コマンド操作の一貫性と区分情報†

守屋慎次†† 中谷吉久†††

本論文では，コマンド操作における一貫性の概念を区分情報という概念を用いて明らかにし，一貫性を検査する方法を示す．本論文では次に示す方法をとった．(1) 対話型システムにおける1つの操作には，4側面(概念的，意味的，構文的，語彙的)が存在し，さらに各側面は，入力側，入出力間，出力側という3つの段階に分類できることを示した．(2) 1つの操作の一貫性も，前述の4側面および3段階の12種類に分解できることを示し，次の3つの形式的な定義を仮説として示した．すなわち，上記12種類のそれぞれの一貫性，4側面のそれぞれの一貫性，そして全側面にわたる一貫性である．さらにその過程において，(3)操作の定義域と値域，および写像の集合のそれぞれを区分するという概念を示し，区分に用いられる情報，すなわち「区分情報」をユーザインタフェースの視点から4側面と3段階に分類した．そして主題区分情報という別格の区分情報が存在することを示した．本論文の意義は，①1つのコマンド操作を設計の段階に対応させて分解する手法と，4側面および3段階に分解された個々の一貫性から，順次，より広範囲の一貫性を検査する方法とその判定基準とを与えたこと，②区分情報という新しい有用な概念を示したことである．

1. はじめに

ユーザインタフェースに関する研究・開発が盛んに行われているが，しかし依然として，ユーザインタフェースの特性や構造に関する概念には不明確なものがある．例えば重要な特性のひとつである一貫性 (consistency) は，概念そのものがまだ不明確であり，したがって一貫性のある設計法や一貫性の評価法はまだ確立されていない．

本論文の目的は，(1)対話型システムにおける操作の一貫性とはどのようなものであるかを明らかにしながら分類し，(2)一貫性を検査するための方法を示すことである．本論文では，1つの操作には4側面×3段階の，従って 12 種類の一貫性が存在することを示し，それぞれの一貫性および全体の一貫性を検査するための方法を述べる．さらにその過程で，「区分情報」という，広く有用な概念を示す．

本論文で提案する分類法と一貫性の検査法には，次の(1)～(4)のような価値がある．

(1) 1操作の一貫性の定義を与えている．

(2) システムの設計者または利用者が，一貫性を分析・検査あるいは訂正するための方法と基準を与える．

(3) 対話型システムの一貫性を増す設計法を与える．

(4) 本論文で用いられる4側面，3段階，および区分情報の考え方は，一貫性だけでなく他の概念の整理に使える．

対話型システムにおける一貫性を分類し，その性質を求めるに際して，次の問題がある．すなわち，これまでに提案されている一貫性の分類と定義に混乱が見られる．このことを次の4つの観点から検証してみよう．

① インタラクションを性格づける3要素[6]，すなわち「操作」，「対象」，「時間」のうちのどの一貫性を論じているのか．

守屋と中谷[1),2)]，Reisner[4)]，Payne ら[3)]は操作に関してその一貫性を論じた．Kellogg[5)] は conceptual consistency と communication consistency という概念によって操作の一貫性に言及し，physical consistency という概念によって対象の一貫性に言及しているが，操作の一貫性と対象の一貫性が明確には区別されていない．

② 1操作における一貫性か，多操作間における一貫性か．

守屋と中谷は「取消」という1種類の操作について，その一貫性を論じた．Reisner も1操作の一貫性を論じた．Payne らは1操作の一貫性と多種類の操作間での一貫性を明確には区別していない．Kellogg は，彼が挙げた一貫性の個々について明確には定義していないので不明である．

③ 一貫性の考え方の拠り所．

研究者によって一貫性を考察する際の考え方の拠り

---

† Consistency and "Partition Information" in a Command Operation by SHINJI MORIYA (Department of Electrical Communications Engineering, Faculty of Engineering, Tokyo Denki University) and YOSHIHISA NAKAYA (Computer Section, Technological Administration Division, Industrial Research Institute of Kanagawa Prefecture).

†† 東京電機大学工学部電気通信工学科
††† 神奈川県工業試験所技術管理部電子計算科

所が様々である．一貫性の欠如は設計時に潜入すると考えてよい．したがって対話型システムを設計する段階[7]を，一貫性の考え方の拠り所とすべきであると考える．このことに着目したのは，本論文がはじめてである．

④ 入力側，入出力間，出力側のどの一貫性か．

守屋と中谷は入力側と出力側を区別せずに，またReisner, Payne らは入力側だけの一貫性を扱っている．Kellogg は明確な記述がなく不明である．

上に挙げた問題点（の一部）を解決するために，本論文では次に示す方法をとる．ある1種類の操作における一貫性を（①②），設計の段階に添って（③），入力側，入出力間，出力側のすべて（④）を区別しながら分類し，それぞれの一貫性を明らかにしていく．

本論文の構成は次のようになっている．2章では，対話型システムを設計する4段階[7]に対応して，操作に4側面があることを，区分情報の概念を説明し分類しながら示す．次いで，3章では1つの操作の一貫性を12種類に分解し，それぞれの一貫性および全体の一貫性を検査するための方法を示す．4章では他の研究と比較する．

2. 設計の4段階，操作の4側面，区分情報

本章では，対話型システムの動きには3つの段階(2.2節）と4つの側面（2.3節）があることを区分情報という概念を導入して示す．また，この4側面は対話型システムを設計する4段階に対応していることも示す．この4段階を2.1節に述べる．

2.1 対話型システムを設計する4段階

本節では，対話型システムを設計する際の4段階について説明する．各段階は Foley と van Dam[7] が提唱した4段階を基礎にして，筆者らの考えにもとづいて概念をより一般化し整理したものである．

（1） 概念設計 (conceptual design)

概念設計では，①人と自然と人工物とからなる環境中において，②設計しようとするシステムを同定し(すなわちシステムの目的や範囲を定め)，③そのシステムと人，そのシステムと自然，およびそのシステムと他の人工物との関係づけを行う．別の言い方をすれば，上記よりは狭い言い方であるが，誰がいつどこで何をどのように処理するシステムかを，人，時，場所，仕事，メディア，端末，通信網，ファイルなどの，単体としての最大単位の水準において設計する．(たとえば，携帯可能かつ無線通信が可能なペン入力式の電子手帳により，何時でも何処からでも地球上の誰にでも，筆跡を実時間式または蓄積式で双方向に，しかも相手と話をしながら送受信できるシステム)

（2） 意味設計 (semantic design)

意味設計では，①設計しようとしているシステム内において，②起こり得る状態（たとえば，ウィンドウやメニューやカーソルなどの表示と消去，画面のスクロールやメニューの選択など）を同定し，③それら状態間の関係づけを行う（たとえば，現在の状態からどのような入力によって何を出力し，どの状態に移るかを決める)．

（3） 構文設計 (syntactic design)

構文設計では，①各状態およびそれに対する入力と出力において，②論理的な元素を同定し（言いかえると，意味を失わずにそれ以上分解することができない単位，すなわちトークン[7]を定め)，③トークンを時間的および空間的に配置する順序を決める．入力におけるトークンの例として，MS-DOS コマンドにおけるオペレータ名，ドライブ名，ファイル名の「拡張子」などがある．

（4） 語彙設計 (lexical design)

語彙設計では，①物理的な装置（例．キーボード）や物理的な空間（例．画面）において，②物理的な元素，すなわちプリミティブ[7]（例．キー，色，字の幅，位置）を同定し（何を採用するかを決め)，③そのプリミティブを(3)のトークンと関係づける．すなわち，トークンをプリミティブの列または組によって表現する．入力の語彙設計例として，コマンドにおける「削除」というトークンへの文字Dの割当て，がある．

上記は文献 7) を基礎としているが，本論文では以下の3つの工夫をした．まず段階(1)(2)(3)(4)の全体を通じて共通な，①②③という考え方を新たに導入して整理した．ここで①は設計しようとしている対象の範囲を決めることを，②は①の範囲内における設計対象の構成要素を同定することを，③は②の構成要素どうしの関係を定めることを，それぞれ示している．①②③は設計段階(1)(2)(3)(4)の考え方をより鮮明にするために導入した．工夫の第2は段階(1)を新たに設けて一般化したこと，第3は文献 7) における conceptual design と semantic design を段階(2)へ統合して整理したことである．

2.2 操作，および区分情報と主題区分

本節では，操作，区分情報，主題区分という3つの概念について述べる．

(1) 操作の3段階

操作とは次の3つを指す.

- 対話型システムを駆動するために入力される情報,
- 入力途上および入力終了後に生ずる内部状態の遷移,
- 変化を利用者に知らせるために出力される情報.

対話型システムの動きのうち,上記の3つはそれぞれ,入力の段階,入出力間の変換の段階,出力の段階に対応している.さらに,集合論の用語を用いると,入力の段階は定義域上での操作に,入出力間の変換の段階は定義域と値域との間における操作に,出力の段階は値域上での操作に,それぞれ対応している.以上の考え方が本論文における考え方の主軸のひとつであり,もうひとつの主軸が前節で述べた設計の4段階である.

上の定義からわかるように本論文の以後における操作という用語には,利用者の動きや思考は含めていない.しかし,対話型システムが行う操作に,そのシステムの利用者は従わざるを得ないので,操作の一貫性が成立していれば利用者に対してよりよい影響を与えると考えてよい.なお,ある一操作に一貫性があるか否かは,それを検査しようとしている人(利用者や設計者など)の要求の高さ,すなわちその人の考え方に依存していることを3.3節で述べる.

(2) 対応していることを示す方法

設計の4段階が操作の4側面(2.3節)に対応していることを示すのが第2章の目的のひとつであるが,「対応していること」はすでに文献6)で述べられている.しかし,文献6)では対応の実例も,対応の証明もしていない.2.1節で概説した設計の4段階(1)(2)(3)(4)と,2.3節で述べる4側面(1)(2)(3)(4)の両者の形式化は困難であり,したがって完全な証明は少なくとも現時点では望めない.

そこで本論文では,具体例を示すことによってこの対応関係が存在することを明らかにする.取り上げる具体例は「一太郎 Ver. 3」における,「字の大きさを変更する指令」である.以後,指令という言葉は,一太郎 Ver. 3というシステムを駆動するために入力される情報を指す.

(3) 区分情報

一太郎 Ver. 3では,たとえば文書ファイル中のある文字列は全角,隣の文字列は半角という具合いに,文書ファイルそのものが字の大きさによって区分されていることがある.また,たとえば文書ファイル中のある文字列は英字で,隣の文字列は漢字という具合いに,字の種類によって区分されていることがある.このように,操作の対象となる定義域または値域を区分する情報を区分情報と呼ぶことにする.例示した全角や半角は「字の大きさ区分」をつくり出す区分情報であり,アルファベットや漢字は「字種区分」をつくりだす区分情報である.たとえば一太郎 Ver. 3の字の大きさ区分情報には次がある.すなわち,全角,半角,上付1/4角,下付1/4角,(横)倍角,縦倍角,4倍角があり,このほかに,半角,上付1/4角,下付1/4角のそれぞれの縦倍の大きさがある.したがって「字の大きさ区分情報」の種類は10である.

区分情報には,定義域だけを区分するもの,値域だけを区分するもの,両者を区分するものの3種がある.それぞれを定義域区分情報,値域区分情報,写像区分情報と呼ぶことにする.これらをそれぞれ,定義域区分,値域区分,写像区分と略記することにする.字の大きさ区分情報は定義域区分かつ値域区分である.その理由は,ある大きさの字(定義域上)が,他の大きさの字(値域上)に変更されるからである.(写像区分の例は2.3節の(1)(2)で示す.)

(4) 主題区分情報

字の大きさを変更する指令ひとつをとってみても,様々な区分情報が関係してくることを2.3節の(1)(2)(3)(4)で示すが,それらのうち字の大きさ区分情報を主題区分情報と呼ぶことにする.これは,以下における議論が,字の大きさを変更する操作に一貫性があるか否かに関するものであり,字の大きさが議論の主題となっているからである.他の指令に対する主題区分情報を例示すると,一太郎 Ver. 3において文章中に「文字飾り」をつける指令の場合,下線や網掛けや文字色などの,文字飾りの種類が主題区分情報となる.また,文章の部分を移動する指令の主題区分情報は,字,行,「ブロック」などの,移動の単位である.

2.3 操作の4側面,および区分情報

前節では,対話型システムが行う操作が3つの段階からなることを述べた.本節の目的の第1は,操作の3段階がさらに4つの「側面」に分類でき,しかもその4側面が設計の4段階に対応していることを示すことである.目的の第2は,区分情報も4側面と3段階に分類できることを示すことである.2つの目的はいずれも4側面と3段階に関する話題であって密接に関係しているため,本節では2つの目的を同時に並行し

て述べていく．そこで，第2の目的だけに関する説明は[ ]でくくって区別することにする．

（1） 字の大きさを変更する指令の概念的側面

標題は，字の大きさを変更する指令を実行する際にこの対話型システムが行う操作の概念的側面，という意味を表す．

① この指令によって操作される対象

操作の対象は，文書ファイル中の文字列である．したがってこの操作の定義域と値域はいずれも文書ファイルそのものである．

●文書ファイルの種類［定義域区分かつ値域区分］

一太郎 Ver. 3 が扱うことのできる文書ファイルには外部文書ファイルと内部文書ファイルとがある．外部文書ファイルはフロッピーディスクなどの外部記憶装置上に保存されるファイルであり，内部文書ファイルはキーボードから入力されたり，外部ファイルから読み込まれる主記憶上のファイル［すなわち内部データ構造］である．外部文書ファイルには，一太郎 Ver. 3 の用語による「通常」，「リンク形式1」，「リンク形式2」の3種類があり，内部文書ファイルには1種類がある．［したがって合計4種類のファイルがあることになり，そのそれぞれが，一太郎 Ver. 3 における文書ファイル群の世界を区分する要素，すなわち区分情報となる．］

字の大きさを変更する指令の定義域および値域となるファイルは内部文書ファイルだけである．［したがって，この指令の一貫性を論ずる際，ファイルの種類は内部文書ファイルだけとして扱ってゆくのが妥当である．］内部文書ファイルは文字列からなるファイル，すなわちテキストファイルである．

② この指令によって行われる操作

この指令は，内部文書ファイル中の文字列を他の大きさに変更する．

●字の大きさを変更する方法の種類［写像区分］

この指令はすべての文書ファイル（「すべて」とはいっても一太郎 Ver. 3 では内部文書ファイルのみ）上で定義されている．［すなわちこの指令が，文書ファイル群を区分することはない（ファイルの集まりは内部文書ファイルのみであるから）．］したがって，文書ファイルの水準（すなわち概念的側面）においては，字の大きさを変更［すなわち写像］する方法は1種類である．

［表1の概念的側面の行を見られたい．Dcon, Mcon, Rcon を区分集合と呼ぶことにする．例えば，集合 Dcon は「内部文書ファイル」という区分情報からなり，集合 Mcon は写像 fcon という区分情報からなる．］

（2） 字の大きさを変更する指令の意味的側面

この操作の意味的側面の一部を利用者の視点から以下に述べる．

① 字の大きさと変更単位

画面に表示される内部文書ファイルの姿は，字を横一列に並べたものが行，行を縦一列に並べたものが内部文書ファイルである．

●字の大きさ［定義域区分かつ値域区分，表1の Dsem 1 と Rsem 1］

この区分については主題区分情報としてすでに 2.2 節(3)(4)で述べたが，本来この区分は意味的側面に属することだけを示す目的で，再度ここでとりあげた．

●変更単位［定義域区分，表1の Dsem 2］

字の大きさを変更する指令は，字単位または行単位で大きさを変更可能である．［この変更単位は変更の範囲を指定する際だけに用いられる情報であるので，これは定義域区分．］

どの大きさの字を，字と行のいずれの単位で変更できるかを，**表2**に示した．表2に載せた字の大きさは7種類であるが，これは 2.2 節(3)に述べた 10 種のうちからよく使うものだけを例示のために選出したものである．選出の理由は，以後の議論を簡素化するためである．

② 大きさの表示法，および字種

●大きさの表示法［値域区分，表1の Rsem 2］

字の大きさを表示する方法として次の2つがある（**表 3**）．大きさを示す印と字そのものとの組で表示する「高速モード」，字そのものを指定の大きさで表示する「精細モード」である．一太郎 Ver. 3 では①に示した7種類の大きさの字を，2つの表示法のいずれによっても表示できる（表3）．

●字種［定義域区分，表1の Dsem 3］

字種には，平・片仮名，漢字，数字，英字，句読点，特殊記号などがある．［大きさの変更が，変更したい文字列の字種に依存することがあるのでこれは定義域区分．］

●字種・字の大きさ変換の種類［写像区分，表1の Msem 1］

どの字種をどの大きさに変更できるかを**表4**に示した．表4によると，変換［すなわち写像］のされ方に

表 1 区分情報は4側面および3段階に分類できる．(注．表中の「高速」「精細」，「E」「F」は一太郎 Ver. 3 の用語である)
Table 1 "Partition information" can be classified into the four aspects and the three stages.

3段階 / 4側面	定義域区分 (D)	写像区分 (M)	値域区分 (R)
概念的側面 (con)	●文書ファイルの種類区分 Dcon＝{内部文書ファイル}	●字の大きさを変更する方法の種類区分 Mcon＝{fcon} ここで fcon : Dcon→Rcon	●文書ファイルの種類区分 Rcon＝{内部文書ファイル}
意味的側面 (sem)	●字の大きさ区分 Dsem 1＝{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，縦倍角，4 倍角} ●変更単位区分 Dsem 2＝{字単位，行単位} ●字種区分 Dsem 3＝{平仮名，片仮名，漢字，英字，数字，句読点}	●字種・字の大きさ変換の種類区分 Msem 1＝{fsem 11, fsem 12} ここで fsem 11 : {平仮名，漢字}→{全角，(横)倍角，縦倍角，4 倍角} fsem 12 : {片仮名，英字，数字，句読点} →{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，縦倍角，4 倍角} ●字の大きさ・字の大きさ変換の種類区分 Msem 2＝{fsem 21, fsem 22, fsem 23, fsem 24, fsem 25} ここで fsem 21 : {全角} →{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，縦倍角} fsem 22 : {半角，上付 1/4 角，下付 1/4 角} →{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角} fsem 23 : {(横)倍角} →{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，4 倍角} fsem 24 : {縦倍角}→{全角，縦倍角，4 倍角} fsem 25 : {4 倍角}→{(横)倍角，縦倍角，4 倍角}	●字の大きさ区分 Rsem 1＝{全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，縦倍角，4 倍角} ●大きさの表示法区分 Rsem 2＝{高速モード，精細モード}
構文的側面 (syn)	●指令の種類区分 Dsyn＝{E 指令，F 指令}	●指令の種類・出力トークン列変換の種類区分 Msyn＝{fsyn} ここで fsyn : Dsyn→Rsyn	●出力トークン列区分 Rsyn＝{〈字とその大きさ〉，〈行幅〉，〈字とその大きさ〉}
語彙的側面 (lex)	●オペレータ語彙区分 Dlex＝{文字飾り，書式設定}	●オペレータ語彙・出力語彙変換の種類区分 Mlex＝{flex} ここで flex : Dlex→Rlex	●出力語彙区分 Rlex＝{P1, P2, P3, P4}

表 2　○印行の大きさの字は，その列の単位で変更できる．(この表は一太郎 Ver. 3 のもの．Ver. 4 ではすべての大きさの字を，任意の単位で変更可．)

Table 2　The symbol "○" in this table denotes the units (character unit or line unit) in terms of which text can be resized. The operations shown in this table correspond to those found in Ichitaro Version 3. 0. In the case of Version 4. 0, the text can be resized in terms of any unit, i. e. either in character units or line units).

大きさ ＼ 単位	字	行
全角	○	
半角	○	
(横) 倍角	○	
上付 1/4 角	○	
下付 1/4 角	○	
縦倍角		○
4 倍角		○

表 3　字の大きさの種類と 2 つの表示モードにおける表示例
＞ 印は行の左端に表示される．

Table 3　This table shows the different sizes of characters and how they look on the display screen.

大きさ 表示ﾓｰﾄﾞ ＼ 字の大きさ	高速	精細
全角	a	a
半角	a	a
(横) 倍角	a	a
上付1/4角	a	a
下付1/4角	a	a
縦倍角	＞ a	＞ a
4倍角	＞ a	＞ a

は次の 2 種類がある．まず，平仮名と漢字はそれぞれ，全角，(横)倍角，縦倍角，4 倍角に変換可能である [表 1 の写像 fsem 11]．また，片仮名，英字，数字，句読点はいずれも 7 種類のすべてに変換可能である [表 1 の fsem 12]．

● 字の大きさ・字の大きさ変換の種類 [写像区分，表 1 の Msem 2]

どの大きさの字がどの大きさに変換できるかを表 5 に示した．表 5 によると大きさの変換のされ方には次の 5 種類がある [次のアイウエオがそれぞれ表 1 の写像 fsem 21～fsem 25 に対応]．

ア．全角は，4 倍角を除くすべてに変換できる．

イ．半角と上付 1/4 角と下付 1/4 角は，縦倍角と 4 倍角を除くすべてに変換できる．

表 4　どの字種をどの大きさに変更できるかを示す．○印は変更可を，—印はもとの大きさのまま不変なことを意味する．

Table 4　This table shows the different sizes to which different characters can be resized. The symbol "—" denotes that, even in the resizing operation is performed, the sizes of the characters do not change.

from ＼ to	全角	半角	(横) 倍角	上付 1/4 角	下付 1/4 角	縦倍角	4 倍角
平仮名	○	—	○	—	—	○	○
片仮名	○	○	○	○	○	○	○
漢字	○	—	○	—	—	○	○
英字	○	○	○	○	○	○	○
数字	○	○	○	○	○	○	○
句読点	○	○	○	○	○	○	○

表 5　1 操作によって変更できる字の大きさを示す．E または F が記入された行の大きさの字は，その列の大きさに変更できる．・印は変更できない．—印は「E と F 指令のいずれでも変更可能」を意味する．E は「文字飾り」指令を，F は「書式設定」指令を意味する．

Table 5　This table shows which sizes of text can be resized to which sizes, using the single resizing operation. The symbols "E" and "F" denote the sizes to which text (having the size corresponding to the left of the table) can be resized. The symbol "・" shows that resizing is not possible. The symbol "—" stands for "either E or F." The symbol "E" represents the operation which is used when text is to be underlined, shaded, enlarged, etc. The symbol "F" represents the operation used while formatting text.

from ＼ to	全角	半角	(横) 倍角	上付 1/4 角	下付 1/4 角	縦倍角	4 倍角
全角	—	E	E	E	E	F	・
半角	E	—	E	E	E	・	・
(横) 倍角	E	E	—	E	E	・	F
上付 1/4 角	E	E	E	—	E	・	・
下付 1/4 角	E	E	E	E	—	・	・
縦倍角	F	・	・	・	・	—	E
4 倍角	・	・	F	・	・	E	—

ウ．(横)倍角は，縦倍角を除くすべてに変換できる．
エ．縦倍角は，全角と縦倍角と4倍角にのみ変換できる．
オ．4倍角は，(横)倍角と縦倍角と4倍角にのみ変換できる．

[意味的側面における写像区分の例として，上記において Msem 1 と Msem 2 を説明したが，この側面における写像区分はこれですべてではない．定義域区分における任意の区分集合から，値域区分における任意の区分集合への写像が，この側面における写像区分になり得る．すなわち，写像区分集合の種類は，定義域区分集合と値域区分集合を定めることによって自動的に決定される．]

(3) 字の大きさを変更する指令の構文的側面

この指令の構文的側面の一部を利用者の視点から以下に述べる．

① 指令の種類 [定義域区分，表1の Dsyn]

字の大きさを変更することができる指令に2種類ある．それらを表5に記号E，Fで示した．これらをE指令，F指令と呼ぶことにする．

② 出力トークン列 [値域区分，表1の Rsyn]

字の大きさを画面に表示する際の表示形式をあらわすトークンとして，以下で説明する〈行幅〉と〈字とその大きさ〉の二種類がある (表 6)．

〈行幅〉

一太郎では，縦倍角と4倍角文字を表示する際 (表3)，その字の高さは全角の高さ (すなわち縦倍の 1/2) で表示する．そして，その行が縦倍であることを示す符号 ">" が，その行の左端に表示される．この左端符号を用いて高さを示すトークンを〈行幅〉と表すことにする．

〈字とその大きさ〉

全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角のそれぞれの字を画面に表示する際 (表3)，一太郎 Ver. 3 では，字そのもので大きさを表示するか，字そのものとその字の大きさを示す印との組で表示する．この方法によって字の大きさを示すトークンを〈字とその大きさ〉と表すことにする．

上記の2つのトークンを用いると，それぞれの大きさの文字は次のようなトークン列で表現することができる．すなわち，縦倍角と4倍角の字はふたつのトークンの列

〈行幅〉〈字とその大きさ〉

によって表し，全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角の字は

〈字とその大きさ〉

と表すことができる．

③ 指令の種類・出力トークン列変換の種類 [写像区分，表1の Msyn]

一太郎 Ver. 3 では，E指令とF指令のいずれによっても，〈字とその大きさ〉および〈行幅〉〈字とその大きさ〉のいずれの形にも表示できる [表1の写像 fsyn]．

(4) 字の大きさを変更する指令の語彙的側面

この指令の語彙的側面の一部を利用者の視点から述べる．

① オペレータ語彙 [定義域区分，表1の Dlex]

字の大きさを変更するという目的に対して，E指令を実行するにはメニュー上で「文字飾り」という語彙を選択するが，F指令を実行するには同じメニュー上で「書式設定」という語彙を選択する．

② 出力語彙 [値域区分，表1の Rlex]

字の大きさを表示するトークンの，それぞれのプリミティブを表6に示した．〈行幅〉には1種の，〈字とその大きさ〉には3種のプリミティブがある．[表6の右端欄の P_1〜P_4 は本論文における呼び名である．]

③ オペレータ語彙・出力語彙変換の種類 [写像区分，表1の Mlex]

一太郎 Ver. 3 では，「文字飾り」と「書式設定」のいずれをメニューから選んでも，表6に示すプリミティブ P_1〜P_4 のいずれの形でも表示できる [表1の写像 flex]．

以上(1)(2)(3)(4)で述べたように「字の大きさを変更する指令」には，概念的，意味的，構文的，語

表 6 字の大きさを示すトークンとプリミティブ
Table 6 This table shows the tokens and primitives that represent the different sizes of characters.

字の大きさを表示するトークン	プリミティブ	説 明	本論文中におけるプリミティブの呼び名
<行幅>	>	>の行の幅が倍である	P_1
<字とその大きさ>	a a a a	1/4角 半角 全角，縦倍角 (横) 倍角，4倍角	P_2
	a	1/4角	P_3
	a ■	(横) 倍角 4倍角	P_4

集的の4側面があり，各側面には定義域，写像，値域という3段階がある．本論文は「字の大きさを変更する指令」という操作の一例をもってして，一太郎 Ver. 3 の全操作にもこれらの4側面と3段階があるとは主張していない．しかし，操作にこれらの4側面と3段階があることはこの一例から明らかである．

次に，これらの4側面と，2.1 節の設計の4段階とを比較すると，それぞれがうまく対応していることが分かる．ここでは「うまく対応していること」を証明しているわけではなく，「うまく対応していること」を読者に納得してもらうことを目的としている．2.2 節(2)でも述べたように，設計の4段階と操作の4側面のいずれもが現時点では形式化は困難であるため，また，形式化を試みるよりは考え方を体系化しながら例示するほうが圧倒的にわかりやすくなるため，このような方法をとった．

表1において示してきたように，区分情報は4つの側面と3つの段階に，すなわち 12 種に分類することができる．また，主題区分情報という「別格」の区分情報が存在することも2.2節で示した．

3. 操作の一貫性

本章では，まず，表1で示した 12 種類の区分情報のそれぞれに対応して 12 種類の一貫性があることを示し，次に，それぞれの一貫性を検査する方法を述べる．

一貫性という用語は定義せずに用いてきているが，本章において明確な定義を与える．定義を与えるまでに用いていく一貫性という用語の意味は，市販の辞書にみられるものとする．

3.1 12種類の一貫性とそれらの定義

本節では，ある1つの操作をとりあげ，その操作に一貫性が成立しているとはどのようなことかを調べ，一貫性という概念を定義する．

(0) 準　　備

表1では，区分情報を集合の形で，すなわち区分集合として表現している．この理由は，一貫性の議論をする際に，集合論における直積の概念を用いると便利なためである．

2つの有限集合を $A=\{a_1,\cdots,a_n\}$，$B=\{b_1,\cdots,b_m\}$ とし，それらの直積を $A\times B=\{(a_1,b_1),(a_1,b_2),\cdots,(a_n,b_m)\}$ と表す．(a_i,b_j) を直積 $A\times B$ の要素と呼ぶ．

また，主題区分集合 S を次のように定める．

$S=$ {全角，半角，(横)倍角，上付 1/4 角，下付 1/4 角，縦倍角，4倍角}

3.1 節の冒頭で述べた「ある1つの操作」とは，2章でとりあげた「文字の大きさを変更する指令」である．以後において「文字の大きさを変更する指令の○○的側面の一貫性」と略記するが，これは正確には「字の大きさを変更する指令を実行する際に対話型システムが行う操作の○○的側面の一貫性」を意味する．2.3 節の各側面(1)(2)(3)(4)をそれぞれ参照しながら次の(1)(2)(3)(4)の順に述べる．

(1) 字の大きさを変更する指令の概念的側面の一貫性

標題(1)の一貫性は次の3段階，すなわち入力側，出力側，入出力間のそれぞれの一貫性に分解できる．この3段階の一貫性はそれぞれ，定義域，値域，写像の各区分情報に基礎をおいた一貫性である．

● 入力側の一貫性

この操作の概念的側面の入力側は一貫性がある．その理由は次のとおり．表1に示したように，定義域区分には，文書ファイルの種類区分集合 Dcon があるが，Dcon のどの要素に対しても，この指令が定義されているからである．

● 出力側の一貫性

この指令の概念的側面の出力側は一貫性がある．その理由は次のとおり．表1に示したように，値域区分には，文書ファイルの種類区分集合 Rcon があるが，Rcon の全要素に対してこの指令が定義されているからである．

● 入出力間の一貫性

この指令の概念的側面の入出力間は一貫性がある．その理由は次のとおり．表1に示したように，写像区分には，字の大きさを変更する方法の種類区分集合 Mcon があるが，Mcon の要素が唯一であるから，すなわち Dcon の全要素から Rcon の全要素に対して唯一の写像 fcon が定義されているからである．

(2) 字の大きさを変更する指令の意味的側面の一貫性

概念的側面の場合と同様に入力側，出力側，入出力間のそれぞれの一貫性に分解することができる．

● 入力側の一貫性

この指令の意味的側面の入力側は一貫性が欠如している．その理由は，表2の縦欄（字の大きさ区分集合 Dsem 1）と横欄（変更単位区分集合 Dsem 2）の間の対応が完全ではなく，欠けている（○印がない）部分があるからである．すなわち，直積 Dsem 1×Dsem 2

の要素のうち，対応が欠けているものがあるからである．

表1上の意味的側面の定義域区分には字種区分集合Dsem3もある．したがって，Dsem1×Dsem2×Dsem3の全要素に対してこの指令が定義されていなければ一貫性は成立しない．逆に，全要素に対してこの指令が定義されていれば一貫性が成立する．

したがって，これまで未定義のまま用いてきた一貫性という概念を，それと同値で，しかも，より明確な概念によって次のように定めることができる．

C1. この指令が，この側面の定義域区分におけるすべての区分集合の，直積の全要素に対して定義されているとき，この側面における入力側の一貫性が成立するとよぶ． ■

● 出力側の一貫性

この指令の意味的側面の出力側は一貫性がある．その理由は，2.3節(2)の②「大きさの表示法，および字種」で述べたように，7種類の大きさの字を，2つの表示法のいずれによっても表示できるからである．すなわち，値域区分における2つの区分集合の直積・Rsem1×Rsem2の全要素に対して，この指令が定義されているからである．

したがって，出力側の一貫性の定義は，入力側のC1と同様に考えることができる．

C2. この指令が，この側面の値域区分におけるすべての区分集合の，直積の全要素に対して定義されているとき，この側面における出力側の一貫性が成立するとよぶ． ■

C1とC2の違いは，「定義域」と「値域」，そして「入力側」と「出力側」という言葉だけである．

● 入出力間の一貫性

この指令の意味的側面の入出力間は一貫性が欠けている．理由は2つあり，その第1は表4にある．すなわち，字種によっては，変更できたりできなかったりする大きさが存在する．このことは，表1で意味的側面の写像区分に位置するMsem1の要素が，空または唯一でないからであるといえる．(要素が空の場合，すなわち写像が存在しない場合，字は，どの大きさへも変更できないことを意味する．すなわち，どの大きさへも変更できないという意味において一貫している．)

一貫性欠如の第2の理由は表5にある．例えば，全角文字を縦倍角に変更できるが，半角文字を縦倍角にはできない．したがって一貫性がない．このことは，表1でMsem2の要素が，空または唯一でないからであるといえる．

以上の2つの理由をまとめれば，Msem1×Msem2の要素が空または唯一でないからであるといえる．したがって，入出力間において一貫性が成立することを，次のように定義できる．

C3. この側面の写像区分におけるすべての区分集合の，直積の要素が空または唯一であるとき，この指令の入出力間の一貫性が成立するとよぶ． ■

(3) 字の大きさを変更する指令の構文的側面の一貫性

構文的側面の一貫性も，入力側，出力側，入出力間のそれぞれの一貫性に分解することができる．2.3節(3)を参照しながら以下に例示してみよう．

● 入力側の一貫性

この指令の構文的側面における入力側は一貫性がある．理由を以下に述べる．

字の大きさを変更する指令には，表5に示すE，Fの2種類がある．表5をみると，各行にEおよびF指令が存在する（一印はEとF指令のいずれでも変更できることを意味する）．すなわち，定義域上の，7種類の大きさの字のいずれに対しても，EとFのいずれの指令をも適用できることができる．したがって，一貫性がある．

区分集合を用いて上のことを説明してみよう．まず主題区分（すなわち字の大きさ区分）集合Sと，指令の種類区分集合Dsynとの直積を考える．このとき，この一貫性が成立することを，C1とまったく同じ文言で表現できる．(注1. すなわち，この側面の定義域区分におけるすべての区分集合と主題区分集合とからなる集合 $X=\{\text{Dsyn}, S\}$ を考え，X の全要素の直積 Dsyn×S を，C1における「直積」とみなす.)

● 出力側の一貫性

この指令の構文的側面における出力側の一貫性は成立しない．理由は以下のとおり．

2.3節(3)②で示したように，また，表3と表6に例示するように，〈行幅〉が用いられるのは縦倍角と4倍角を表示する場合だけである．したがって一貫性がない．

区分集合を用いてこのことを説明してみよう．出力トークン列区分集合Rsynと主題区分集合Sとの直積を考える．このとき，この一貫性が成立することを，C2とまったく同じ文言で表現できる．(注2. すなわち，この側面の値域区分におけるすべての区分集合と

主題区分集合とからなる集合 $X=\{Rsyn, S\}$ を考え，X の全要素の直積 Rsyn×S を，条件 C2 における「直積」とみなす.)

● 入出力間の一貫性

この指令の構文的側面における入出力間は一貫性がある．理由は，2.3 節(3)の③で述べたように，E指令とF指令のいずれによっても，〈字とその大きさ〉および〈行幅〉〈字とその大きさ〉のいずれの形にも表示できるからである．すなわち，表1の構文的側面の行が示すように，Dsyn の全要素から Rsyn の全要素に対して唯一の写像 fsyn が定義されているからである．区分集合を用いた議論は，C3 と全く同じ文言で表現できる．

(4) 字の大きさを変更する指令の語彙的側面の一貫性

語彙的側面の一貫性も，入力側，出力側，入出力間のそれぞれの一貫性に分解することができる．2.3 節(4)を参照しながら説明する．

● 入力側の一貫性

一貫性が欠如している．理由は次のとおりである．E指令では，メニュー上の「文字飾り」を選択するが，F指令では同一メニュー上の「書式設定」を選択する．したがって一貫性がない．この場合はオペレータ語彙区分集合 Dlex と主題区分集合Sとの直積をとる．定義は C1 と同じである．(上記(3)における注1と同様に考える.)

● 出力側の一貫性

一貫性が欠如している．次のいずれもが一貫性欠如の理由になる．表6中の P_1 は縦倍角と4倍角だけに使っている．P_3 は 1/4 角だけに使っている，など．この理由を区分集合を用いて説明するには出力語彙区分集合 Rlex と主題区分集合 S との直積をとる．定義は C2 と同じである．(上記(3)における注2と同様に考える.)

● 入出力間の一貫性

一貫性が成立している．理由は，2.3 節(4)の③で述べたように，「文字飾り」と「書式設定」のいずれをメニューから選んでも，表6に示すプリミティブ P_1〜P_4 のいずれの形でも表示できるからである．すなわち，表1において Dlex の全要素から Rlex の全要素に対して唯一の写像 flex が定義されているからである．一貫性の定義は C3 と同じである．

3.2 一貫性の検査法

前節において，字の大きさを変更する指令の4側面のそれぞれにおいて，入力側，入出力間，出力側という3段階のそれぞれにおける一貫性，すなわち 12 種類の一貫性を議論した．またそれぞれにおいて一貫性の定義を述べた．それらの定義は各側面の各段階に分散しているが，本節ではそれらを統合する．統合した結果は一貫性を検査する方法を示している．

これまでの議論は一太郎 Ver. 3 における字の大きさを変更する指令を例題として述べている．したがって一太郎 Ver. 3 の他の指令や，他システムにおける「字の大きさを変更する指令」に対しても成立することは保証していない．そこで，定義そのものを仮説として述べ，仮説として示す定義が，一太郎 Ver. 3 の他の指令に対しても正しいか否かの検証，および，一太郎 Ver. 3 以外の対話型システムに対しても正しいか否かの検証は，今後の課題としたい．

前節で述べた C1 は，意味的と構文的と語彙的のそれぞれにおける入力側の一貫性が成立する定義になっていることを前節で述べた．残りのもう1つの側面，すなわち概念的の場合であるが，3.1 節(1)「入力側の一貫性」に立ち戻ってもう一度考えてみると，他の3側面と同様に，概念的の場合にも C1 が成立することがわかる（ただし，3.1 節(3)「入力側の一貫性」の注1と同様に考える)．したがって，入力側の一貫性が成立することの定義は，4側面にわたって C1 ということになる．

同様にして，出力側の一貫性が成立することの定義は，4側面にわたって C2 である（ただし，3.1 節(3)「出力側の一貫性」の注2と同様に考える)．

さらに，C1 と C2 の違いは，「定義域」と「値域」，および「入力側」と「出力側」という言葉だけであった．そこで C1 と C2 を統合した次の定義を仮説とすることにする．

[仮説としての定義1] 概念的，意味的，構文的，語彙的の各側面における入力側（出力側）の一貫性

ある1操作の，概念的，意味的，構文的，語彙的のいずれか1つの側面を考える．さらに，その側面の定義域（値域）区分におけるすべての区分集合と主題区分集合とからなる集合 X を考え，X の全要素の直積をとる．この直積の全要素に対してその操作が定義されているとき，その操作のその側面における入力側（出力側）の一貫性が成立するという． ■

入出力間の一貫性の定義として C3 を，4つの側面で用いることができる(概念的側面の場合も)．そこで次の定義を仮説とする．

[仮説としての定義2] 概念的，意味的，構文的，語彙的の各側面における入出力間の一貫性

ある1操作の，概念的，意味的，構文的，語彙的のいずれか1つの側面を考える．その側面の写像区分におけるすべての区分集合の，直積の要素が空または唯一であるとき，その操作のその側面における入出力間の一貫性が成立するという． ■

仮説としての定義1と2は，各側面における各段階の一貫性を定義したものである．次に述べる仮説は，ある1つの側面において，3段階を統合した一貫性の定義である．

[仮説としての定義3] 1側面の一貫性

ある1操作の，任意の1側面における3段階のそれぞれの一貫性がすべて成立するとき，その1操作のその側面における一貫性が成立するという． ■

次に述べる仮説は，4側面を統合した一貫性の定義である．

[仮説としての定義4] 1操作の全側面にわたる一貫性

ある1操作の，4側面のそれぞれの一貫性がすべて成立するとき，その操作の全側面にわたる一貫性が成立するという． ■

仮説としての定義 1，2，3，4 は，ある1操作の一貫性を調べる際に，トップダウンに4側面と3段階に分解する手法と，ボトムアップに判定する基準とを与えている．

3.3 一貫性と区分集合群との関係

区分集合，そして区分集合間の関係（例えば表2～6）のそれぞれは，一貫性の議論における要となっている．

しかし，3.1 と 3.2 節では，ある側面における定義域（または値域）の区分集合群（たとえば仮説としての定義1における集各 X）をどのようにして決定するかについては明らかにしていない．また，1つの区分集合を定めることになる要素群の範囲を，どのようにして決定するかについても明らかにしていない．本節では，この要素群・集合群と一貫性との間の関係について述べる．

一貫性を調べるには各区分集合と区分集合群を定め，区分集合間の関係を定める必要がある．定める人が設計者で，定めるのが設計時の場合，要素群・集合群と関係の両者が設計者によって決められる．定める人が利用者の場合，（すべてではないが多くの対話型システムでは）関係はすでに定められているので，定める余地があるのは要素群と集合群だけである．

要素群を小さめに定めれば定めるほど，また，集合群を小さめに定めれば定めるほど一貫性は成立しやすい．大きめに定めれば成立し難くなる．どのように定めるかは，定める人の要求の高さに依存する．要求の例として次の区分集合を考えてみよう．

対話型システム区分集合
= {MS-DOS, 一太郎, WordStar}

このように定めると，MS-DOS と一太郎と WordStar という3つのシステムの間で，ある操作の一貫性が成立することを要求することになる．（なお，対話型システム区分集合は概念的側面に属する．）

一貫性ある対話型システムを設計しようとする際，多種多様な区分情報をどのような考え方にもとづいて一貫性と関係づけたらよいであろうか．あるいはどの区分情報が使いやすさにとってより重要であろうか．本論文ではこれらの点については明らかにしていない．これらは今後の検討課題としたい．

4. 検　　討

第1章では，対話型システムの一貫性に関するこれまでの研究に4つの問題点があることを指摘し，本論文でのアプローチを示した．本章では次の3つの観点から，本論文で述べた一貫性の議論を他の研究と比較しつつ検討する．

（1） インタラクション空間の3軸における一貫性

「インタラクションの空間」は「対象」，操作，「時間」の3軸から成るという[6]．3軸のうち操作は，概念的，意味的，構文的，語彙的の4側面を持つという[6]．

したがって，文献 6）の考え方が正しいとすれば，本論文で示した一貫性は，上記3軸のうちの1軸に関するもの，と位置づけることができる．残りの2軸に関する一貫性については今後の検討課題としたい．

（2） 一貫性の定義

Payne らは語彙の consistency の例として，ロボットを前方および後方へ移動する指令名はそれぞれ ADVANCE, RETREAT の方が GO, BACK より良い，と述べている．

英語の consistency と日本語の一貫性の間には概念の差があることを認めた上で，Payne らが挙げた例は，一貫性に関する話題から外れていると考える．

上のように，一貫性に関する論文の中で一貫性の意味が不透明なものもあるので，本論文では一貫性が表

す意味を明らかにすることに注力した．

（3） 操作の4側面の一貫性

Reisner が挙げた structural consistency は本論文で挙げた構文的側面の入力側の一貫性に，Payne らが挙げた semantic consistency, syntactic consistency, lexical consistency はそれぞれ概念的側面，構文的側面，語彙的側面の入力側の一貫性に類似している．しかし4側面および3段階（入力側，入出力間，出力側）の一貫性のすべてに言及し，しかも体系化を試みたのは，本論文がはじめてである．

5. おわりに

本論文では対話型システムにおける操作の一貫性とはどのようなものであるかを，実システムの1操作を例題として分析し有用な結果を得た．

本論文で次のことを明らかにした．

（1） 対話型システムを設計する4段階（概念，意味，構文，語彙の各設計段階）を，文献7)を基礎にしてより一般化し整理した．このことは，4段階に強く関係した概念（たとえば一貫性）を明確にする際に，また対話型システムを設計する際に有効と考える．

（2） 1つの操作には，設計の4段階と対応した操作の4側面（概念的，意味的，構文的，語彙的）が存在し，さらに各側面は，入力側，入出力間，出力側という3つの段階に分類できることを示した．

（3） 1つの操作には，（2）で分類した4側面と3段階のそれぞれに対応して，したがって合計 12 種類の一貫性を考えることができることを示し，12種類のそれぞれについて一貫性の概念を形式化した．

（4） 次のような3つの定義を仮説として示した．すなわち，上記 12 種類のそれぞれの一貫性が成立することの定義，4側面のそれぞれの一貫性が成立することの定義，全側面の一貫性が成立することの定義である．これらの3定義は，上記（3）によって分解した個々の一貫性から，順次，より広範囲の一貫性を求める方法とその判定基準とを与えている．上記（3）も含めたこのような系統的な考え方は，一貫性だけでなく，より広い場合に応用できると考える．

（5） 操作の定義域と値域，そして写像の集合のそれぞれを区分するという概念を示した．定義域と値域に区分が存在するという事実，すなわちコンピュータの符号体系そのものが様々な区分から成るという事実は，衆知のことであって何も新しくない．本論文の主張は，表1に示したように区分情報をユーザインタフェースの視点から分類したこと，すなわち，区分情報が4側面と3段階に分類できること，それを一貫性と関係づけたこと，そして主題区分情報という別格の区分情報が存在することを示したことである．なお，表1に例示するように，区分情報は，いわゆる属性情報（全角，半角や色など）を包含する，より広い新しい概念である．

（6） 一貫性の議論を形式化してゆく上で，区分集合とその直積，区分集合間の関係が有力な道具であることを示した．なお，区分という概念の重要性は一貫性においてだけではない．文献8)では同じような概念を境界と称し，境界を「融合」することがユーザインタフェースの使いやすさに通ずることを豊富な例で示している．符号化情報を扱うコンピュータシステムにおいて区分という概念は本質的な重要性を持っているので，この概念は広い範囲で有用と考える．

本研究の今後の課題を述べる．

（1） 仮説が，同一システムのすべての操作に対して，あるいは異なるシステム間の同様な操作に対して，真に成立することを検証する．

（2） 本論文では主題区分情報を用いて一貫性を議論しているが，主題区分情報とはどのようなものであるか，については明らかにしていない．

（3） 利用者にとってどのような区分情報が使いやすさにとってより重要かを検討する．

参考文献

1) 守屋慎次，中谷吉久：インタラクションの図化・分析・検査の一手法，情報処理学会論文誌，Vol. 29, No. 6, pp. 596-604 (1988).
2) 中谷吉久，守屋慎次：インタラクションの図化・分析・検査ツール，情報処理学会グラフィクスとCAD シンポジウム，pp. 53-62 (1988).
3) Payne, S. J. and Green, T. R. G.: Task-Action Grammars: A Model of the Mental Representation of Task Languages, *Human-Computer Interaction*, Vol. 2, pp. 93-133 (1986).
4) Reisner, P.: Formal Grammar and Human Factors Design of an Interactive Graphics System, *IEEE Trans. Softw. Eng.*, Vol. SE-7, No. 2, pp. 229-240 (1981).
5) Kellogg, W. A.: Conceptual Consistency in the User Interface: Effects on User Performance, *Human-Computer Interactive—INTERACT '87*, pp. 389-394 (1987).
6) 守屋慎次：インタラクションの空間と記述ツール系，情報処理学会計算機システムのヒューマンインタフェース—モデル・評価・展望—シンポジウム，pp. 43-52 (1988).

7) Foley, J. P. and van Dam, A.: *Fundamentals of Interactive Computer Graphics*, Addison-Wesley, Menlo Park (1982).
8) 守屋慎次：ユーザインタフェースの研究・開発動向—そのニーズとシーズ—，日経コンピュータ別冊「マルチメディア時代のユーザ・インタフェース」，pp. 21-36 (1989. 7).

(平成3年2月18日受付)
(平成3年9月12日採録)

守屋 慎次（正会員）

昭和48年東京電機大学大学院博士課程修了．工学博士．現在，東京電機大学工学部電気通信工学科教授．昭和56年ニューヨーク州立大学，昭和57年イリノイ大学の各計算機科学科客員準教授．ユーザインタフェース，パターン認識，ストロークシステム，グループウェア，音声入力技法，インタラクションの分析・評価・モデリング法の研究に従事．Interacting with Computers 誌の Special Editorial Board．電子情報通信学会，計測自動制御学会，人間工学会，テレビジョン学会，ACM, IEEE 各会員．

中谷 吉久（正会員）

昭和34年生．昭和57年東京電機大学工学部電気通信工学科卒業．昭和63年同大学院博士課程満期退学．現在，神奈川県工業試験所技術管理部電子計算科勤務．対話型システムにおけるユーザインタフェースの分析・検査・評価法および音声入力装置を用いたユーザインタフェースの研究に従事．電子情報通信学会，計測自動制御学会ヒューマン・インタフェース部会各会員．