_{i+2,1,1}$ の3カ所が考えられる．基準文字が削除された後も，それらの間に構成されているレベル1の領域の区別は残っていると考えるわけである．一般化して言えば，もし基準文字が D_{i+1} から D_j（ただし $i<j$）まで削除された後に，D_i と D_{j+1} の間に挿入すれば生成フォーカスは (D_iD_{j+1}) となるが，この場合の挿入位置は削除された文字数と同数だけ増えて，$x_{i,s,1}$ のほかに $x_{i+1,s,1}\cdots x_{j,s,1}$ が可能であると考えられる．表6の割当は，この中で最低位，つまり図6であれば，$x_{i,1,1}$ を用いることを与えている．このように複数の挿入可能位置が考えられる時に最低位を用いることを，基準文字の削除があった場合の規約としておく．

表6の生成フォーカスは，共有者サイトに到着した時に見かけ上，非隣接状態（例えば $x_{i,2,1}$ が間に入るなど）や基準文字削除による高位フォーカスあるいは低位フォーカスが欠如した状態となる場合があるが，本モデル上での挿入位置は表6で一意に定義されているから，両サイトで必ず同一の文字順位を持つテキストを構成できる．もちろんこの際には，基準文字が削除されてもモデル全体の構造は変わらないと考えておくことが必要であり，画面表示の時には削除された基準文字は表示せずに，残った基準文字とレベル1の領域の文字を表示すれば良いわけである．これに対して，入力された当該サイトの文字が削除される時は，先述のようにレベル1の領域からその文字を除去して構わない．（ただし，後に述べるように，後者は一括入力の場合についてのみ可能である）

さらに挿入後のキャレットの位置であるが，基本的には挿入文字の直ぐ高位隣側，移動の場合は仮想文字が挿入されるべき位置の高位側が，これまでの議論から妥当であることが分かる．その例外は，保存形のフォーカス制御を行った場合において，低位サイトがレベル1の領域の最高位に挿入したとき（図5では $x_{i,1,n}$ を挿入したとき）であり，この時だけは①の位置ではなく，高位サイトの文字の領域2の直ぐ高位側，すなわち②の位置となる．したがって，2点入力位置モデルを用いた共有テキストの編集制御を考えると，保存形のフォーカス制御は余分な場合分けが必要になることが分かる．したがってこれから先の議論では，フォーカス制御は分離形だけを考える．

2サイトで4文字ずつ入力した場合の1つの例を，付録Aに挙げておく．

この2サイト一括入力は，両サイトの編集命令が両エディタプロセスにすべて到着した時点で終了する．

* 操作者が，画面に表示されている文字 D_i と D_{i+3} の間に挿入しようとしている時である．

この時，モデルの中で分離していたレベル1の領域の入力文字列と基準文字の間の区別をなくし，最初に与えられたと同様の1次元のテキストとして**更新**する．この時，削除の印が付けられていた基準文字は，もちろん捨て去られ，エディタ画面に表示されている1次元の文字列それ自身が内部モデルの構造となる．つまり，更新されたテキストの文字は，再びそれ全体が以後の編集に対する基準文字となる．この更新によって，両サイトのテキストは編集途中で互いに相違を含む状態を経た後で，再び完全に一致した同一のテキストとなる．

各サイトにとっての更新の時期は，相手からの編集命令の到着が終了した時である．しかし，エディタプロセスは，通常は相手からの編集命令の到着を待っているだけなので，何れが最後の編集命令であるかは分からない．したがって，この2サイト一括入力の更新の実行は，ここでは概念的な理解にとどめる．4.2節の交代入力において更新時期の決定方法が与えられる．

4.2 2サイト交代入力時の共有テキスト

2サイト交代入力とは，当該サイトの入力と共有者サイトの編集命令の到着が交互して継続する入力形式である．**図7**に2サイト交代入力において編集命令が両サイトのエディタに到着するタイミングを示す．図1と同様，縦の下向き方向が時間の経過であり，縦の太線はエディタプロセスへ編集命令が継続的に到着している部分である．編集命令伝達メッセージを表す矢印は，図7では特徴的な一部についてだけ記載して大部分は省略している．

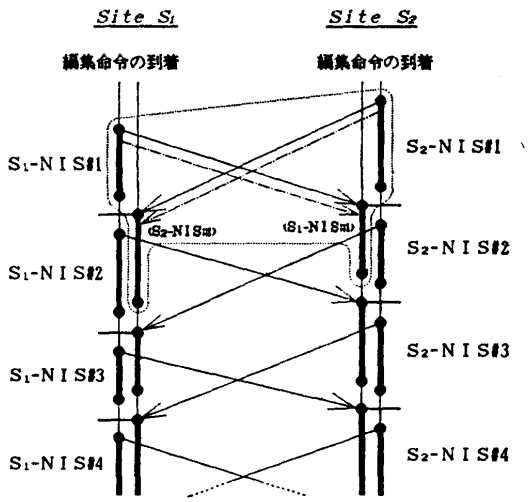


図7 2サイト交代入力時の編集命令の到着タイミング
Fig. 7 Edit command arrival timing of two site interleaved input.

交代入力では，両サイトからの編集命令が入り交じってエディタプロセスに到着する場合を考えるが，まずこれらの編集命令の系列をグループに分けておく．各サイトの自分自身の編集命令については，編集開始後の最初の入力が始まってから，共有者サイトから最初の編集命令が到達するまでを1つのグループとする．このグループは，共有者からの編集命令によってそれぞれが分離されることなく当該サイトにおいて引き続いて実行される最初の一群の命令なので，これをレベル1の **NIS** (Non-Interleaved Set) と呼ぶ．サイト S_1，サイト S_2 についてのレベル1の NIS をそれぞれ **S_1-NIS #1**, **S_2-NIS #1** と書く．NIS #1 は，共有者からの最初の編集命令の到着によって自身の命令系列から区切られているので，図7においても共有者からの編集命令の先頭が NIS #1 を命令列から切り出すように記述してある．こうして切り出された NIS #1 は，共有者からの編集命令が到着する以前に入力された編集命令の集合であり，それゆえに両者の NIS-#1 すなわち S_1-NIS #1 と S_2-NIS #1 は編集開始前の共通のテキストに対して対等な効力を持つと考えられる．この点は，3.2節で扱った2サイト一括 n/m 次入力と同じである．両サイトの NIS #1 とそれらが共有者サイトに到着した部分だけを取り出してみれば，その配置が一括入力そのものであることからも，それが分かる．(図7の点線で囲まれた部分) したがって，NIS #1 に属する編集命令の処理は，一括入力の場合とほぼ同様に行って良いことになる．これについては，後にさらに詳しく述べる．

NIS #1 の切り出しで注意すべきこととして，当該サイトで最初の1文字を入力する以前に，共有者サイトからの編集命令が到着した場合の処理（先着編集命令の処理）がある．この場合は，実は2サイト同時入力ではなくて，1サイトだけが単独で入力したと同じことであるから，両サイトにおいてその先着編集命令は全く同じ条件で実行される．したがって，これをレベル1の NIS，つまり並行に入力された最初の命令群に含めて扱う必要はない．先着編集命令は，受信サイトではこれを単純に（2点入力位置モデルを意識せずに）直ちに実行すれば良い．しかし，送信サイトでは先述のように，相手からの編集命令が到着するまでを NIS #1 と考えているので，自分が送信した（先着）編集命令を NIS #1 に含めない対策が必要となる．このためには，受信サイトが1文字目の入力後に

それを編集命令として送信する時に，それまでに到着している先着編集命令を編集命令に加えて送れば良い．あるいは，受信サイトが送信サイトに対して，先着編集命令を受信したことを，その時点で1つ1つ返答しても良い．先着編集命令を送信したサイトでは，それらに対応する文字についてだけ，単独のサイトで編集したと同じように基準文字の中に組み入れる．そしてここで得られたテキストが，その後の両サイトの並行的入力に対して共有される初期状態となる*．

各サイトの自身の編集命令は，NIS #1 に引き続いて自身のエディタプロセスに到着するのであるが，図7から両サイトの NIS #1 と共有者サイトに到着したそれら（点線部）を取り除いてみると，最初に NIS #1 が切り出される前と相似な形態が残ることが分かる．したがって，残った部分について NIS #1 を切り出したと同様の手続きを経れば，これを NIS #2 として切り出すことができる．このように切り出しの作業を繰り返せば，すべての編集命令の系列をレベル付けられた NIS の集合に分けることができる．エディタプロセスが実際に編集命令を受信するときに，それが NIS の何レベルであるかを知るためには，共有者サイトが編集命令伝達メッセージを送信する時に，その命令が現在どのレベルにあるかの情報を付加しておけば良い．自分自身のサイトの編集命令の NIS レベルは，最初が1であり，共有者から NIS レベル1の最初の命令を受けた時点以降をレベル2とする．さらに共有者からレベル2の最初の命令を受けた時点以降をレベル3とするというように，レベル n の最初の命令の受信により，以降の自身の編集命令をレベル $n+1$ とすれば良い．

さて，到着した編集命令の実行であるが，2章の定義で述べたとおりエディタプロセスは逐次形であるので，編集命令の実行は1命令ずつ行われる．図7において，共有者サイトからの NIS #1 を受信しつつある時には，自己のサイトの NIS #2 が並行して到着するのであるが，両者は到着の順番に従って実行される．この場合の処理を，サイト1を例として考えてみよう．

先に述べたように，並行に入力される S_1-NIS #2 に先立つ S_1-NIS #1 と，共有者の S_2-NIS #1 は，編集開始時の共有テキストに対して対等に編集する権利を持っていると考えられる．これらはどちらも編集前のテキストとそれら自身によって挿入された文字だけを見ながら入力された（文字から導かれた）編集命令だからである．そして，両者の NIS #1 がすべて到着した後のテキストの構造は，NIS #1 だけの処理について考えれば，4.1節で議論した一括入力に対する2点入力位置モデルと同様に決定されるのが妥当であろうことが分かる．このためには，共有者のレベル1の命令 S_2-NIS #1 の実行は，自己サイトのレベル2の編集命令 S_1-NIS #2 とは全く無関係に，同じレベルの S_1-NIS #1 だけと干渉し得ると考えて，先に述べた2点入力位置モデルに対して行われねばならない．これは，S_2-NIS #1 の実行が，並行に到着する自己サイトのレベル2の編集命令 S_1-NIS #2 に優先して行われるということである．優先とは，自己サイトのレベル2の編集命令が，基準文字およびレベル1の領域の文字列を変えることがあっても，共有者サイトのレベル1の編集命令 S_2-NIS #1 を実行する際にはその効果を無視して進めるということである．共有者サイトのレベル1の編集命令がこうして優先されるべき理由をさらに分かりやすく言えば，入力の順序から考えて S_1-NIS #1 が S_1-NIS #2 に優先するのは明らかだが，それならば S_1-NIS #1 と対等と見なされる S_2-NIS #1 も当然 S_1-NIS #2 に優先しなければならないということである．したがって，レベル1の編集命令に基づいて構成された2点入力位置モデルのレベル1の領域までの部分は，共有者からのレベル1の編集命令が完結すれば，レベル2の編集命令の効果に優先してその時点で更新してよいことになる．レベル1の編集命令の到着が終了したことは，レベル2の編集命令の到着により分かる．更新の具体的な手続きについては，本節の最後に述べる．

これに対してレベル2の編集命令の実行，つまり S_1-NIS #2 の実行においては，それが入力される時点では共有者サイトのレベル1の編集命令が必ずしもすべて到着していない．にもかかわらず上述のように，それらすべてのレベル1の編集命令が実行された結果を優先することが要求されている．したがって，S_1-NIS #2 に属する編集命令の実行に共通することは，それがどの時点で入力されたものだとしても，レベル1の編集命令が挿入するすべての文字（既に挿入された文字のみならず到着しつつある，またはまだ到着していない挿入文字）についても，これを最終的には認める前提で実行されねばならないということである．

* この議論から分かるように，編集命令が同時に入力されるとは，図7において編集命令伝達メッセージを表す矢印が互いに交差することである．交差しない編集命令は，両サイトで同じ条件で実行される直列化された，並行処理を必要としない命令である．どれだけの編集命令が‘同時’となるかは，入力の頻度と通信遅れ時間によって決まる．

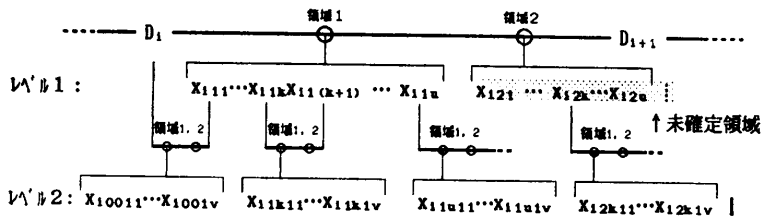


図 8 2サイト交代入力時の2点入力位置モデル（ただし，低位サイトから見た表示である．また，領域の文字の添え字間のコンマは省略してある．$1\leqq k<u$）

Fig. 8 Two input position model for two site interleaved input. (Described at the low order site. Commas between subscripts of a character in the region are omitted. $1\leqq k<u$)

このためには，レベル1の領域の文字が既に更新されて基準文字となっていると見なして，これに対して次のレベルの2点入力位置モデルを構成することが必要である．ここで再び2点入力位置モデルを再帰的に用いることができるのは，レベル2の編集命令系列も，レベル1と相似な取り出し方がなされているからである．しかしもちろん，基準文字と見なすべきレベル1の領域の文字がその時点で全部は確定していないことに対応して，モデルを若干修正することは必要である．

図8に2サイト交代入力の場合の2点入力位置モデルを示す．ただし，これは低位サイトから見た状態の表現であって，共有サイトからレベル1の編集命令 S_2-NIS #1 を受信しつつ，自らのレベル2の編集命令 S_1-NIS #2 を実行している途中の段階を表している．ただし，この図に至るまでの入力された S_1-NIS #2 の命令の種類，その結果としてのルートフォーカスの取り方とキャレットの位置は1つの例である．以下では特に断らない限り，レベル1の2点入力位置モデルを論じるに当たって準備した定義を，レベル2以上のレベルにも適用できると考える．また，レベル2の領域の文字の記述方法は次のとおりである．

[定義]

レベル2の領域の文字 $x_{i,r,j,s,k}\cdots$ 2点入力位置モデルのレベル2の領域に存在する文字 x は，最初の3つの添え字として，その文字が参照する低位ルートフォーカスの添え字をそのまま継承する．ただし，第2，第3の添え字は，文字 x の低位ルートフォーカスがレベル1の文字でなく基準文字の場合には，両者とも0となる．これは，図8の最低位のレベル2の領域を見れば分かるように，レベル1の領域を参照しないので領域の番号およびその中での順位がないからである．さらに，4番目の添え字としては，その文字がレベル2の領域1，2のどちらに属するかを示す領域の番号 s を持つ．これは，レベル1の領域と同様で，入力サイトのアドレスに等しい．さらに，5番目の添え字として，任意の時点におけるその時点でのその領域の先頭からの順位 k を持つ．なお，図8および表7におけるレベル2の領域の文字の5番目の添え字 v は，その領域の最高順位を示す．■

図8が導かれる最初の段階，つまり NIS #1 を入力した際に生成されるフォーカスとその挿入位置は，先述の表6に示したとおりである．レベル2の編集命令が入力されたときに生成され得るフォーカスを**表7**に示す．ただし，表7も低位サイトにおける場合だけを記載している．また生成フォーカスに対応する挿入位置は，挿入後の時点での表記である．表7は左右，上

表 7 2サイト交代入力において，サイト1（低位サイト）がレベル2の編集命令 NIS #2 を入力する時の生成フォーカスと挿入位置

Table 7 Generated foci of NIS #2 and corresponding insert positions at the low order site during two site interleaved input.

基準文字及びレベル1の領域の削除文字なし			同 削除文字あり		
生成フォーカス	ルートフォーカス	挿入位置	生成フォーカス (i<j)	ルートフォーカス	挿入位置
D_iD_{i+1}	D_i-	X_{10011}	D_iD_{j+1}	D_i-	X_{10011}
D_iX_{111}	D_iX_{111}	↑	D_iX_{j11}	↑	↑
D_iX_{121}	D_i-	↑	D_iX_{j21}	↑	↑
$X_{11k}X_{11(k+1)}$	$X_{11k}X_{11(k+1)}$	X_{11k11}	$X_{11k}\,\$_1$	$X_{11k}-$	X_{11k11}
$X_{11u}X_{121}$	$X_{11u}-$	X_{11u11}	$X_{11u}\,\$_1$	$X_{11u}-$	X_{11u11}
$X_{11u}D_{i+1}$	↑	↑	$X_{11u}D_{j+1}$	↑	↑
$X_{12k}X_{12(k+1)}$	$X_{12k}-$	X_{12k11}	$X_{12u}\,\$_1$	$X_{12u}-$	X_{12u11}
$X_{12u}D_{i+1}$	$X_{12u}-$	X_{12u11}	$X_{12u}D_{j+1}$	$X_{12u}-$	X_{12u11}
D_iX_{10011}	D_iX_{111}	X_{10011}	$D_i\,\$_2$	D_i-	X_{10011}
$X_{11k}X_{11k11}$	$X_{11k}X_{11(k+1)}$	X_{11k11}	$X_{11k}\,\$_2$	$X_{11k}-$	X_{11k11}
$X_{11u}X_{11u11}$	$X_{11u}-$	X_{11u11}	$X_{11u}\,\$_2$	$X_{11u}-$	X_{11u11}
$X_{12k}X_{12k11}$	$X_{12k}-$	X_{12k11}	$X_{12k}\,\$_2$	$X_{12k}-$	X_{12k11}
$X_{12v}X_{12v11}$	$X_{12v}-$	X_{12v11}	$X_{12v}\,\$_2$	$X_{12v}-$	X_{12v11}
$X_{1ef1e}X_{1ef1(e+1)}$	*	$X_{1ef1(e+1)}$			
X_{1ef1v} ?	*	X_{1ef1v}	←	←	←

ただし，$1\leqq k<u$, $e\in\{0,1,2\}$, $0\leqq f\leqq u$, $0\leqq g<v$. u, v は各々レベル1，2の領域の最高順位．"?" は，低位フォーカスより高位にある任意の文字を表す．"*" は生成フォーカスの添え字に対応する適切なルートフォーカス．"$\$_1$" は，同じ行の"削除文字なし"に記載されたもの以外で，低位フォーカスより高位にある任意のレベル1の領域の文字．"$\$_2$" は，同じく任意のレベル2の領域の文字．

下の4つの部分に分割されている．左半分が，NIS #2の編集命令の中に，基準文字とレベル1の領域の文字を削除する命令が含まれなかった場合であり（“削除なし”と略記），右半分はそれが含まれる場合（“削除あり”と略記）である．さらに，破線より上半分が，NIS #2 の編集命令入力時の生成フォーカスが，既に入力したレベル2の領域の文字を含まない場合であり，下半分が含む場合である．

左上の場合，つまり“削除なし”でレベル2の領域の文字が生成フォーカスに含まれない場合では，ルートフォーカスが確定するのは2つの場合だけである．これ以外では，生成フォーカスの間にレベル1の領域2を挟むことになる．レベル1の領域2は，この時点では共有者からまだ到着しつつある編集命令が文字を挿入する可能性のある未確定の領域である．したがって，レベル1の領域2を挟むフォーカスが生成された時，その高位フォーカスをそのまま高位ルートフォーカスとするとルートフォーカスが隣接文字でなくなる場合が生じる．それならばむしろ，高位ルートフォーカスは確定せずに，低位ルートフォーカスだけの片持ちでレベル2の領域を仮に構成しておく方がモデルを単純化できる．表7ではこの場合の高位ルートフォーカスを“-”で表している．そしてレベル1の領域の文字が基準文字として更新された時点で，その時低位ルートフォーカスに隣接する文字を高位ルートフォーカスと考えれば良い．

表7の左下，つまり“削除なし”でレベル2の領域の文字を生成フォーカスに含む場合は，既に存在するレベル2の領域に挿入することになる．したがって，ここに記されたルートフォーカスはこの時新たに設定されるものではないが，参考として示す．レベル2の領域の文字が低位フォーカスならば，高位フォーカスが何であってもレベル1の場合と同様，低位フォーカスに隣接する高位側が挿入位置となる．表中では低位フォーカスより高位にある可能なすべての文字を？で表している．高位フォーカスがレベル2の領域の文字である場合は，何れもその挿入位置は高位フォーカスの低位側である．なお，＊印は生成フォーカスに対応する適切なルートフォーカスを示す．

表7の右半分の場合，つまり“削除あり”では，生成フォーカスの間に未確定領域を必ず挟むのでルートフォーカスはすべて片持ちである．この時は，一括入力と同様に複数の可能な位置の中で最低位を挿入位置とするが，それは低位フォーカスに隣接する高位側である．生成フォーカスとしては多くの可能性があるので，レベル1の領域の文字からなる高位フォーカスをまとめて $\$_1$ で示した．またレベル2の領域の文字からなる高位フォーカスを $\$_2$ で代表した．

表7では場合に分けてルートフォーカスと挿入位置をすべて示したが，これらに共通して言えることは，ルートフォーカスと挿入位置を決定するのは，低位の生成フォーカスのみであること，また高位ルートフォーカスは，更新される時点で初めて決まるものもあるだけでなく，たとえ入力の時点で決まってもそれが後に参照されることは決してないことが分かる．したがって，挿入位置を決める手順を要約すれば，与えられた低位フォーカスがレベル2の領域の文字でないときは，それを低位ルートフォーカスとしたレベル2の領域の最低位が挿入位置である．また，低位フォーカスがレベル2の領域の文字である場合は，それは自己サイトの文字であるから，レベル2の領域においてその文字に隣接した高位側が挿入位置である，と表すことができる．フォーカス制御，すなわちキャレットの位置の決定も，以上の文字挿入の位置に従って 4.1 節と同様に行う．つまり，挿入文字の直ぐ高位隣側，フォーカス移動の場合は仮想文字が挿入されるべき位置の高位置である．なお，ここでは低位サイトを例に示したが，高位サイトの場合も考え方は同じであり，表7に対応する挿入位置も簡単に導くことができる．

以上で2サイト交代入力における挿入・移動の処理が与えられたので，次に削除の処理について述べなければならない．文字の削除を行う場合には，それがいかなるレベルの編集命令によるものだとしても，またいかなるレベルの文字が削除の対象であろうと，被削除文字にラベル付けするに止め，モデル自体の構造は保持しておく（もちろん，エディタ画面からは削除される）．これは，2サイト交代入力においては，一時でもバッファに存在した（表示された）文字はレベルの如何に係わらず，それらはすべて後で共有者サイトから参照される可能性があるからであり，また，既にそれが2点入力位置モデルのルートフォーカスとなっているならば，これを削除することはそのモデルを宙吊りとする結果となるからである．したがって，現在ルートフォーカスとして参照されている文字はもちろんのこと，将来参照される可能性のある文字は，削除の対象となってもこれをすぐにはモデルから抹消しないでおく．そして，参照の可能性がすべてなくなった

時に，モデルから完全に除去する．参照の可能性がなくなる時点は，削除される文字がどのレベルの文字であるか，そしてどのサイトの何れのレベルの編集命令によって削除されたかによって異なる．したがって，被削除文字に付ける削除ラベルとしては，それを削除した削除編集命令の発行サイト，およびその編集命令のレベルの番号を付加しておく．

最初の更新は，両サイトのレベル1が共にすべて到着した時点で行われるが，その時点で削除のラベルを持つ文字が，どのレベルから参照されている，あるいは将来参照される可能性があるかをまとめたのが，**表8**である．以下の議論は，図7を参考にしながら考えると分かりやすい．まず，最初の更新の時点で存在し得る文字の種類は，表8で被削除文字として示されている4種類，つまり基準文字，自己サイトのレベル1，レベル2の領域の文字，共有者サイトのレベル1の領域の文字である．そして，これらの文字に削除のラベルを付けることができる編集命令の NIS レベル（と発行サイト）が3通り，これらが表8における削除編集命令のレベルとして，自己サイトのレベル1，自己サイトのレベル2，共有者サイトのレベル1と記されている．これらの組み合わせとして，削除のラベルを付けられた文字が，如何なるレベルの領域の文字からルートフォーカスとして参照され得るか（更新の時点以前に設定された参照だけでなく，将来共有サイトで参照される可能性も含めて）がこの表の意味であり，“自1”とは自己サイトのレベル1の領域の文字を表す．“自2”は同じくレベル2の文字，また“共1”とは，共有者のレベル1の領域の文字である．さらに，○印はそれらの中で最も時期的に遅い時点に発生する参照主体文字であることを示す．

表8 削除ラベルを付けられた文字をルートフォーカスとして参照し得る（領域の文字の）レベル
Table 8 The level of a character in a region that can refer the delete labeled character as a root focus.

削除編集命令の NIS レベル ＼ 被削除文字	基準文字	自己サイトのレベル1の領域の文字	自己サイトのレベル2の領域の文字	共有サイトのレベル1の領域の文字
自己サイト・レベル1	自1 共1 ○共2	○共2	—	—
自己サイト・レベル2	自1 自2 共1 共2 ○共3	自2 共2 ○共3	○共3	自2 共2 ○共3
共有者サイト・レベル1	自1 ＊自2 共1	—	—	＊自2

表8の幾つかの欄について説明を加える．まず，左上隅の欄，つまり基準文字が自己サイトの NIS レベル1の削除編集命令で削除されていた場合は，自己のサイトでこれを参照できるのは，削除以前に自己サイトのレベル1の挿入されていた文字だけである．当然，レベル2以降の入力時には削除済みであって，被削除文字がレベル2以降の領域の文字から参照されることはない．さらに，共有者サイトにおいては，その基準文字に対する削除命令が自己サイトから到達するまでは，共有者サイトでの参照が起こる．共有者サイトに自己サイトのレベル1の削除編集命令が到着するのは，共有者サイトが NIS レベル2の編集命令を入力している時点であり，したがってそれまでに起こる参照は，共有者のレベル1の領域の文字によるものと，その削除編集命令が到着するまでに入力された共有者のレベル2の領域の文字によるものである．注意すべきは，この共有者サイトにおける参照は，被削除文字を参照している挿入文字が自己サイトに到達するまでは，参照が起こるか否かが分からないことで，したがってこの場合，共有者の NIS レベル2の編集命令がすべて到着するまでは，モデルから被削除文字を抹消できない．これが○印を付けた意味である．次に左下の欄，つまり基準文字を共有者サイトのレベル1の削除編集命令で削除した場合は，削除以前にこれを参照し得たのは自己サイトのレベル1，レベル2（の一部），そして共有者サイトでは削除編集命令以前に挿入されたレベル1の文字である．この中では，自己サイトのレベル2の文字が最も遅い時期に入力されたことになるが，更新の時点でこの参照があるか否かは確認できる．もしなければ，これ以外はレベル1の領域の文字であるから，更新と同時に被削除文字は抹消できることになる．このように更新の時点で参照の有無を確認できる場合には＊印を付けた．次に，自己サイトのレベル1の領域の文字を自己サイトのレベル1の削除編集命令で削除した場合であるが，この場合は自己サイトが自ら削除した文字を次のレベル以降が参照する可能性はない．共有者サイトでは，被削除文字が挿入文字として到着してから，削除編集命令が到着するまでに，共有者サイトのレベル2の編集命令によって参照される可能性がある．そして，参照は起こ

るとすればこのレベル2だけである．次に，共有者サイトのレベル1の領域の文字が被削除文字の場合であるが，この時は，自己サイトのレベル1の編集命令が削除するということはあり得ない．共有者サイトのレベル1の領域の文字が到着したときには，自己サイトはレベル2となっているからである．したがって，自己サイトのレベル2の編集命令が削除する場合はあり得る．この命令で削除された被削除文字は，自己サイトの削除前のレベル2の挿入文字が参照する可能性がある．また，共有者サイトでは自己サイトで実行された削除編集命令が到着するまでは，被削除文字を参照する可能性がある．この自己サイトの NIS レベル2の到着は，共有サイトの NIS レベル3と並行しているから，参照の可能性があるのは，共有者のレベル2およびレベル3の領域の文字であり，したがって共有者の NIS レベル3の編集命令が自己サイトに到着し終るまでは被削除文字は，モデルから抹消できないことになる．これ以外の欄の解釈も同様である．以上を要約すれば，自己サイトで削除を行ったならば，その削除編集命令が共有者サイトに到着するまではそこで参照が起こるから，到着した時点の共有者サイトのレベルに属する編集命令がすべて自己サイトに到着するまで，被削除文字は抹消できないということ，さらに，共有者サイトで削除が行われたならば，その被削除文字を自己サイトで参照しているレベルが更新で基準文字となる時まで抹消できないということである．またもし，同一文字に自己サイトと共有者サイトの両方の削除ラベルが付いた場合では，各場合について調べれば保持すべき期間は容易に求まるが，表8の両該当欄の最大参照レベルまで考えれば十分である．

最後に，2サイト交代入力における更新手順であるが，既述のように最初の更新はレベル1の編集命令が共有者サイトからすべて到着した時点（実質的には共有者サイトのレベル2の最初の編集命令が到着した時点）に行われる．この更新によってレベル1の領域の文字が新たに基準文字に組み込まれる．組み込みの順位は，ルートフォーカスとなっている基準文字の間に，領域1の低位から高位，領域2の低位から高位までの順である．この時，基準文字およびレベル1の領域の文字に削除のラベルが付けられている場合は，そのラベル付けされた被削除文字が，レベル1の領域以外の文字からは参照されていない状態，あるいは将来も参照されることはない状態となった時点でモデルから抹消する．最初の更新時であれば，それが可能なのは表8の最下段で，かつ自己サイトのレベル2の領域の文字からの参照がない場合だけである．それ以外の抹消できない被削除文字については，それが最終的に抹消できる時期を該当欄から調べてラベルに記載しておく．そして，基準文字の中に削除ラベル付きのまま残しておく．第2回目の更新時は，レベル2の編集命令が共有者サイトから到着し終わったときであるが，この時までに新たに削除のラベルが付けられた文字の参照レベルも，表8を繰り返して用いることで決められる．2回目の更新では，表8の中でレベルを表している数字をすべて1ずつ増やして考えれば良い．n 回目の更新ならば $(n-1)$ 増やせば，初回と全く同じ議論が再帰的に成り立つ．

4.3 多サイト交代入力時の共有テキスト

まず3サイトの場合について，2サイトと同様の形式のタイミングチャートを**図9**に示す．各サイトの上下方向の3本の線は，それぞれ左からサイト1，2，3で入力された編集命令の到着を表している．この図では，メッセージ伝達の矢印が多すぎて見にくくなることを避けるため，あるサイトから送信された編集命令伝達メッセージが，他の2つの共有者サイトにおいては同時に受信されるように記述されている（2つの共有者サイトが同時に受信する時点は水平な破線で示されている．このような仮定は，サイトが近接してい

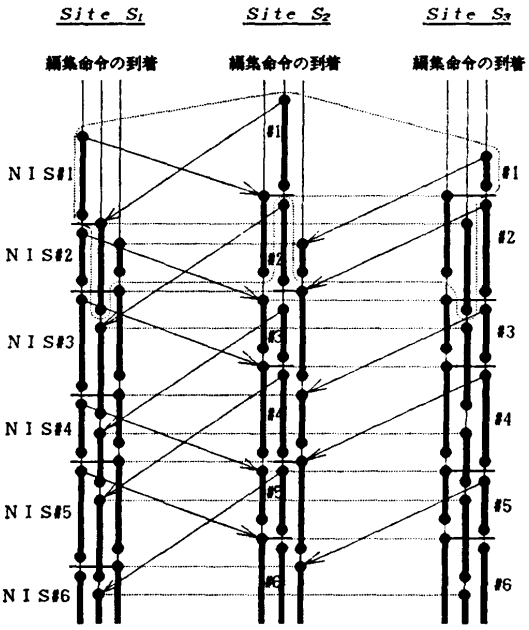


図9 3サイト交代入力時の編集命令の到着タイミング
Fig. 9 Edit command arrival timing of three site interleaved input.

て通信路が単一のバス形構造を持つ場合に成り立つ). しかしながら以下の議論においては，サイト間の結合がこの例のようなバス形であって，かつそれゆえに通信遅れ時間が等しいことは必ずしも必要な条件ではなく，2章で与えた仮定つまり，あるサイトから送信されたメッセージが，そのサイトのメッセージについては送信された順序で共有者サイトに到着することだけが満足されていれば良い．この理由は以下の記述において明らかとなる．

3サイトの編集命令の系列についても，2サイトの場合と同様にまずNISレベルへ分解する．4.2節で述べたように，最初のNISすなわちレベル1のNISは，それが他のサイトからの編集命令の到着より以前に入力されたがゆえに，他サイトで同様の状況で入力された最初のNISと対等と見なされる．したがって，図9でも他のサイトから最初の編集命令が到着する時点までの編集命令がNIS#1である．最初の編集命令は，2つの共有者サイトから到着するが，その中でより早く到着した方によって，レベル1のNISが切り出される．この切り出しは，どのサイトでも同様に行われ，それぞれのサイトでNIS#1が決定される．3つのサイトのNIS#1を，図9においても図7と同様に破線で囲んであるが，この部分は以降のレベルに属する編集命令に優先して実行され，また更新される．さらに，レベル2以降の編集命令系列についても，どちらかの共有者サイトから現在の自分のレベルと同じレベルのNISを受信した時点から，自身の編集命令は次のレベルとなる．この手続きにより，各サイトの編集命令の系列はすべてレベル付けされる．

このレベル付けにおいて注意すべきことは，レベルを切り出した編集命令が，どちらの共有者サイトの命令であるかの区別は必要ないことで，これによって図9を書く際に使用された通信時間に関する仮定は緩和しても良いことになる．つまり，レベル1について考えてみると，各サイトのNIS#1がどのサイトからの編集命令で切り出されたかは問題ではない．というのは，図9ではサイト1とサイト2はお互いにNIS#1を切り出し合っているが，サイト3はこの場合は全く偶然にサイト1によって切り出されている．しかし，もしネットワークがバス形でなく，かつサイト1の編集命令の到着が遅れたために，切り出しがサイト2の編集命令に因ったとしても，サイト3のNIS#1が少々その命令数を増すだけであり，3つのサイトのNIS#1が相似な切り出し形態を後に残しつつ，破線で囲むように分離できることには変わりない．したがって，異なるサイトからの編集命令伝達メッセージの到着順が受信サイトごとに異なっていても，NISレベルへの分解が行われること自体については影響しない．これは，サイト間の編集命令伝達メッセージの通信遅れ時間が，何れの2つのサイトを注目するかによって異なること，また特定のサイト間の通信遅れ時間が常に一定である必要はないこと，を許容するということである．

また，先着命令の処理については2サイトの場合と同様な状況，つまり，あるサイトからの編集命令が，他のすべてのサイトが何も入力する以前にそれらのサイトに到着した場合には，これをNIS#1に含めないことが可能である．そうでない場合，つまりサイトAでは入力が行われる以前にサイトBから編集命令が到着したが，それ以外のサイトでは既に入力が始まってからサイトBの編集命令が到着したならば，サイトAでは自己のレベル1の編集命令は該当文字なしとして，以降のレベル進行に従うことにしなければならない．

テキストの更新は，各サイトにおいてすべてのサイトからレベル1の編集命令が全部到着した時に行われる．レベル1の編集命令が揃うまでに並行に入力されるNISのレベルは，2サイトでは2レベルだけであったが，3サイト以上では，1つでも通信時間あるいは通信処理等が遅いサイトがあれば，そのサイトのレベル2のNISが到着するまでに進行した全レベルを後に述べるモデルの中で保持しなければならない．図9の場合では，3つのサイトの通信遅れ時間がほぼ同じで，最大3レベルが並行して入力されているので，一時点において3つのレベルまで保持することが必要である．そして，その時点の最低レベルの編集命令がすべてのサイトから到着した時に，そのレベルを2サイトの場合と同様に更新する．

3サイト交代入力の場合のテキストのモデルを**図10**に示す．ただし，図10はサイト1における編集途中の段階の表現である．各サイトの入力文字は，ルートフォーカス間に設けられた3つの領域に入力されるので，これを3点入力位置モデルと呼ぶ．レベル1，レベル2の編集命令による生成フォーカスに対応する入力位置は，それぞれ表6，表7と同様である．ただし，領域を表す添え字は1から3までをとることになる．また，レベル3以降の編集命令による生成フォーカスに対応した入力位置は，4.2節のレベル2の手順の要約と同様である．つまり，与えられた低位フォー

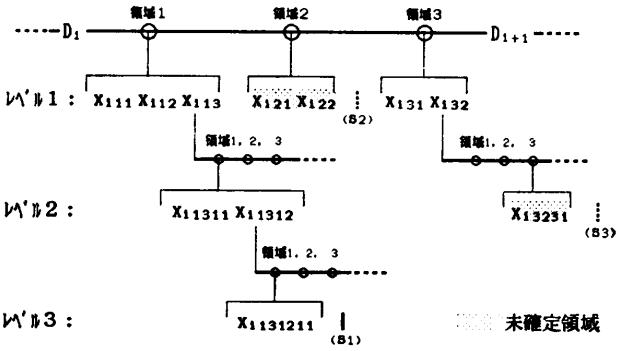


図 10 3サイト交代入力時の3点入力位置モデル（サイト1での表示）

Fig. 10 Three input position model for three site interleaved input. (Described at the site No. 1.)

カスがその編集命令のレベルの領域の文字でないときは，それを低位ルートフォーカスとした3点入力位置モデルを構成し，入力サイトの該当領域の先頭に位置づける．低位フォーカスが当該レベルの文字であれば，その文字の隣接高位側が挿入位置である．なお，図 10 は初期フォーカスを (D_iD_{i+1}) として分離形のフォーカス制御を行い，かつ挿入だけが行われた場合を示している．

先にサイト間の通信遅れ時間が等しい必要はないことを述べたが，その通信遅れ時間の不均衡によって2サイトでは生じなかった新たな場合が発生する．それは，あるサイトにおいて到着した編集命令がフォーカスと指定している文字が他サイトの挿入文字であって，かつそれが当該サイトに未到着の場合である．編集命令伝達メッセージの到達時間がサイト間ごとに異なると，メッセージの到着順が逆転することも有り得るわけであるが，この時は先に着いた編集命令の処理を未到着文字が到着するまで単純に遅らせれば良い．

削除の実行に関しては，2サイトの場合と同様に，被削除文字にラベル付けをするだけですぐに抹消はしない．2サイトの場合と異なる点は，編集命令伝達メッセージの到達時間がサイト間ごとに相違することを前提とするがゆえに，表8のように参照し得るレベルを特定することができなくなることである．そこで，自己サイトで削除を行ったならば，その削除編集命令をすべての共有者サイトに送信するが，共有者サイトからは到着後に確認返答 (acknowledge) を送ってもらうことにする．ただし，この返答は全サイト宛のマルチキャストとしておく．これにより自己サイトでは，返答をもらったサイトからの参照が以後なくなることを知るわけで，すべてのサイトからの返答を得た後，最初の更新において被削除文字をモデルから抹消することが可能となる．さらに，共有者サイトでも，その削除編集命令が自分以外のすべてのサイトにも到着したことを，各サイトからの返答を受信することにより確認できる．そうすればもはや，他のサイトからもそれを参照する編集命令は来ないことが分かるので，自己サイトで参照しているレベルが更新で基準文字となる時に被削除文字を抹消するということになる．削除によって文字が完全に抹消されるまでには，返答を待つだけの時間的な経過が必要となるが，これはあくまでもモデルの内部構造の問題であって，エディタ画面上ではもちろん，削除編集命令の受信と同時に対象文字は除去される．

さて以上の議論では，3サイトであることが機構上に特別な制限を与えていない．したがって，同様な考え方に基づいて，4サイト以上の構造化も可能である．多サイト交代入力のモデルの構成法をまとめると次のようになる．ここで，共有エディタを使用する全サイト数を N とし，任意のサイトのアドレスを s で代表する．また，サイト S_s で現在入力中の NIS レベルを L_s とする．

①任意のサイトの入力文字から導かれた編集命令の NIS レベルは，他サイトから現在のレベル L_s と同じレベルの編集命令を受信した時点以降を L_s+1 レベルとする．なお，レベルの初期値は1である．

②サイト S_s のレベル L_s の編集命令に対応する挿入位置は，N点入力位置モデルのレベル L_s の領域 s とする．N点入力位置モデルのルートフォーカスは，基準文字，もしくは L_s-1 以下のレベルの挿入文字からなる．

③サイト S_s のレベル L_s の編集命令で生成された低位フォーカスがレベル L_s の文字でないときは，その低位フォーカスがN点入力位置モデルの低位ルートフォーカスとなる．

④サイト S_s のレベル L_s の編集命令で生成された低位フォーカスがレベル L_s の文字であるとき，それは自己サイトの挿入文字であるが，その場合の挿入位置はレベル L_s の領域 s の低位フォーカスに隣接した高位側である．

⑤他のサイトからの編集命令がフォーカスとして指定した文字が，当該サイトに未到着である時は，その編集命令の処理をフォーカス文字の到着まで遅らせる．

⑥サイト S_s で文字を削除する場合は，エディタ画

面に表示されるバッファ文字の中からは削除するが，N点入力位置モデルの中ではラベルを付けるだけとする．この削除編集命令を受信した他のすべての共有者サイトは，確認返答を共有者にマルチキャストするので，削除の実行を終了したサイトのアドレスをこのラベルに記載しておく．

⑦共有者サイトから，削除編集命令を受信したならば，バッファ文字から削除し，モデルにラベル付けすると共に，すべての他の共有者サイトにその削除編集命令に対する確認返答をマルチキャストする．また，他の共有者サイトからマルチキャストされてきた確認返答を受信したら，その被削除文字のラベルに送信者のアドレスを記載する．

⑧ラベル付けられた既削除文字が，他サイトからの編集命令によってフォーカスとして参照されたとしても，通常の文字と同様に考えて挿入位置を決定する．

⑨現在存在する最低位レベルの領域の文字は，すべてのサイトからその最低位レベルの編集命令が全部到着し処理された時点で更新され，基準文字に組み込まれる．このとき削除のラベルが付けられており，かつルートフォーカスとして参照されているか，共有者サイトからの参照の可能性を残している文字は，ラベル付きのまま基準文字に組み入れる．これらの文字は，その時点で既に削除のラベルが付けられていた基準文字と共に，共有者サイトからの参照の可能性がなくなるまではラベル付きの状態で基準文字に残しておく．

⑩削除のラベルが付けられた文字に対する共有者サイトからの参照の可能性は，その被削除文字を削除した削除編集命令への確認返答が，すべての共有者サイトから発せられて，ラベルに記載された時点で完全になくなる．参照の可能性がなくなれば，その時点以降で最初に更新が行われる時に，N点入力位置モデルの構造からもそのラベル付き文字を抹消することができる．

5. おわりに

本論文では，複数のサイトから同時に編集が可能な共有エディタの基本的な設計を示した．本設計において並行な編集を実現するために用いた方法とは，編集操作が実行される際の順序の影響を受けない構造を共有テキストに与えることであり，別の言い方をすれば，共有テキストの各文字間を1要素として，それを全サイトに多重化して分配するということである．これによって，通信遅れ時間を伴う編集命令が，その到着の順序を逆転したとしても，同一の編集結果を得ることが可能となっている．しかしながら，文字間を単位とするということは，各文字が集合して単語あるいは文を構成するために持つべきより上位から見た構造を全く無視しているということである．したがって，もし編集操作が接近した領域で行われて競合すれば，各入力者が意図したものと全く異なった結果となり，単語や文の構造として明らかに無意味な文字列を生じる可能性があるわけである．この時は，さらにテキストの変更が必要になることは言うまでもないが，これらを含むすべての編集操作の最終結果が全サイトで少なくとも同一にはなっていることを，本エディタの設計によれば保証することができる．

本設計において，テキストの中でも最小の単位である文字を管理対象としたのは，1章に述べたように，任意の時点で任意の位置での並行編集を認めるのだとした前提に基づくが，この編集の並行性については以前から議論の対象となるところである．例えば，誰もが自由に編集できる状況を可能とするならば，作業者全員の注視点，ひいては議論の統一的な文脈を保つことに差し支えるという見方，あるいは，逆に自由な編集が制限されるならば，作業者が議論への参加のタイミングに余計な神経を使い，また積極的に参加して貢献しようとする心理的な動機付けに抑制的な作用を持つ，などの意見もある[1),2)]．しかし，並行な編集を必要とするか否かは，恐らく作業者の組織上の構成あるいは能力面での構成や，対象議題の性格にも大いに依存するであろう．したがって，設計の選択肢としては並行性を可能とすることも含めておいて，場合によって意図して並行性を抑制することを考えた方が良いと思われる．ただし，共同作業の進行に少なからぬ影響を持つと思われるそのような意図的な操作を加える場合には，何が共同作業にとって本質的であるかについての十分な研究がなされるべきであろう．そしてそれによって初めて，人間の共同作業の中に計算機による支援環境を含める際に必要となる人間同士の間の新しい取り決め (social protocol) が，より適切に形成され得るはずである．

参考文献

1) Stefik, M., Foster, G. et al.: Beyond the Chalkboard: Computer Support for Collaboration and Problem Solving in Meetings, *Comm. ACM*, Vol. 20, No. 1, pp. 32-47 (1987).
2) Sarin, S. K. and Greif, I.: Computer-Based

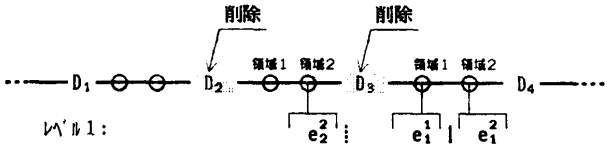
Real-Time Conferencing Systems, *IEEE Comput.*, Vol. 13, No. 10, pp. 33-45 (1985).
3) Crowley, T. and Forsdick, H.: MMConf: The Diamond Multimedia Conferencing System, *Proc. Groupware Technology Workshop*, IFIP W. G. 8.4 (1989).
4) 阪田，上田：構内型マルチメディア在席会議システムの実現とその評価，情報処理学会論文誌，Vol. 31, No. 2, pp. 249-258 (1990).
5) 鳩野，上田ほか：グループ協同作業支援のためのマルチメディア在席対話システム，情報処理学会論文誌，Vol. 30, No. 4, pp. 527-535 (1989).
6) Opper, S.: A Groupware Toolbox, *BYTE*, pp. 275-282 (1988).
7) Leland, M. D. P., Fish, R. S. et al.: Collaborative Document Production Using Quilt, *CSCW '88*, pp. 206-215 (1988).
8) Conklin, J. and Begeman, M.: gIBIS: A Hypertext Tool for Exploratory Policy Discussion, *CSCW '88*, pp. 140-152 (1988).
9) Trigg, R. H.: Guided Tours and Tabletops: Tools for Communicating in a Hypertext Environment, *CSCW '88*, pp. 216-226 (1988).
10) Mantei, M.: Capturing the Capture Lab Concepts: A Case Study in the Design of Computer Supported Meeting Environments, *CSCW '88*, pp. 257-270 (1988).
11) Garcia-Molina, H.: Message Ordering in a Multicast Environment, *The 9th ICDCS*, pp. 354-361 (1989).
12) Bernstein, P. A. and Goodman, N.: Concurrency Control in Distributed Database Systems, *ACM Comput. Surv.*, Vol. 13, No. 2, pp. 185-221 (1981).

付 録 A

2サイトで4文字ずつ入力した場合の編集過程の例

付表 1 4/4 次一括入力の編集過程の例
Annexed table 1 An example of editing process during 4/4 order packaged input.

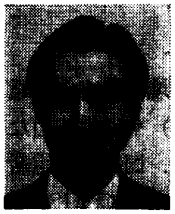
低位サイトの編集命令	低位サイトの編集結果	高位サイトの編集結果	高位サイトの編集命令
	$D_1 \mid D_2D_3 \mid D_4$ (初期状態)	$D_1 \mid D_2D_3 \mid D_4$ (初期状態)	
$\rightarrow(D_2D_3)$	$D_1D_2 \mid D_3 \mid D_4$	$D_1 \mid D_2D_3e_1^2 \mid D_4$	$(D_3D_4),Ins(e_1^2)$
$del(D_2),(D_1D_3)$	$D_1 \mid D_3 \mid D_4$	$D_1 \mid D_2D_3 \mid e_1^2D_4$	$\leftarrow(D_3e_1^2)$
$\rightarrow(D_3D_4)$	$D_1D_3 \mid\mid D_4$	$D_1 \mid D_2 \mid e_1^2D_4$	$del(D_3),(D_2e_1^2)$
$(D_3D_4),Ins(e_1^1)$	$D_1D_3e_1^1 \mid\mid D_4$	$D_1 \mid D_2e_2^2 \mid e_1^2D_4$	$(D_2e_1^2),Ins(e_2^2)$
到着の編集命令			到着の編集命令
$(D_3D_4),Ins(e_1^2)$	$D_1D_3e_1^1 \mid e_1^2 \mid D_4$	$D_1D_2 \mid e_2^2 \mid e_1^2D_4$	$\rightarrow(D_2D_3)$
$\leftarrow(D_3e_1^2)$	$D_1D_3e_1^1 \mid\mid e_1^2D_4$	$D_1 \mid e_2^2 \mid e_1^2D_4$	$del(D_2),(D_1D_3)$
$del(D_3),(D_2e_1^2)$	$D_1 \mid e_1^1 \mid e_1^2D_4$	$D_1e_2^2 \mid\mid e_1^2D_4$	$\rightarrow(D_3D_4)$
$(D_2e_1^2),Ins(e_2^2)$	$D_1e_2^2 \mid e_1^1 \mid e_1^2D_4$	$D_1e_2^2 \mid e_1^1 \mid e_1^2D_4$	$(D_3D_4),Ins(e_1^1)$



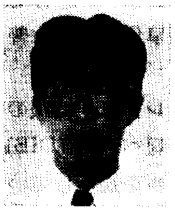
付図 1 4/4 次一括入力後の2点入力位置モデルの例（サイト1における表示）
Annexed fig. 1 An example of two input position model after 4/4 order packaged input. (Described at the site No. 1.)

を付表1に示す．左右端の欄がそれぞれ低位サイト，高位サイトに到着した編集命令で，中央部が編集経過である．時間は下向きに経過する．さらに最終的に到達した結果における2点入力位置モデルを付図1に示す．

(平成2年9月12日受付)
(平成3年9月12日採録)



池井 寧（正会員）
昭和63年東京大学大学院工学系研究科産業機械工学博士課程修了．工学博士．大阪大学工学部助手．分散システム，ヒューマン・インタフェース，マイクロマシンの研究に従事．日本機械学会，計測自動制御学会，日本ロボット学会各会員．



都築 功兒（正会員）
1966年生．1990年大阪大学工学部電子制御機械工学科卒業．現在同大学院工学研究科電子制御機械工学専攻修士課程に在学中．ヒューマン・インタフェースに関する研究に従事．

大川 善邦（正会員）
1934年生．1957年東京大学工学部機械工学科卒業．1962年東京大学大学院数物系研究科博士課程修了．工学博士．1985年より大阪大学工学部電子制御機械工学科教授．並列処理，分散制御システム，ロボットのリアルタイム制御などに興味をもっている．計測自動制御学会，AAAI, ACM, IEEE Computer Society などの会員．