インターネット応用特論

６．文字通信：文字コードと国際化（３）、電子メイル、SMTP、MIME

太田昌孝

mohta@necom830.hpcl.titech.ac.jp

ftp://ftp.hpcl.titech.ac.jp/appli6.ppt

文字コードとは？

- ある文字集合に含まれる文字からなる文字列のデジタル化の規則
  - 個々の文字にコード（数値）を振ることだけではない、とはいえ、有限状態なら等価
- 文字集合の含む文字の数が問題
  - 多いと多数のビットが必用
  - 少ないと多様な文字が表現できない
    - 同種の文字の細かな違いも表現できない

ISO 8859／1

- ASCIIの94（95）文字に、西欧のラテン文字（飾りつき文字等）や記号96文字を追加
  - ISO2022を無理やり拡張
    - 33～126を、32～127に
- でもいろいろ変
  - 通貨記号、NBSPには問題あり
  - ウムラウト付yには大文字なし
    - 「ヴ」にひらがながないようなものらしい

ISO－2022－JP
（JUNETコード、RFC1468）

- JUNETでの漢字利用のために開発
- ISO　2022に準拠
- 7ビットで全ての文字を伝送
- エスケープシーケンスにより、G0の文字集合を切り替える
  - 最初はASCII
  - 行末ではASCII（かJIS　X　0201）に戻す
    - 行をまたいでの状態の管理が（ほぼ）不要

ISO－2022－JPの文字集合

- ASCII
- JIS　X　0201（ラテン）
- JIS　X　0208（78年版、83年版）
- JIS　X　0201（仮名、いわゆる半角仮名）は含まない

JIS X 0208の複雑さと単純さ（1）

- 文字数は多い
- 横書きと縦書き
  - あまり真面目に考えてない）
- 片方向（左から右）のみ
  - 現代ではそう
- リガチャー（文字の形の前後関係による変化）が不要
  - 合成用丸はあるが、、、

JIS X 0208の複雑さと単純さ(2)

- 文字同定とその字形に共通認識がない
  - これは大問題
- カタカナひらがなの対応が明確で規則的
  - ヴは例外
  - 飾り文字はない
- 文字幅が一定
- 広く普及していてどこでもつかえる
  - 現在の日本では

文字同定のあいまいさ（包摂）

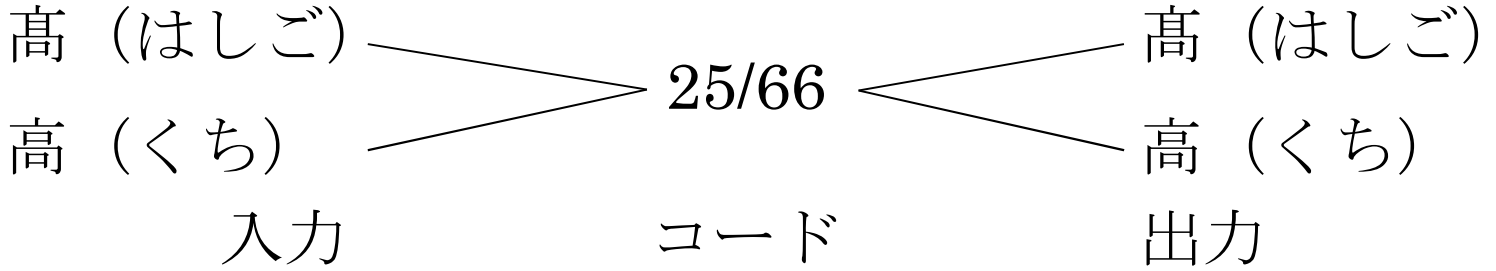
- JIS　X　0208は字形の細かな差を規定しない
  - 「国」と「國」は違う字
  - 「竜」と「龍」は違う字
  - 「高」と「髙」は同じ字
  - 「A」と「a」は？「A」と「Α（アルファ）」は？
- JIS　X　0208は「高」と「髙」という字形を包摂する？！

包摂とは？

- JIS漢字において
  - 一つの区点に、複数の字体が対応した状態

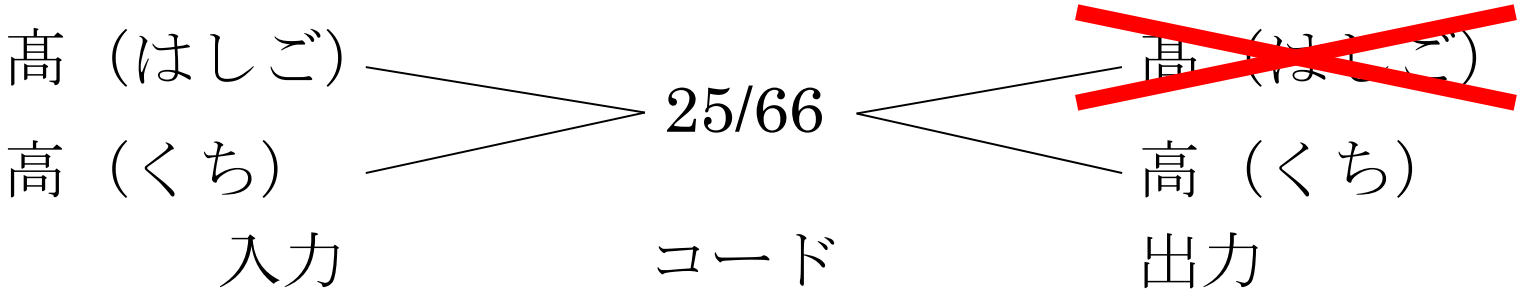
高（くち）と髙（はしご）

    - 代表例：「高（くち）」と「髙（はしご）」
  - 入力時には、どちらの字も同じ区点になる
    - 出力時には、どちらの字体を出力してもよい（？）



包摂とは？

- 入力時には、どちらの字も同じ区点になる
  - は、いいとして、、、
- 出力時には、どちらの字体を出力してもよい（？）
  - 「髙（はしご）」の出力はありえないし実装もないが、入出力は対称だからと言われると、うまく反論できなかった



JISにおける包摂の定義

- 78JIS「漢字の異体字の取扱い」
  - 一つの符号位置に表示されている一つの字形は，ある範囲の変異（ゆれ）を許容し，それらを代表する一例であると考えるべきである
- 97JISでは、ユニコード寄りに
  - 複数の字体を区別せずに，それらに同一の句点位置を与えること
  - 各句点位置では，そこに包摂される字体は相互に区別されない

包摂の問題

- ISO10646（ユニコード）で、日中韓の漢字を同じ区点にする根拠とされている
  - 日中韓の対応する字体のどれも出力可能
    - というか、区別した出力が不可能
- すでにある字体は追加できない？
  - JIS漢字の第3、4水準拡張では、「髙（はしご）」が追加されなかった

文字コードとは？

- 文字列と数値列を対応させる規則？
  - 定義が広すぎて、使い物にならない
  - 文字の入出力なら画像のままで可
- 文字列と数値列を対応させる有限状態の規則
  - 有限状態でないと、検索が事実上不可能！！
  - プレーンテキストは有限状態、構造化テキストはより複雑な状態を持つと、棲み分けるべき

プレーンテキストと構造化テキストの構造

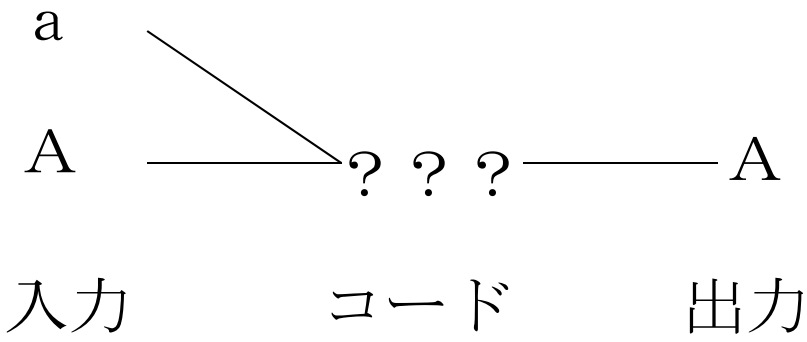
- プレーンテキストでも、ネストしない構造は持てる
  - 一重下線、タブ、フォント切替、等はUNIXの一部コマンドでもサポート、縦横切替も可
  - 十分多くのビットで有限状態を区点に織り込めば
    - 各文字単位にコードが振れる
- ネストした構造は、構造化テキストで
  - 章、節、段落構造、多重下線、ネストした双方向性等

文字と包摂（１）

- 包摂は、JIS漢字コードにおいて初めて出現した概念？
  - JIS漢字は極めて字数が多いため？
    - ラテン文字コードでも包摂あり！

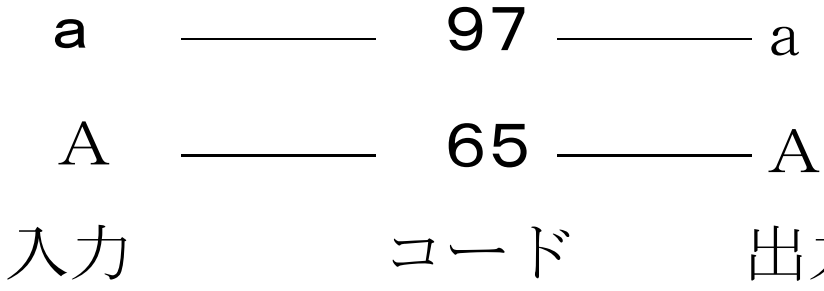
ラテン文字コードと包摂

- 昔、1バイトが6ビットだった頃
  - ラテン文字は大文字のみコード化
  - 小文字は？
    - 入力時には、大文字としてコード化
    - 出力時には、大文字として印字



ラテン文字コードと包摂解除

- １バイトが7ビット（ASCII）になると
  - ラテン文字は大文字小文字別にコード化



  - 大文字小文字混在環境にすんなり移行
  - 6ビット時代のファイルは、そのまま

ラテン大文字小文字は同じ字？

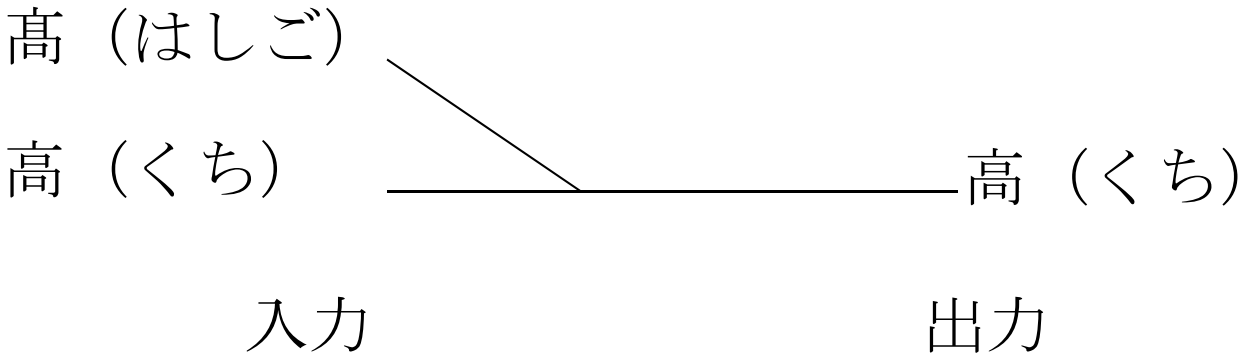
- 6ビット時代には、否も応もない
- UNIXでは、小文字の利用を強力に推進
  - 大文字しか扱えない端末では、Aを「＼A」、aを「A」として入出力
  - コマンドではデフォールトで大小文字を区別
    - UNIX利用者は、大小文字を違う字と認識しがち
- 最近のプログラムの検索では
  - デフォールトで大小文字を同一視
    - 大多数のネイティブの認識？

文字と包摂（２）

- 包摂は、文字コードにおいて初めて出現した概念？
  - 活字やタイプライターでも包摂あり！

活字と包摂

- 昔、活字の組み合わせで印刷していた頃
  - 「髙（はしご）」の字母がない活字では「髙（はしご）」は？
    - 入力時には、「高（くち）」として文選
    - 出力時には、「高（くち）」として印刷



活字と包摂解除

- 「髙（はしご）」の字母がある活字や、字母を追加した場合は、「髙（はしご）」は？
  - 入力時には、「髙（はしご）」として文選
  - 出力時には、「髙（はしご）」として印刷

髙（はしご）　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿髙（はしご）

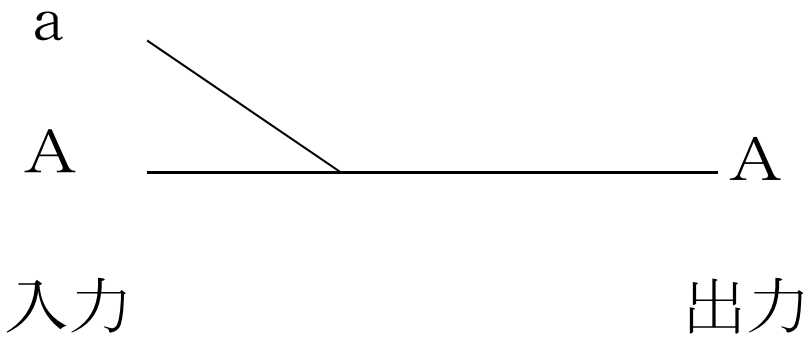
高（くち）　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿高（くち）

入力　　出力

  - 他の印刷物は、そのまま

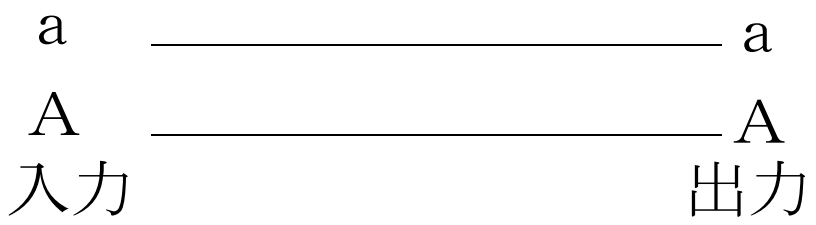
タイプライターと包摂

- 安物（おもちゃ）のタイプライターでは
  - ラテン文字は大文字しか打てない
  - 小文字は？
    - 入力時には、大文字として打鍵
    - 出力時には、大文字として印字



タイプライターと包摂解除

- 本格的タイプライターでは
  - ラテン文字の大小文字が打てる
  - 小文字は？
    - 入力時には、小文字として打鍵
    - 出力時には、小文字として印字



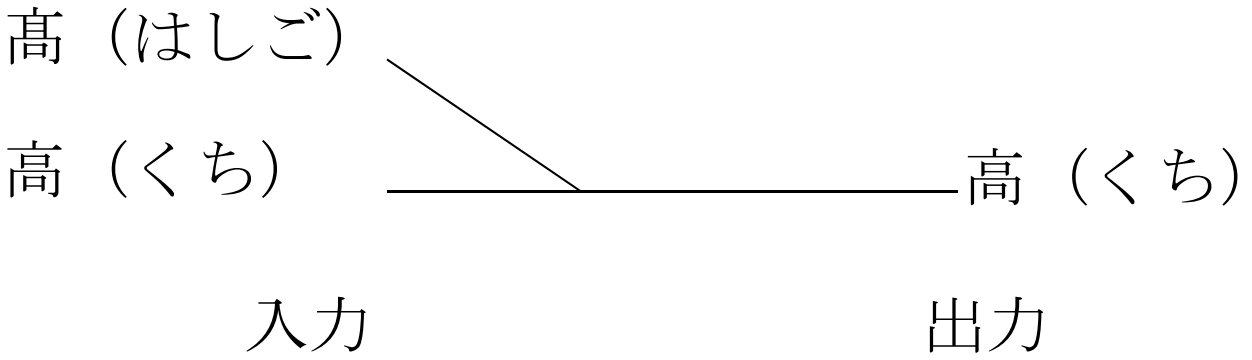
    - 安物（おもちゃ）での印字物は、そのまま

文字と包摂（３）

- 包摂は、活字において初めて出現した概念？
  - 手書き文字でも包摂あり！

手書き文字と包摂

- 「髙（はしご）」は「高（くち）」と同じ字だと思ってる、常用漢字を習った書き手は？
  - 入力時には、「高（くち）」として認識
  - 出力時には、「高（くち）」として手書き



手書き文字と包摂解除

- 書き手が「髙（はしご）」を「高（くち）」と違う字と認識した後の、「髙（はしご）」は？
  - 入力時には、「髙（はしご）」として認識
  - 出力時には、「髙（はしご）」として手書き

髙（はしご）　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿髙（はしご）

高（くち）　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿高（くち）

入力　　出力

  - 過去の筆記物は、そのまま

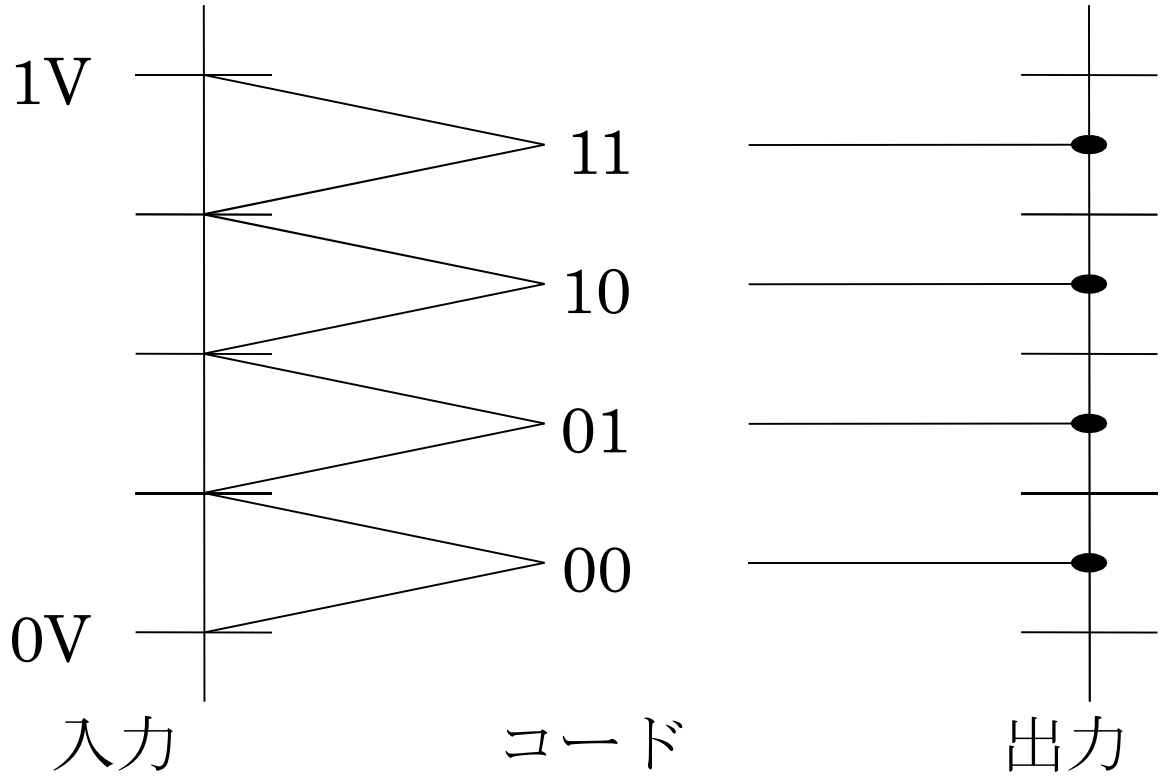
文字と包摂（４）

- 包摂は、文字に内在する概念？！
- では、文字とは？
  - 単なる図形との違いは？
    - 具体的な図形は、アナログ情報
    - 文字は抽象概念であり、デジタル！
      - 包摂は、デジタル化（＝細部の違いを問題にしない）に内在する概念では？
      - 電圧のAD／DA変換と対比してみる
        - » ０～１V、2ビット、リニア

デジタルとアナログ

- デジタルは、細部の違いを問題にしない
  - 多少の雑音や誤差は除去可能
    - 文字は、筆のかすれ、紙の虫食い、石碑の欠け等を問題としないデジタルな概念
- 文字や言語は（話し言葉も）デジタル
  - 書道の字、声、歌はアナログだが
- 問題はビット数（どこまで細部を見るか）
  - １００ビットもあれば、微妙な感情も伝わる

電圧の理想的AD／DA変換



理想的なAD／DA変換では

- 各コードの電圧範囲は0～1Vを4等分
- 0．0625Vも0．125Vも0．1875Vも
  - 代表電圧（0．125V）としてエンコードされる
    - 0．0625Vはエラー（対応するエンコード符号が存在しない）とはされない
      - 1．5Vの入力はエラーとしてもよい
  - 代表電圧（0．125V）としてデコードされる
    - わざわざ0．0625Vで出力することはない
      - 0．125Vの出力で、平均誤差が最小になる

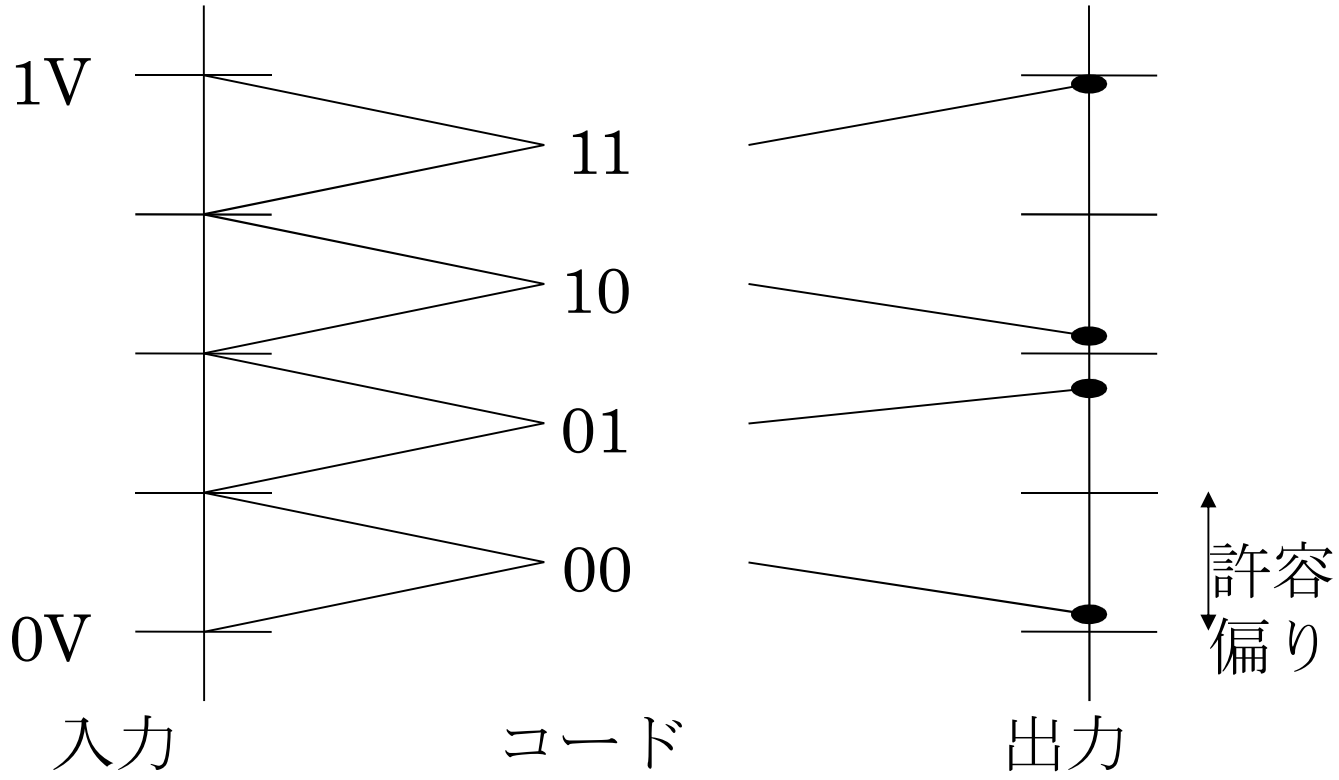
理想的なAD／DA変換に倣った文字コード入出力

- 各区点の字体範囲は規格で厳密に規定
- 「髙（はしご）」も「高（くち）」も
  - 代表字体（「高（くち）」）としてエンコードされる
    - 「髙（はしご）」はエラー（対応するエンコード区点が存在しない）とはされない
      - 全然対応しない字体はエラーとしてもよい
  - 代表字体（「高（くち）」）としてデコードされる
    - わざわざ「髙（はしご）」で出力することはない
      - 「高（くち）」が、入力時の想定出力との差が最小

包摂とは

- デジタル化に伴う量子化誤差！
  - 入力の際にだけ発生
  - ある区点として入力できる字体を全て出力してよいわけではない
    - 理想的なDA変換に対応する文字出力では
      - 出力してよいのは、代表字体のみ
    - ラテン文字コード、活字、タイプライター、手書き等の包摂の場合と同様
    - 現実的なDA変換に対応する文字出力は？
      - 許容誤差があり、これが出力での包摂に見えた

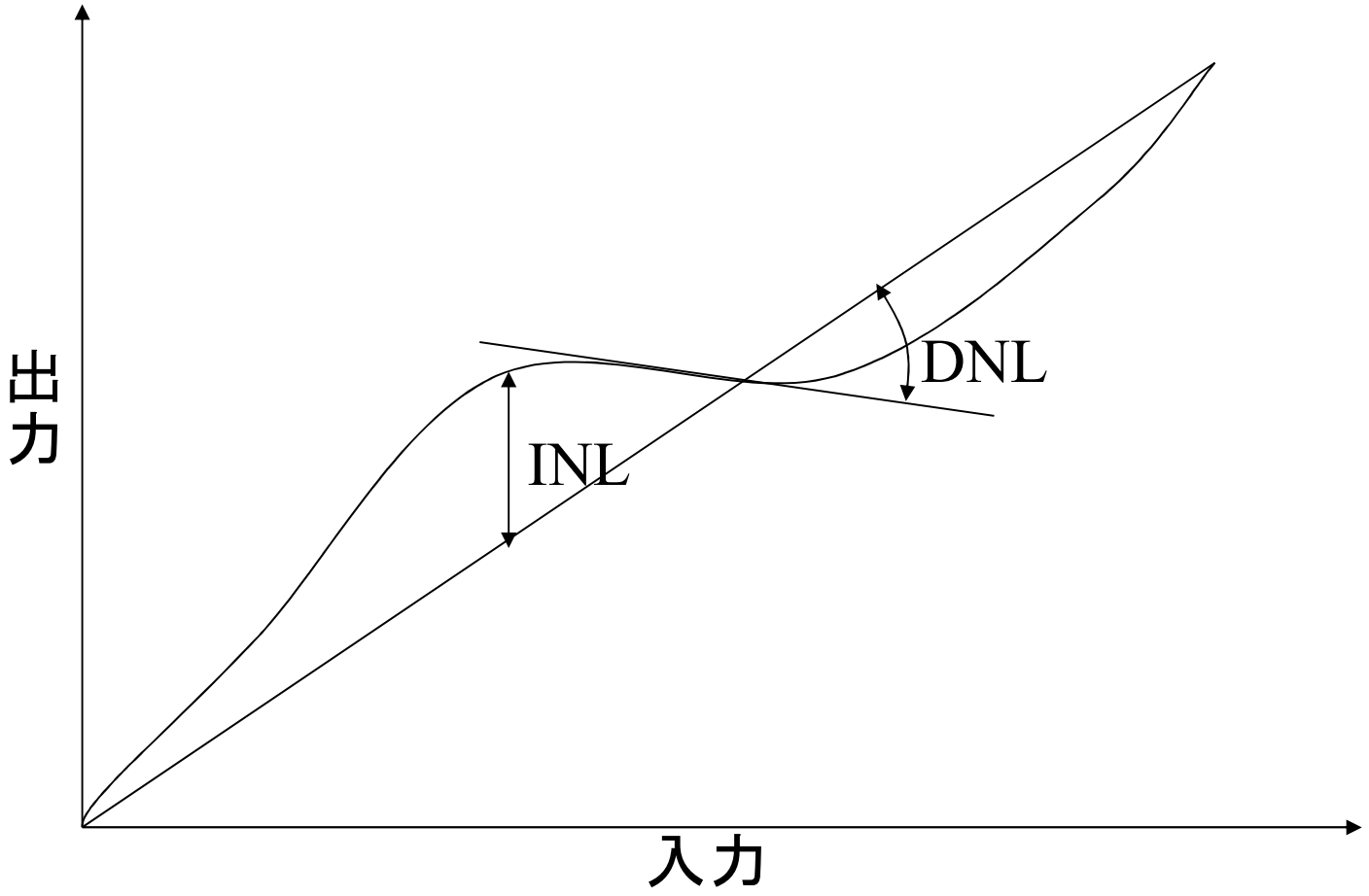
電圧の現実的DA変換



現実的なDA変換では

- 0．0625Vも0．125Vも0．1875Vも
  - 代表電圧（0．125V）としてエンコードされるが、代表電圧（0．125V）からの許容誤差範囲内でのデコードが許される
    - よくある許容誤差は、±1／2LSB未満
      - 単調性が保証され、別のコードは同じ電圧にならない
    - それ以外の許容誤差もいろいろ考えられ、実際に使われている
      - ビット数の少ない場合、±1／4LSB等
      - ビット数の多い場合、INL±6LSB＆DNL±2LSB等

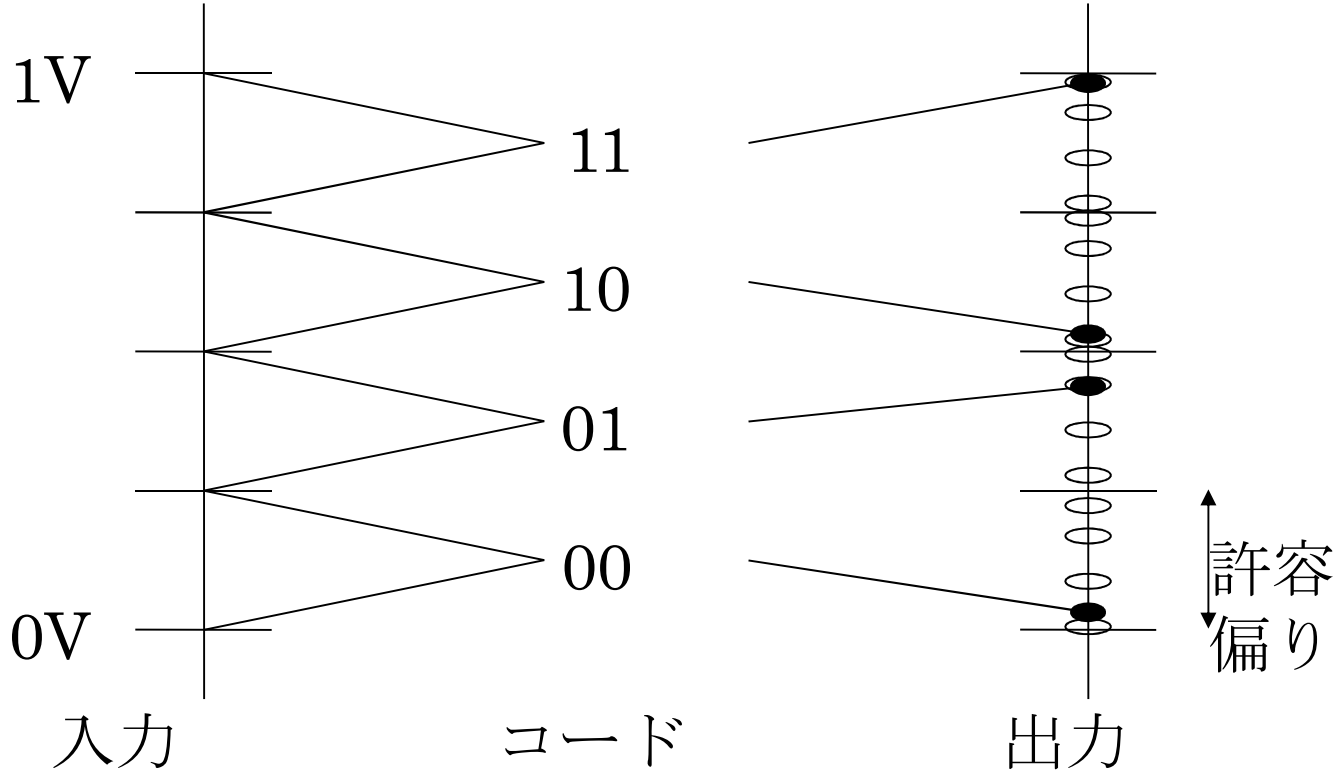
INL(Integral Non-Linearity)と DNL(Differential Non-Linearity)



現実的なDA変換に倣った文字コード出力

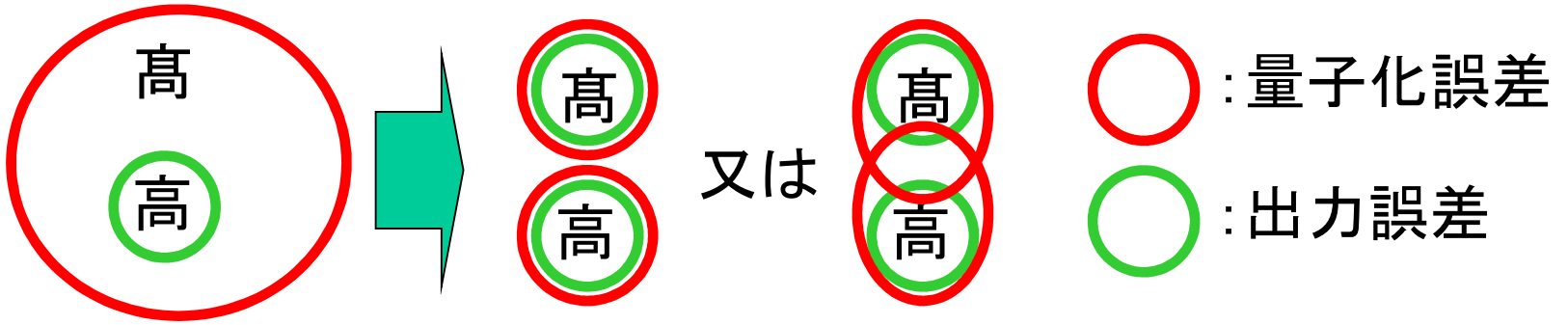
- 「髙（はしご）」も「高（くち）」も
  - 代表字体（「高（くち）」）として入力されるが、代表字体（「高（くち）」）からの許容誤差範囲内での出力が許される
    - JIS漢字（JIS　X　0208：1997）での許容誤差は
      - 「同じ種類の図形文字中の他のいかなる図形文字とも区別できなければならない」
    - それ以外の許容誤差も、いろいろあってよい
      - 文字数の多い場合は、似た異体字が結果的に同じ出力になってもよい（DNLがあることに相当）

複数の代表電圧を認めると許容偏りを小さくできない

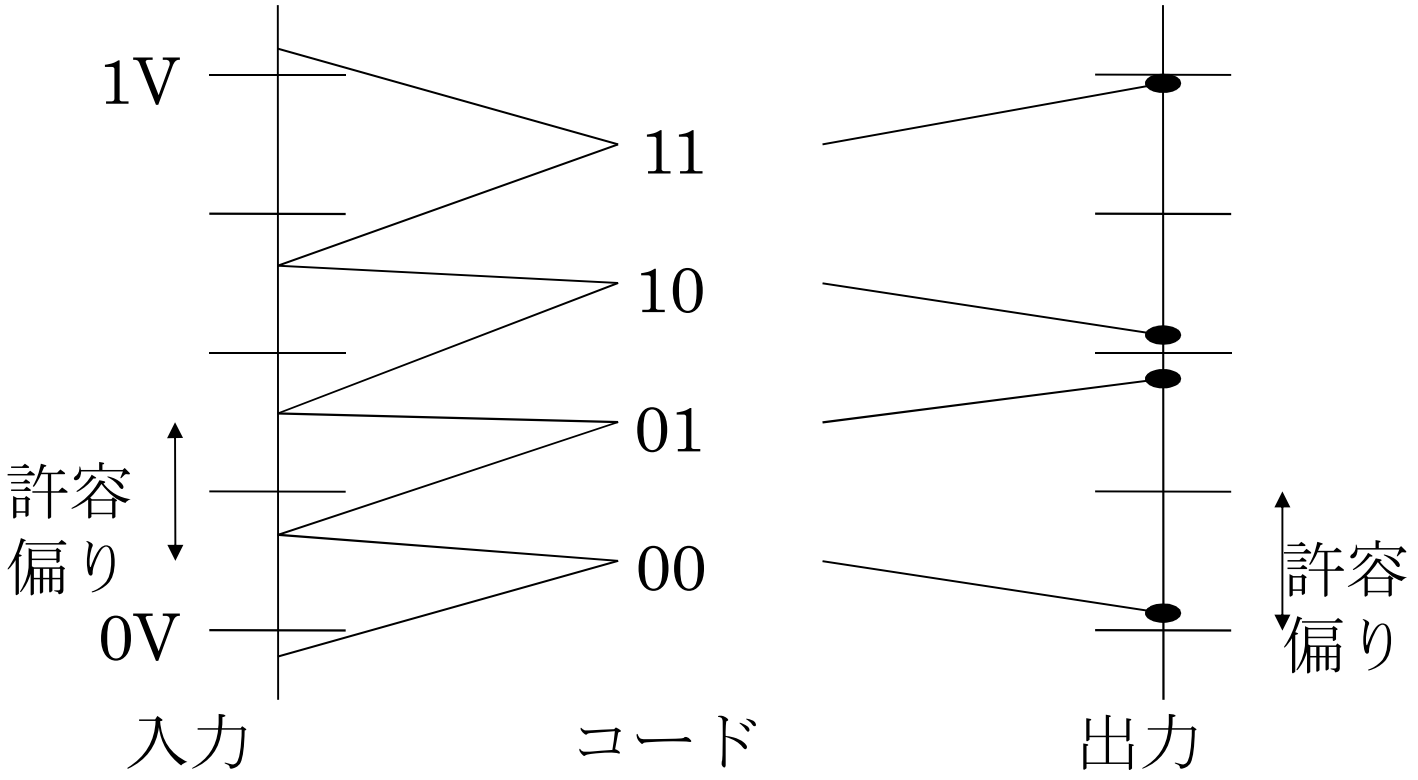


AD／DA変換のビット数増加と JIS漢字への区点の追加

- 電圧範囲の拡張（0～1V⇒0～2V）
  - 全く新たな字体の追加
- 電圧範囲の細分
  - 既存字体類似字体の細分
    - （「高（くち）」から「髙（はしご）」の分離）



電圧の現実的 AD／DA変換



現実的なAD／DA変換では

- 出力にも入力にも誤差は不可避
  - 工業規格は、誤差のないことを要求してはいけない
    - 入力誤差により同じ電圧が別のコードになることも
  - 現実の機器には、許容誤差がある
    - 同じ代表電圧の集合に対して、機器のグレードによって複数の許容誤差があってもよい
  - 入出力を繰り返すと、誤差は累積する

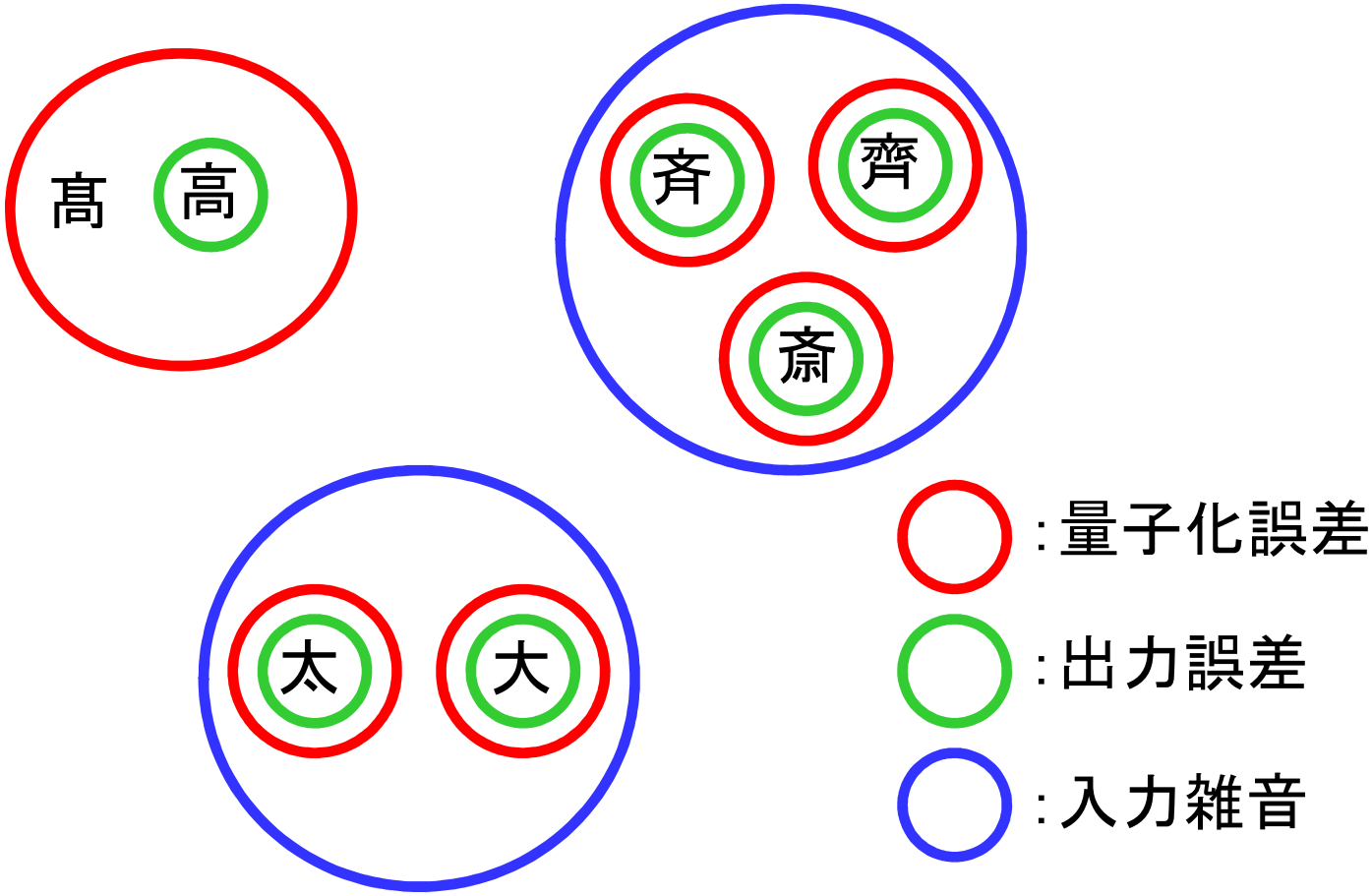
現実的なAD／DA変換に倣った文字コード入出力

- 出力にも入力にも誤差は不可避
  - 工業規格は、誤差のないことを要求してはいけない
    - 入力誤差により同じ字体が別の区点になることも
  - 現実の機器には、許容誤差を定めるべき
    - 同じ代表字体の集合に対して、機器のグレードによって複数の許容誤差があってもよい
  - 入出力を繰り返すと、誤差は累積する
    - 写本を繰り返すと、誤りが累積するようなもの

雑音

- 文字入出力でも、雑音は避けがたい
- 熱雑音
  - 文選工の活字の拾い間違い、仮名漢字変換での変換ミス等
  - 慎重に入力すれば減る（温度を下げる）
- ショット雑音
  - 少数のドットで図形を出力するための誤差等
  - ドット数を増やせば減る（電流を増やす）

JIS漢字の
ありがちな運用実態



代表字体の決め方

- 光速度等の標準とは違って
  - 文字は人間の決め事
    - 「王様の足の長さ＝1フィート」でよい
    - 常用漢字表や康熙字典が、標準原器
      - 表外字体や法務省人名用異体字等も
  - 比較的短期間に変化することも
    - 変化した場合に、代表字体を変更するか、新字体を追加するかは、議論の余地あり
      - 過去の文書の字体を変える必要があるか？

現行JIS規格の問題点とあるべき形

- 区点に複数の代表字体と包摂範囲が規定され、入力誤差が許容されていない
  - 区点には一個の代表字体だけ規定すべき
    - 包摂範囲は常識で判断可能
      - 「高（くち）」が代表字体である区点に「髙（はしご）」をエンコードしてよい
    - 包摂範囲の境界線は、利用者任せにすべき
- 出力誤差の規定が±１／２LSB相当だけ
  - よい出力機器は、代表字体のみを出すべき
  - 機器により、似た字体を同じ図形にしてもよい

ISO10646（ユニコード）の問題点

- 包摂が、入力での量子化誤差である以上
  - 入力では、日中韓の対応する漢字を同じ区点にしてもかまわないが、
    - 「骨」の字等の包摂範囲は、日中で大きく違うが
  - 出力すべきは、代表字体のみ
    - 日中韓のどれか一つ
    - 出力許容誤差として日中韓の字体を認める？
      - 極めて誤差の大きい（性能の悪い）出力装置でしかない
        - » 実際の多くの出力機器の誤差は、遥かに小さい
      - 一時に出力できる字体は一つのみ

包摂と検索

- 包摂があると
  - 文字比較が単純になる？
    - 「高」と「髙」は同じ字
- あいまい検索はどのみち必須
  - 「国」と「國」は違う字
  - 「竜」と「龍」は違う字
  - 「A」と「a」の同一視と同じこと
  - 「太田」と「大田」の同一視が必要な場合も

包摂の害毒

- 「高」と「髙」の書き分けができないのは
  - 一部の応用（含戸籍事務）では致命的
- JIS X 0208が「高」しか含まなければ
  - 後の拡張で「髙」を追加可能だった
- 文字の異同の判断は場合によって異なる
  - 標準化は無意味
  - JIS X 0208は字形だけ載せればよかった
    - もともと文字文化とはそういうもの

包摂規準と標準

- 包摂の基準は
  - 人、用途などによりさまざま
- 文字コードはそれぞれの包摂規準をもつ
  - 音声符号化方式のビット数が違うのと同じ
- 統一文字コードは包摂範囲を狭く

UNICODE

- 全世界の文字を16ビットでエンコードするために米国で生まれた規格
  - そもそも無理だし、ISO　2022があれば不用
- 欧米文字だけは素直にエンコード
- 漢字は日中台韓ごちゃまぜ（無茶な包摂）
  - ハングルは、要素にばらす場合も
- 他の文字は各国のISO2022準拠規格を
  - 16ビット空間に無理に詰め込んだだけ

包摂と電圧のAD／DA変換を対比すると、、、

- CJK漢字は、代表字体が同じでも、包摂範囲は各国で違う
  - そもそも代表字体自体が違う
- それぞれコードを分けないと、無理
  - Cだけ、Jだけ、Kだけの漢字を使い、他が混在しない場合だけは、問題は起きないが、、、
    - 各国の国内規格を使えばそれで十分だし、バイト数も節約できる

ユニコード環境（UTF－８）での日中メイルの混在

- GB－2312の中国字メイルとISO－2022－JPの日本字メイルは普通に読める
- UTF－８で表示される漢字は、字体が変
  - UTF－８の日本字メイル中の漢字（JIS漢字）はJIS漢字用フォントで出力するよう設定
  - UTF－８で中国字メイルがくると
    - JIS漢字はJIS漢字用フォントで、それ以外はUTF－８用の（変な）フォントで出力され、それらが混在

TEXとHTMLでの空白の扱い

http://www.itrc.net/report/meet27/data/6a/usui/index.html

- ラテン文字では、単語間は1つ以上の空白や改行
  - システムは空白一文字と認識
- TEXでは、漢字の間の空白や改行は無視
- HTMLブラウザでは、空白 文字になる
  - なんとかしようにも、ユニコードではCJKV（ベトナム）の漢字が区別不可能
    - ベトナムでは、単語間に空白を入れるらしい

ユニコードの惨状
（wikipediaを参考に）

- 双方向性サポートで有限状態性を喪失
- １６ビット化は破綻（サロゲートペア等）
- 異体字セレクタ（オプション）の導入
  - 文字コードと文字集合のコードの混在
- YEN SIGN問題
  - 日本語用のフォントデータの0x5Cの位置には円記号の字形を当ててしまう
  - 韓国ではウォン記号

LANGUAGE TAG （RFC 1766）

- 各国言語をラベルづけ
  - ISO 639の拡張（日本語はJA）
- 相手の希望に応じた言語を提供する
  - MIMEではContent－Languageヘッダに
- UNICODEで日中台韓の漢字の区別？
  - 言語とスクリプトの区別すらついてない

日本語の表記に用いるスクリプト

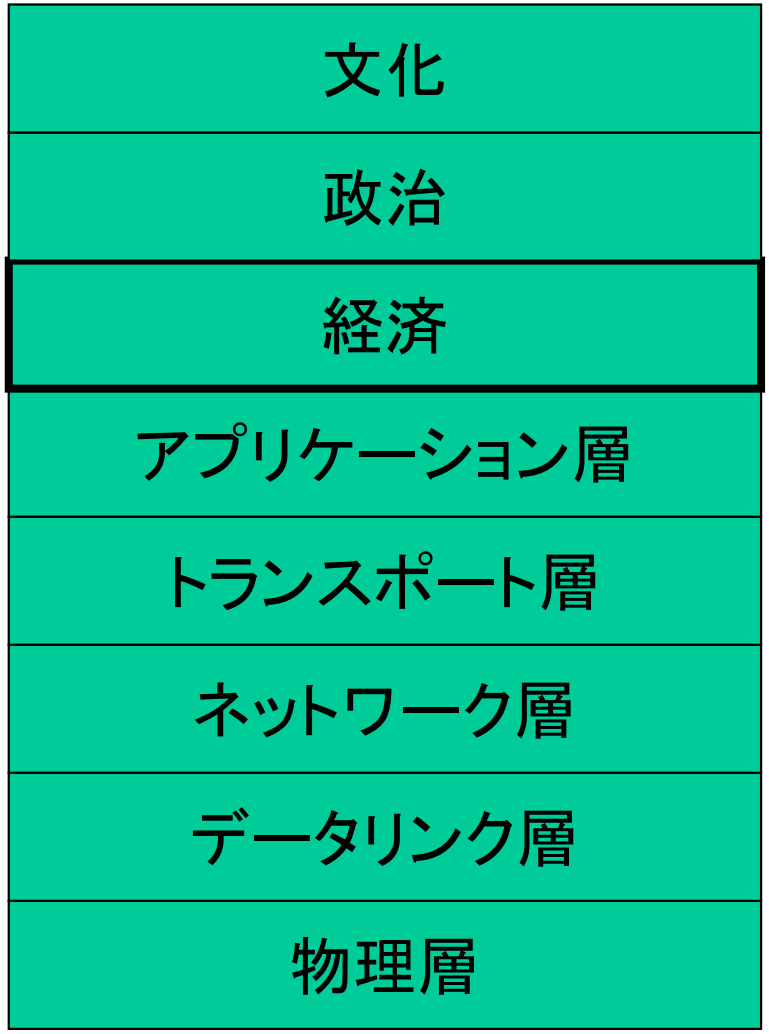
- かな（ひらがな、カタカナ、万葉仮名）
- 漢字かな混じり
- ローマ字（ヘボン、訓令、他）
  - 「まさたか」をフランス語風に書くと“massata ka”
- その他、各地の文字による表音表記

「国際化」ドメイン名

- これまでのドメイン名で使える文字
  - ０－９、Ａ－Ｚ(、ａ－ｚ)とハイフン
- 「国際化」ドメイン名
  - 漢字などもドメイン名に使えるように、、、
- 技術的にはどうにでもなる
  - DNSは8ビット(ただし英大小文字は同一視)
  - ASCII文字にエンコードしてもいい
- 全く使われていない

「国際化」ドメイン名の背景

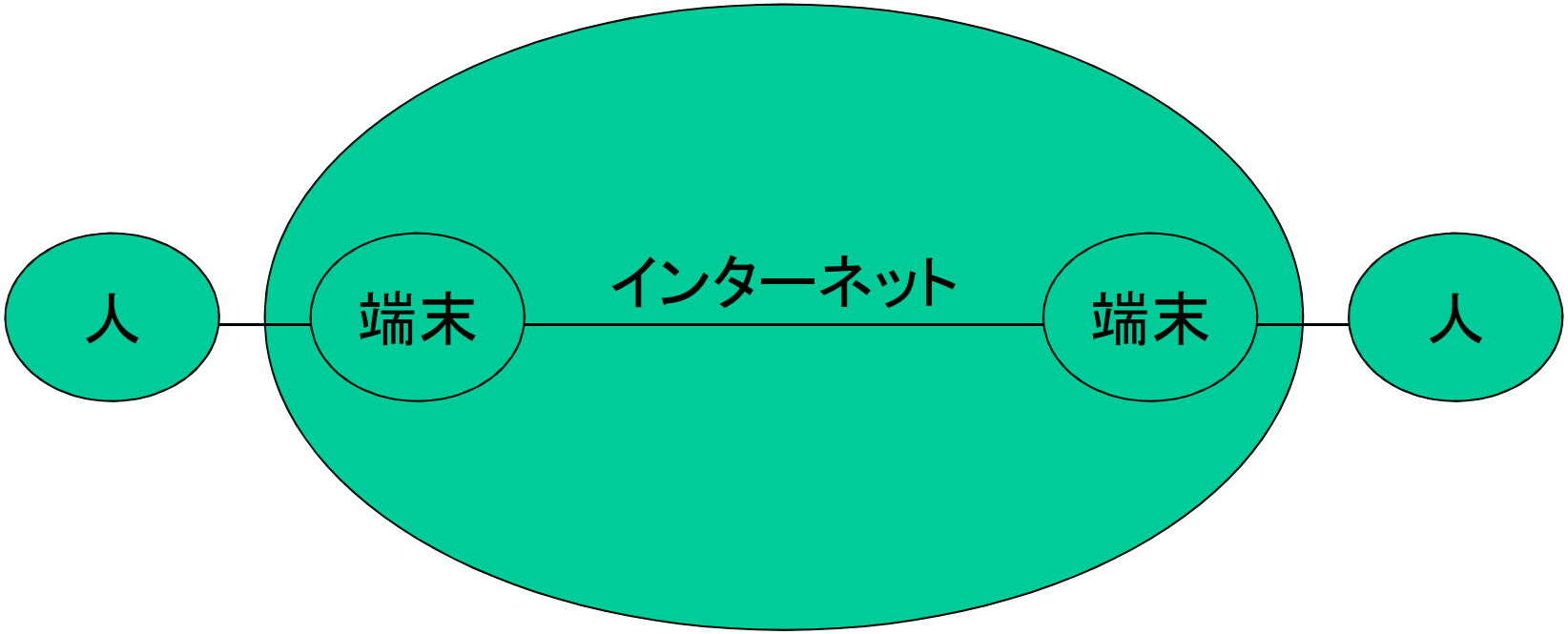
- ドメイン名（＝商標）は商売になった
  - COMの下では1ドメイン名35USD／年
- 新規登録は漸減
- COM対抗ドメイン名の出現
  - BIZ、INFO、MUSEUM、NAME
- COM需要の掘り起こし
- JPNIC（JPRS）はJP需要を振興したい



インターネットの上のレイヤリング構造

インターネットと国際化

- インターネットにより
  - 世界中の端末が接続される
- 全ての端末は国際化されるべき？
  - たぶんそのとおり
- インターネットにより
  - 世界中の人が接続される
- 全ての人は国際化されるべき？
  - ある意味ではそうだが、、、



エンドツーエンド原理は端末では終わらない

国際的に通用する文字

- 英数字と少数の記号のみ
- 漢字ドメイン名は漢字文化圏の外では
  - 文字の一致不一致程度ならわからなくもない
  - 入力は絶望的
- パスポートでも国際航空券でも
  - 名前の表記は英字
- 現在のドメイン名こそ国際化ドメイン名
  - 漢字ドメイン名は局所化ドメイン名

漢字ドメイン名のかかえる諸問題

- 類似商標
  - 「国」と「國」、「竜」と「龍」、「高」と「髙」、「―（はいふんまいなす）」と「一（おんびき）」
    - 国ごとに判断基準は異なる
- 「漢字．JP」や「漢字．日本」は変
  - 「漢字株式会社」を「漢字．会社．JP」に自動変換してくれれば、、、
    - もはやドメイン名ではない

その他の名前空間

- 各種提案はあるが、、、
- 個々の名前が唯一無二ならDNSで十分
- 重複を許すと、結局サーチエンジンにほかならない
  - サーチエンジンなら類似文字検索も自由自在
  - サーチエンジン業者に手数料を支払うと検索結果の優先度を上昇できる

RFC822と電子メイル

- RFC822: STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES
- 電子メイルのフォーマットを規定
- メイルはヘッダと本文からなる
  - ヘッダは空行で終わり本文が続く

ヘッダの構造

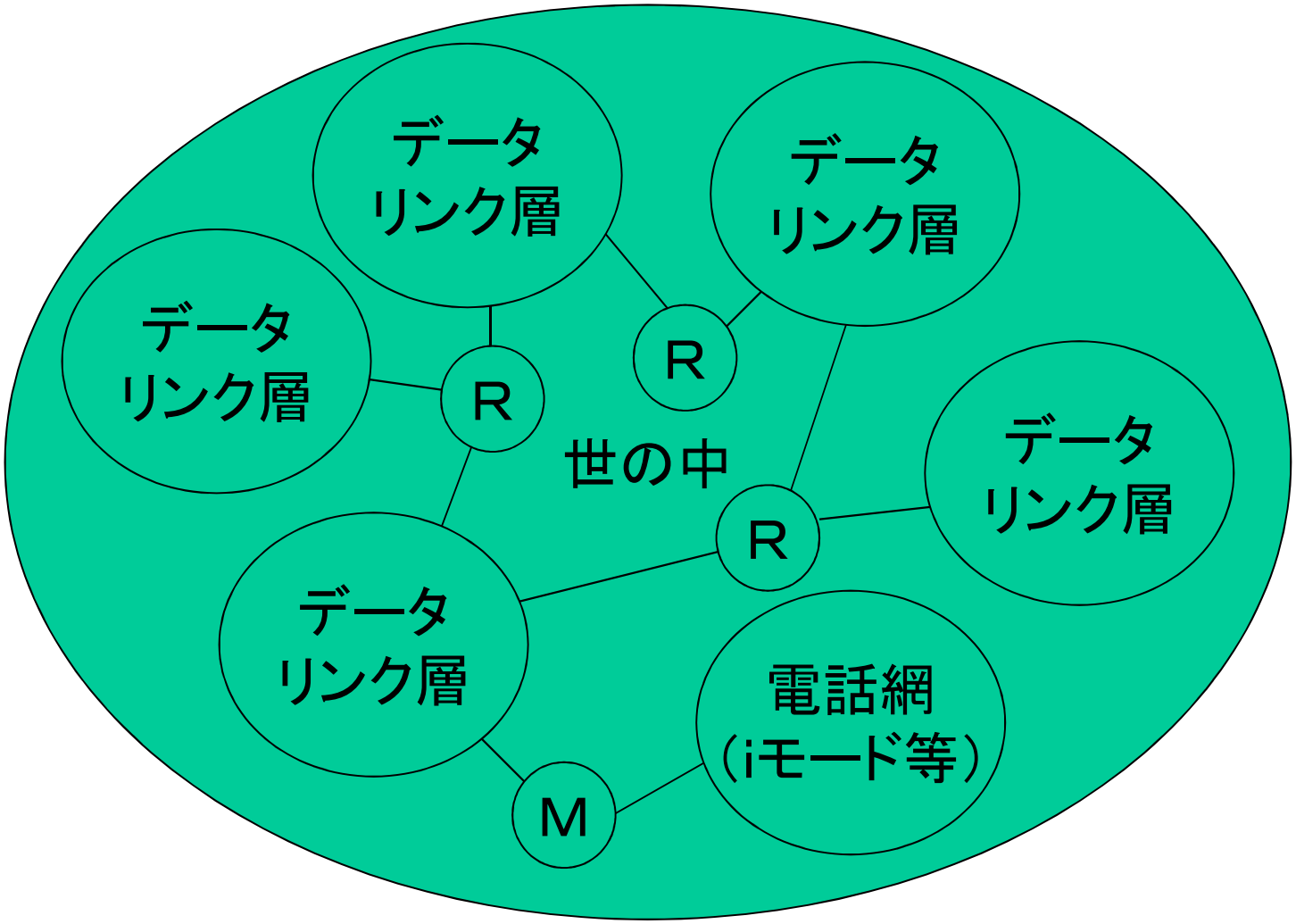
- 複数のフィールドからなる
- フィールドの行頭はコロンで終わるフィールド名で始まる
- 空白文字で始まるのは継続行
- フィールドの内容は
  - メイルアドレス（To:、From:、Cc:等）
  - 日付（Received:等）
  - ただの文字列（Subject:等）

ヘッダフィールドの例

- To:、Cc:、Bcc:
  - あて先
- From:
  - 差出の名義人
- Sender:
  - 実際の差出人
- Received:
  - 転送履歴

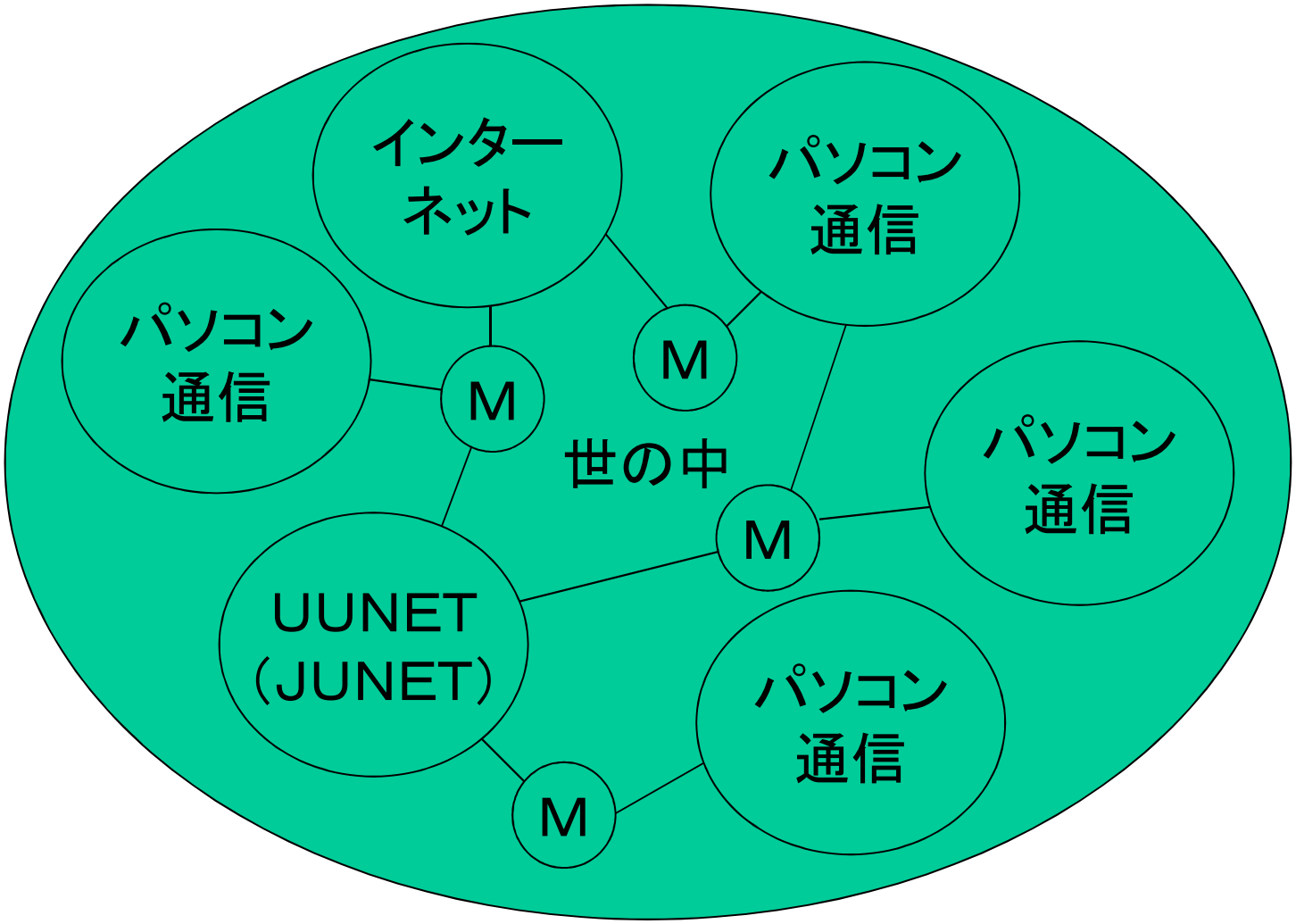
SMTP（RFC821）と電子メイル

- Simple　Mail　Transfer　Protocol
- インターネットで電子メイルをやりとりするプロトコル
- ポート番号25を利用



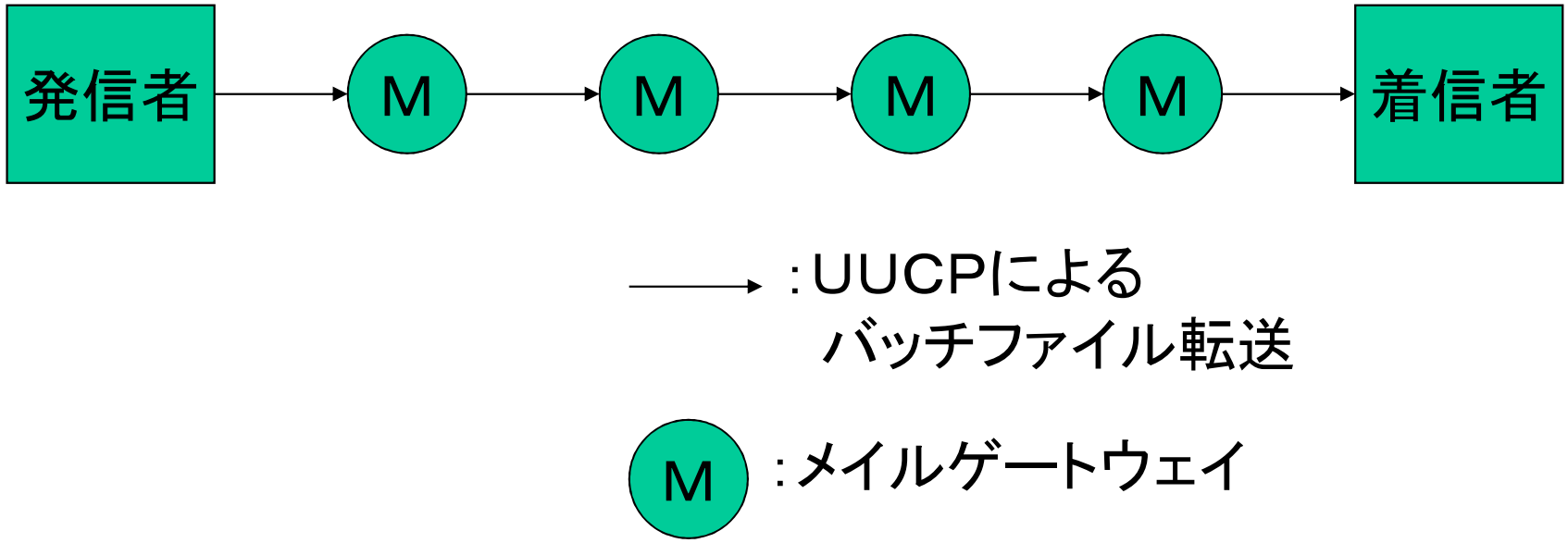
R　：ルータ　　M　：メイルゲートウェイ

現在の電子メイル環境



かっての電子メイル環境とインターネット

UUCP網によるメイル転送



電子メイルと
エンドツーエンド原理

- 電子メイルはインターネット以外でも利用されていた
  - インターネット外部へのメイルも扱う
    - エンドツーエンド原理は成立しない
- 電子メイルは、インターネットにとって、極めて重要なアプリケーション
  - 信頼性は必須
  - エンドツーエンド原理により高信頼化

メイルリレー

- 電子メイルは各種のネットワーク中をリレーされる
- SMTPの配送先が最終目的地とは限らない
  - メイルサーバーはとりあえず受け取るだけ
- 電子メイルは即時に読めなくてもいい
  - 受取人が不在でもかまわない
- 蓄積中継があたりまえ

DNSと電子メイル（RFC974）

- インターネットでの電子メイルアドレスはDNS名（ドメイン名）を利用
- DNSのMXレコードでメイルを受け取るメイルサーバーを指定（複数）
  - １台のメイルサーバーが落ちても他でバックアップ可能
  - メイルサーバーにはプライオリティも
    - 高いプライオリティのサーバが落ちていれば、とりあえず低いプライオリティのサーバで受ける？？

電子メイル配送と マルチホーミング

- 電子メイル（SMTP＋DNS（RFC974））はエンドツーエンドマルチホーミング
  - MXには複数のサーバを指定可能
  - サーバが複数のアドレスを持つ場合
    - 全部のアドレスを試してみる
  - インターネットでもっとも重要なアプリケーションとしては、当然
    - 他にはDNSも

TCPとコマンド

- TCP上でASCII文字でコマンドを出し応答を受け取る
- 行末はCRとLFで区別
- データは同じTCPコネクションで送ってもいい（SMTP）し、別立てでも（FTP）でもいい
  - 同じで送る場合、区切りが必須

SMTPのコマンドとリプライ

- コマンド
  - HELO、MAIL、RCPT、DATA、SEND、SOML、SAML、RSET、VRFY、EXPN、HELP、NOOP、QUIT
- リプライ
  - 3桁の数字＋メッセージ

SMTPのコマンド（１）

- HELO
  - 最初の挨拶（ホスト名の通知）
- MAIL
  - 新しいメイルのコマンド列の開始
- RCPT
  - メイルのあて先
- DATA
  - メイルの本文（ピリオドで終了）、処理の開始

SMTPのコマンド（2）

- SEND、SOML、SAML
  - ログイン中のユーザに直接通知
- RSET
  - リセット
- VRFY、EXPN
  - アドレスの確認、展開
- HELP、NOOP、QUIT
  - ヘルプ、のっぷ、終了

リプライの意味（１）

- １００番台（SMTPでは使わない）
  - 中間報告
- ２００番台
  - 成功
- ３００番台
  - 中間報告（ここまでは成功）
- ４００番台
  - 一時的失敗

リプライの意味（２）

- ５００番台
  - 永久的失敗
- １０の位が０
  - シンタックスエラー
- １０の位が１
  - 一般的情報
- １０の位が２
  - コネクションについて

リプライの意味（３）

- １０の位が５
  - メイルシステムの状態
- １の位
  - その他の区別

コマンド列の例（１）

R: 220 BBN-UNIX.ARPA Simple Mail Transfer Service Ready
S: HELO USC-ISIF.ARPA
R: 250 BBN-UNIX.ARPA

S: MAIL FROM:<Smith@USC-ISIF.ARPA>
R: 250 OK

S: RCPT TO:<Jones@BBN-UNIX.ARPA>
R: 250 OK

S: RCPT TO:<Green@BBN-UNIX.ARPA>
R: 550 No such user here

コマンド列の例（２）

S: RCPT TO:<Brown@BBN-UNIX.ARPA>
R: 250 OK

S: DATA
R: 354 Start mail input; end with <CRLF>.<CRLF>
S: Blah blah blah...
S: ...etc. etc. etc.
S: .
R: 250 OK

S: QUIT
R: 221 BBN-UNIX.ARPA Service closing transmission channel

POPとIMAP

- Post Office Protocol （RFC1939）
- Internet Message Access Protocol （RFC2060）
- メールサーバーからメイルを受け取るプロトコル

MIME （Multipurpose Internet Mail Extensions、 RFC2045～RFC2049）

- RFC822の拡張（複雑化）
  - ASCII文字以外の本文
  - テキスト以外の本文
  - 複数の本文（マルチパート）
  - ASCII文字以外のヘッダー
- 広く普及はしているが、無用だった

ASCII以外の文字

- charsetによるタグ付け
  - “charset＝ISO－2022－JP”
  - 本文中に異なるcharsetは混在できない
    - Multipart／mixedを使えばできる場合もある
      - ただし、インプリ依存
- ISO　2022があれば不用
  - 実際日本はそう
- どうせ知らないcharsetは読めない

8ビット化

- ヘッダでの区別は特殊な文字列で
  - =?CHARSET?B?TEXT?=
- 本文での区別は
  - Content－Transfer－Encodingヘッダ
- Quated　Printableエンコーディング
  - ASCII主体の場合
- Base64エンコーディング
  - 一般（＋、／、０－９、Ａ－Ｚ、ａ－ｚ）

8ビット化は不用

- ISO　2022テキストなら7ビットで十分
  - 実際日本はそう
- バイナリはUUENCODE
  - EBCDIC環境では
    - Base64の文字は透過
    - UUENCODEの文字は変化する場合がある？

テキスト以外の本文

- Content typeによるタグ付け
  - TEXT、IMAGE、AUDIO、VIDEO、APPLICATION、（MULTIPART）、MESSAGE
    - さらにSUBTYPEで細分
- 実情は
  - APPLICATION／OCTET－STREAMのみ
  - ファイル名の拡張子で区別
    - UUENCODEで十分だった

ESMTP（RFC1651）

- Extended　SMTP
- MIMEとともに開発された
- 各種のネゴシエーションが可能
  - もっぱら8ビット転送のための拡張（RFC1652）

ISO－2022－KR（RFC1557）

- 韓国でハングルや漢字をエンコードする7ビット文字コード
- ASCIIがG0、KS　C5601がG1
- SI／SOで切り替え
- SIを使う行では、その前に、G1をKS　C5601に指定
- 同RFCではEUC－KRも指定

ISO－2022－JP－2（RFC1554）

- ISO－2022－JPの拡張
  - KS　C　5601（韓国）、GB2312（中国）、ISO　8859／1、ISO　8859／7(ギリシャ文字)も追加
- 94文字集合はG0、96文字集合はG2で利用（SS2（ESC＋‘N’）
- 7ビットだが、127という数値は出現

まとめ

- 包摂とはデジタル化の際の量子化誤差
  - 出力誤差と混同したUNICODEは破綻
- 国際化議論には出鱈目が多い
  - 国際化ドメイン名、LANGUAGE　TAG
- 電子メイルのフォーマットはRFC822
  - MIME拡張は不要であった
- 電子メイルの配送はRFC821
  - ESMTP拡張は無用

ドメイン名についての RFC821 (SMTP)の 正しい解釈を巡る混乱

太田昌孝
東京工業大学情報理工学研究科
mohta@necom830.hpcl.titech.ac.jp

Tony Finchのメイル（２０１４年３月）

https://www.ietf.org/mail-archive/web/dnsop/current/msg11925.html

- CNAME pointing at MX is a different problem, which does not work consistently in practice. The requirement in RFC 1123 is a restatement of RFC 821 section 3.7 (last paragraph) and page 30 (penultimte paragraph).

RFC821の記述（ドメイン名にエイリアスは許さない）

- Whenever domain names are used in SMTP only the official names are used, the use of nicknames or aliases is not allowed.
- Hosts are generally known by names which are translated to addresses in each host. Note that the name elements of domains are the official names -- no use of nicknames or aliases is allowed.

RFC821をうけてのRFC1123

The domain names that a Sender-SMTP sends in MAIL and RCPT commands MUST have been "canonicalized," i.e., they must be fully-qualified principal names or domain literals, not nicknames or domain abbreviations. A canonicalized name either identifies a host directly or is an MX name; it cannot be a CNAME.

The sender-SMTP MUST ensure that the <domain> parameter in a HELO command is a valid principal host domain name for the client host.

実際の要求(RFC6409)

- Nonetheless, unconditionally resolving aliases could be harmful. For example, if www.example.net and ftp.example.net are both aliases for mail.example.net, rewriting them could lose useful information.

  www.example.net CNAME mail.example.net

  ftp.example.net CNAME mail.example.net

  mail.example.net MX 0 mx.example.net

別名が有害な例

- 昔の脆弱な実装ではループが起こりうる

  cname.example.com CNAME mx.example.com

  mail.example.com MX 0 cname.example.com

  MX 1 other.example.com

  MX 2 mx.example.com
- これは、MXの右側が別名の場合
- MXの左側は？
  - 特に害はない
  - RFC821, RFC1123では、なぜ禁止されているのか？

ドメイン名とホスト名の歴史

- 当初は、hosts.txtというISIが管理するファイルを共有し、ホスト名とIPアドレスを相互変換
- インターネットの巨大化に伴い、緩やかに統合された分散データベースとしての、ドメイン名の導入
  - RFC1034,1035で、仕様は一応完成
    - ホスト名とIPアドレスの変換以外の機能も含む
      - メイルアドレスやメイルドメイン名とメイルサーバの対応に関しては、MB, MD, MF, MG, MINFO, MR, MXと、多種多様の試みがあった
    - でも最初は、DNSにはホスト名とIPアドレスの変換の機能しかなかった？

RFC881には、メイルアドレスからメイルサーバ名への変換機能

- The domain server design also provides for mapping mailbox addresses to the host name of the mail server for that mailbox. This feature allows mailboxes to be related to an organization rather than to a specific host.
- RFC819 (RFC821と同時に策定)には、同様の記述はない
  - メイルドメイン名とホスト名の区別はない？
  - メイルドメイン名がメイルサーバのホスト名？
  - MXの使い方を定めたRFC974でも認めている（古い仕様だったとは書いてないが）

RFC821の意図

cname.example.com CNAME mail.example.com

mail.example.com　A 192.0.2.1

のようなCNAMEを禁止しているだけ

- RFC821の記述には問題はない
- MXの導入に伴い、MXの右側の別名だけを禁止すればよかった
- RFC1123のRFC821の解釈が間違っていた

RFC821をうけてのRFC1123

The domain names that a Sender-SMTP sends in MAIL and RCPT commands MUST have been "canonicalized," i.e., they must be fully-qualified principal names or domain literals, not nicknames or domain abbreviations. A canonicalized name either identifies a host directly or is an MX name; it cannot be a CNAME.

The sender-SMTP MUST ensure that the <domain> parameter in a HELO command is a valid principal host domain name for the client host.

おわりに

- RFC821とRFC1123の理不尽に見える仕様について、考古学的考察を行った
- RFC821の仕様は、当時はMXのような仕組みがなく、メイルドメイン名がそのままメイルサーバ名になることを想定しているようである
- MX導入後のRFC1123（RFC1035の1年11ヶ月後に発行）で、RFC821の解釈を間違ったようである
  - 当時既にDNSが当たり前すぎて、RFC821の時代にも当然存在したと思っていた？
  - インターネットが急速に発展・普及した結果だとすると、他にも同種の誤解があるかも