7E−3

YyonX 実行モデル (3) — ウィンドウ入出力機構 —

田中啓介[†1], 古坂孝史[†2], 井田昌之[†1]
青山学院大学情報科学研究センター 研究教育開発室

1 はじめに

複数プロセスが協調して動作する分散システムにおいて，処理をどのように分散するかは全体の性能を左右する．特にユーザとの会話を必要とするシステムでは，低速な会話入力と高速な処理との適切な融合をはかることにより全体の性能を向上させることができる．

本稿ではサーバ / クライアント方式に基づくウィンドウツールキットである *YyonX* におけるウィンドウに対する入出力処理の方式を検討する．

2 モデルに適した入出力の方針

YyonX は Page モデルと Viewport モデルの二つの実行モデルを持つ [1]．両者は入出力の特性が異なるため，各々に適する入出力方式を検討する必要がある．

2.1 入力処理

入力処理においては通信量だけでなく，ユーザとアプリケーションとの間の仲介機構として，幾つかの考慮点が存在する．特に，*YyonX* が Lisp アプリケーションをその主たる対象とすることから Lisp 処理系に対する入力機構としての特質は重要な考慮点となる．

入力反応速度は主に Page モデルで重要である．

2.2 出力処理

出力処理においては，1) *Yy-client* と *Yy-server* との間欠同期，と 2) *Yy-server* と X サーバとの間の間欠同期，とが存在する．

すべてのオペレーションに対して同期をとることは，確実な動作の終了を保障できる反面，通信量を増大させ，全体としての性能に影響を及ぼす．

そのため，実現系では次のような対応を行なっている．1) に対しては，命令のための通信時には自動的に *Yy-server* − *Yy-client* 間の同期をとる．描画指示などコマンドに関しては設定した回数の実行ごとに同期のための通信を行なう．この間欠同期のタイミングを調整すれば，全体としての出力速度を高めることができる．

2) に対しては，*Yy-server* が *Yy-client* からの通信が途絶えて一定時間を経過した場合に X Server との間の同期をとるという方式を採用している．

これらの処理は主として Viewport モデルにおける出力処理を意識したものである．

3 入力における機能分散

YyonX のサーバ / クライアントモデルにおいて入力機構としては以下の方式が考えられる．

A: クライアント単独方式

Yy-server が X Server から受けとったの入力データをそのまま *Yy-client* に渡すという方式．

B: サーバ単独方式

Yy-server 側で必要な情報をすべて編集した形で *Yy-client* に渡すという方式．

入力機構は完全にアプリケーションから独立した形になる．入力時のエコー処理，編集処理は *Yy-server* によって行なわれる．

C: クライアント / サーバ機能分担方式

高速性の必要な入力エコー処理などを *Yy-server* 側単独で行ない，必要に応じて *Yy-client* に入力状況を伝えるという方式．

4 入力機構における考慮点

4.1 サーバ / クライアント間の通信

方式 A ではすべての入力処理はクライアントに集中している．そのため，*Yy-server* の処理の入力処理における負担は小さくなる．但し，X Server と *Yy-client* の間に処理を何も行なわないプロセスが存在することになり，入力のための通信におけるオーバヘッドは増す．

方式 B では，クライアント側の入力処理における負担は軽減される．入力データとそれをエコーするための通信は X サーバと *Yy-server* の間でのみ行なわれる．これは一般の X Window System のクライアントの入力処理に対応する様式であり，入力時の反応速度を一般的な X クライアントと同等にすることができる．しかし，*Yy-server* は画面をテリトリと呼ばれる矩形の領域に基づいて管理しており [3]，画面上に描画された文字・図形に関しての意味を理解しない．そのため，入力処理にともなう画面描画処理が必要な場合には，描画に必要なすべての情報をその都度 *Yy-client* から得る必要がある．

この方式 B の特質は方式 C でも同様に存在する．方式 C と方式 B との違いは入力の途中で必要に応じて *Yy-server* と *Yy-client* の間で通信が行なわれる点である．但し，この通信は *Yy-server* の入力エコー処理等に直接的な影響は与えない．

4.2 基本処理系との同期

YyonX は Lisp 処理系を基本としたウィンドウツールキットであり，基本オペレーションが Lisp プログラムによって可能となっている．そのため，入力機構は Lisp 処理系を考慮した機構とする．

UNIX 上の Lisp 処理系では，リターンキーの押下によって入力行が Lisp 処理系に渡される方式をとるものが多い．これに対して，Lisp の入力処理においては，リターンキーと入力の確定は必ずしも対応していない．また，Lisp Listener への応用を考えると ')' に対応する '(' を入力時に判別できるような機構を提供することも必要になってくる．

このように Lisp 処理系に依存した入力を単純に実現するには，細かい制御の可能な方式 A が最適である．方

Input/Output Mechanism on Windows of *YyonX*,
Keisuke TANAKA[†1], Takashi KOSAKA[†2], Masayuki IDA[†1],
[†1]Aoyama Gakuin Univ., [†2]Aoyama Gakuin Univ./CSK Corp.

表 1: 三つの入力方式の比較

比較点	方式 A	方式 B	方式 C
Lisp との同期	とりやすい	とりにくい	Server-Client 間のプロトコルにより可能
入力データのための通信	大	小	中
入力時制御のための通信	小	大	中
かな漢字変換	Client が行なう	Server が行なう	入力編集の主担当側
実画面の Client に対する同期	とりやすい	とりにくい	Server-Client 間のプロトコルにより可能

式 C では *Yy-server* 上に Lisp に対する入力であることを意識した機構を備える必要がある．不足する機能は *Yy-client* との通信によって補うことができる．方式 B の場合は *Yy-server* 上に Lisp 処理系に匹敵する入力処理機構を準備する必要がある．

4.3 漢字入力機構との整合性

入力機構においてかな漢字変換をサポートする場合，入力時の編集処理とかな漢字変換のための編集処理が大きく異なっては円滑な入力作業に支障がある．そのため，入力の各方式においてはその入力のための編集を主として受け持つプロセスが変換処理を行なう構成とする方が入力機構を有効に活用できる．

5 評価と実現

これら 3 方式の特徴を表 1 にまとめる．各方式の特徴に照らし合わせて実現モデルに対応した方式を採用する．

5.1 Viewport モデルにおける実現

Viewport モデルでは高い応答性は要求されないため，入力処理も方式 A, B, C のいずれでも実現可能である．しかし，Viewport モデルにおける文字描画位置の制御，画面制御自体は複雑であり，そのために A 以外の方式では *Yy-client* から *Yy-server* へ伝える制御情報が多くなってしまう．入力の即時性をそれほど重視しないモデルであるため，総合的には A 方式が適切である．

この方式は *YyonX* Version 1.0 で実現している．

5.2 Page モデルにおける実現

Page モデルでは応答の高速性が要求される．この場合，実測値の分析 [2] からも明らかなように，方式 A で即時反応を保障することは無理がある．また，*Yy-client* ではアプリケーション本来の処理が中心となることから方式 B, C が有効である．また，Lisp 処理系との協調の面からは A, C をとることが妥当である．これらを総合して以下のような C の方式を採用する．

1. Lisp の入力方式に合わせて，a) 入力のエコーを行なう行編集付き入力，b) 入力のエコーを伴わない位置文字入力，の二方式を提供する．

   これらは，各々独立した *Yy*-Protocol の命令によって起動される．

2. 入力のエコー処理はサーバ側で行なう．同時に，カーソル処理，スクロール処理も行なう．

   これによって入力時の反応速度を高めることができる．*Yy-server* でのウィンドウ描画速度は一描画単位平均 7msec (SUN4/280 での実測値) であり，[1] で設定した目標値を十分に達成できる．また，入力時の単純なスクロールであれば，5msec 程度で処理できることから，このスクロール処理を *Yy-server* で受け持つことは意味がある．

3. 入力時の中間状態の情報は *Yy-client* に送られる

   Yy-client では必要に応じて入力処理の中断，画面位置の変更等をサーバに対して行なうこともできる．これにより，'(' と ')' の対応機能など Lisp 処理系への入力を意識した操作が *Yy-client* から可能になる．入力エコー処理やスクロール処理は *Yy-server* が単独に行なうために *Yy-client* による特別な操作のための通信時間はユーザへの応答時間には関係がない．

4. かな漢字変換はサーバが行なう

   かな漢字変換のためのプロセスは Wnn の jserver を用いる．これによって，辞書を含めたかな漢字変換システムを *Yy-server*, *Yy-client* 共にその内部に持つ必要がなくなる．通信に要する時間は平均して 1msec 以下である (実測値による) が，変換の性能そのものは jserver に依存することになる．

   かな漢字変換機構の *Yy-server* への組み込みも可能であるが，一般に配布できる辞書データが入手できないため今回は実現していない．

この方式は *YyonX* Version 2.0 で実現した．そのため，*Yy*-Protocol に，文字カーソル位置の位置指定，画面描画情報の交換，入力途中状態の通知，入力結果の通知の機能を付加した．

画面制御を行なうためには，*Yy-client* と *Yy-server* の間で制御情報のやりとりが必要になってくる．この通信量の増大はシステム全体の性能の低下につながる．そこで，以下のような特殊化を行なう．

a) 原則として *Yy-server* ではカラム・行単位の座標管理を行なう．

b) 描画の方向は縦，横いづれかに固定される．

これは，Page モデルの特性から妥当な特殊化である．

6 おわりに

入力方式の決定のための基礎データは，方式 A の入力機構をもった Version 1.0 上で得た．ここで述べたすべての入出力機構は Version 2.0 に組み込まれている．

参考文献

[1] 井田昌之他: "*YyonX* 実行モデル (1)", 情報処理学会第 41 回全国大会, Sep. 1990.

[2] 古坂孝史他: "*YyonX* 実行モデル (2)", 同上.

[3] 田中啓介他: "*YyonX* における *Yy-server* の設計", 情報処理学会第 40 回全国大会, Mar. 1990.