3E-5

Apostle: 協調結合に基づく分散処理システム(2) ―― システムの実現 ――

田中啓介，井田昌之
青山学院大学情報科学研究センター

1 全体構成

Apostle は，現在，青山学院大学情報科学研究センターで進めている Trinity 構想の実現形態である[1]. *Apostle* は，3キャンパスに設置される計算機(Component Computer)間結合を基盤とする．その主要な構成要素は，それぞれの Component Computer に実現される Information Server (IS), User Interface (UI), File Cacher (FC) の3つの機構であり，これらの機構により利用者に対してキャンパス間での統一的な計算機利用環境を提供する．図1に *Apostle* 全体の構成を示す．各構成要素は UNIX のプロセスとして実現され，機能の呼び出しはプロセス間通信機能を用いて実現される．

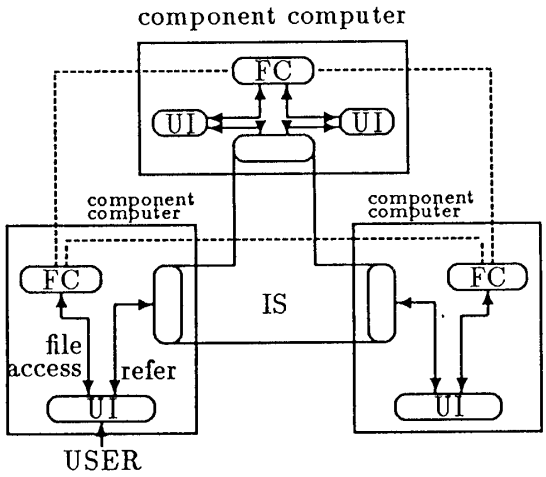

図 1: *Apostle* の全体構造

2 Information Server (IS)

IS は3キャンパスに分散したデータベース管理機構であり，三者が協調して一つのデータベースを管理する．データベースは，*Apostle* の動作に必要な利用者に関する情報を内容とする．

2.1 データベースの内容

データベースの内容は以下の通りである．

- *Apostle* 上での利用者認識名: 通常，課金処理に用いられる認識名は順番号に基づくため，直感的な利用者の認識には役立たない．従って，別の認識名を導入する．
- 管理者による課金処理用認識名
- 環境情報:
  統一的な利用環境を設定するために必要な個人情報であり，1) 個人の private file を保管するためのホームディレクトリ，2) メールボックス，3) 利用者が参加するグループに対してのホームディレクトリ，に関して物理的な位置(Component 名とパス名) が管理される．
- その他の個人情報:
  利用者の本名や所属部門，住所などが割り当てられる．これらの情報は Apostle システムそのもののためには用いられないが，今後開発されるシステム上のアプリケーションに対して提供される．

Apostle: A Distributed System
based on the Harmonic Connection(2)
–Implementation of *Apostle*–
Keisuke TANAKA, Masayuki IDA
Aoyama Gakuin University

2.2 データの参照，更新

3キャンパスの各 Component Computer において同一のデータが管理されることから，データの参照は各 Component Computer 内部で処理される．これによってデータ参照に対してキャンパス間の通信を行なう必要がなくなる．

しかし，常に各キャンパスでのデータベースの内容を同一に保つために更新の際に特殊な機構を必要とする．この更新動作は，以下の手順で行なわれる．

1. 更新要求の発生．この要求は Component Computer 内部で行なわれる．
2. 更新要求の正当性の確認．
3. 全ての Component Computer が動作していることを確認．動作状態にないものが存在する場合は，その回復を待つ．
4. 全ての Component Computer が動作している場合にのみ変更情報が伝えられ，データの更新が行なわれる．
5. 同時に互いに矛盾する変更要求が与えられた場合，その要求は失敗する．

この更新手順では，データベースの維持，管理に関するキャンパス間の通信量を最小にするため，一貫性の確認のような機能は提供しない．そのため，ある Component Computer が停止している状態では要求が待ち状態に入り直ちにデータベースに反映されることがないという欠点を持つ．

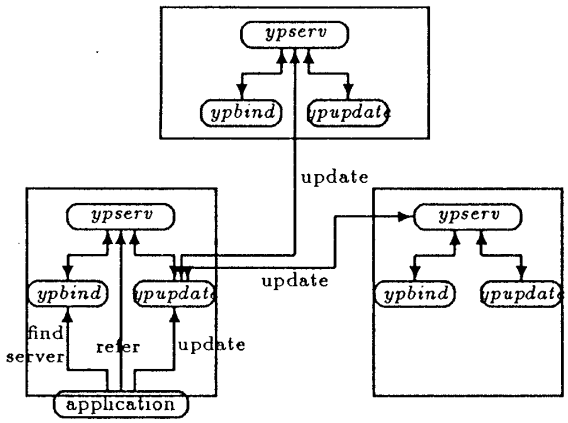

図 2: Information Server の構造

2.3 実現

IS は SUN Microsystems によって提供されている Yellow Pages (YP) システム[2]の拡張によって実現される. SUN が標準で提供する *ypserv(8)*, *ypbind(8)*に *Apostle* 用データを管理する上での変更を加えた上で, データの更新を専門に行なう *ypupdate* を新設した. これらの関係を図 2 に示す.

この実現方法においては, YP プロトコルそのものには変更を加えていない. これは *Apostle* システムの外部に存在する計算機の持つ機能との互換性を保証するためである.

3 User Interface (UI)

UI ではファイルタイプを以下の 4 種とする.

1. private file:
   各個人用のファイルであり, ある一つの Component Computer に置かれる. 基本的に所有者以外の利用者はアクセス出来ない.
2. group file:
   何人かの利用者で共有されるファイルであり, ある一つの Component Computer に置かれる. グループのメンバー以外の利用者はアクセス出来ない.
3. public file:
   一般コマンドやデータファイルであり, Component Computer 毎に置かれる.
4. system file:
   各 Component Computer のオペレーティングシステムが動作するために必要なファイル.

また, UI では利用者を 1)通常の利用者である一般利用者, 2)利用者登録や利用環境の設定等の一般的な管理作業を行なう通常管理者, 3)各 Component Computer の OS の正常動作のための管理を行なう Component 管理者, 4)全体を統括する *Apostle* システム管理者の四段階に分割する. 管理作業のため, 2) は管理範囲にある利用者の個人ファイルと管理上関係するファイル, 3) は system file, 4) は全てのファイルに対して特別に変更権が認められる. この管理権の分散によって, 各管理作業が明確になり, かつ個々の管理者の負担は減少する.

UI は UNIX の shell をフロントエンドとして実現される. また, アクセス権の制御のためにファイル操作に関する UNIX の system call が変更される. 利用者からのコマンドは shell によって解釈され, まず, その利用者がコマンドを実行可能か否かが判断され, 可能な場合にのみ実行が行なわれる. ファイル操作に関しての system call 内では利用者のファイルに対してのアクセス権がチェックされる. この際, 利用者についての情報を引き出すために IS の機能を利用する. また, アクセス対象が個人用のファイルであり, 他 Component Computer 上に存在する場合, そのファイルをアクセスするために File Cacher (FC) の機能を利用する.

4 File Cacher (FC)

利用者個人に対するファイルやグループに対するファイルは三キャンパス内の一つの Component Computer 上にのみ置かれる. したがって, 物理的に他の Component Computer にログインしている場合には, 個人用のファイルにアクセスするためにキャンパス間の通信が必要となる. この通信を最小化させる目的で FC が設置される.

利用者が個人用ファイルをアクセスする場合, その一回目のアクセスでその全内容が利用者が利用している Component Computer 上の一時使用領域に複写される. 以後の同一ファイルに対するアクセスは全てこの複写されたファイルに対して行なわれ, キャンパス間の通信は行なわれない.

ファイルを変更する目的でアクセスが行なわれた場合, 元のファイルには他の変更を拒否するようにロックがかけられる. 但し, private file に対しては原則的にファイルの所有者以外のアクセスは認められていないので, private file に対してのロックは省略される. 変更のあったファイルは利用者のログアウトの時点で元のファイルに書き戻される.

References

[1] 井田昌之, 田中啓介, 'Apostle: 協調結合に基づく分散処理システム(1) -Trinity 構想における構成-,' 第 37 回情報処理学会全国大会, Sep. 1988.

[2] Sun Microsystems Inc. *'The Yellow Pages Protocol Specification,'* Feb. 1986.