「*Internet*広域分散協調サーチロボットインストール方法」

1 PRS・PRSMのプログラム仕様

PRS・PRSMの主要部分をJavaで実装した．PRSは完全なJavaアプリケーションである．PRSMはWWWサーバApacheにJServモジュールを組み込み，処理はJava servletsで行う．

なお，2000年2月末現在，JDKのバグ[1]により，$JDK1.2.1\_03$ SDK(Solaris Production Release)以降のJDK [2]でのみ動作することを確認している．また，動作確認済みのプラットフォームは，Solaris8/Sparc, Solaris2.6/x86, Solaris2.6/x86である．

1.1 PRSの動作

PRSは，まず起動時にPRSMにアクセスし，担当リストをPRSMから取得する．担当のサーバのページをすべて取得し終ると，再び担当リストを取得する．

ページ収集は時間を制限している．平日は午前2時～午前8時，土日は午前2時～午後10時のみページ収集を行う．

一日に一度，午前9時ごろ[3]，収集ログをPRSMに転送する．

初期設定では同時に200個のサーバに対してアクセスを行なう．1つのサーバに対しては20秒おきにアクセスを行なう．ただし，WWWサーバがHTTP/1.1のKeep Alive機能に対応している場合は可能な限り連続でアクセスを行なう．

WWWサーバの反応が速い場合には，アクセス間隔を20秒より短くする．PRSが「このWWWサーバは速い」と判断する基準は，PRSとWWWサーバとのHTTPプロトコルを用いた通信において，

(総転送量÷所要時間)

の値が，その担当するWWWサーバ内の上位20%以上かどうか[4]，である．すなわち，PRSが担当するWWWサーバ内で，そのWWWサーバとの間の通信速度が早い1/5のWWWサーバに対しては，WWWサーバの能力にまだ余裕があると判断する．

WWWサーバの能力にまだ余裕があると判断した場合，転送間隔をデフォルトの20秒から10秒に短縮する．

[1] JDK1.2には，同期回りの実装ミスでデッドロックするというバグが存在する．このバグはJDK 1.3で解決される予定．

[2] Reference Implementationでは動作しない．これは，Reference Implementationは，largefile awareでないいう本質的な問題もあり，Production ReleaseのJITの生成したコードが原因でプログラムが落ちるという致命的な問題があるため．

[3] PRSは起動後20分毎に現時間をチェックしており，午前9時を過ぎた時点(従って午前9時～午前9時20分の間)で本処理を実行する．

[4] 実験データからデータ転送速度14.0byte/ms以上が上位20%に相当することがわかったため，14.0byter/ms以上の転送速度になった場合に10秒に短縮する．

1.2 インプリメント上の注意点

上記のプロトコル仕様に基づき，PRS と PRSM を試作した．インプリメント上で，いくつかの動作上の制限があるので，以下に詳細を示す．

- 仕様では，PRS から PRSM へ収集ログを転送した後，PRSM が集計を行ない新規発見 WWW サーバ等を含めて担当サーバリストの更新を行なう．本来は，PRSM の機能として実装すべきであるが，現在は，別プログラムとして動作し，スクリプトにより連動する仕組みとなっている．

- プロトコル上は，担当サーバリスト・発見サーバリストはサーバのトップページだけでなく任意のページを入れることができる．この機能を使って，ディレクトリ単位での負荷分散も可能となっている．

- プロトコル上は，referer 情報 (そのページを発見することになったリンク元の URL) を含めることが可能である．しかし，情報交換によるトラフィック増加を考え，実装では「サーバのトップページのみ・referer なし」でやりとりしている．これにより，現在の PRS は，あるサーバのトップページからそのサーバ内のリンクだけで辿れないページを収集することができない．トップページだけでなく，すべてのページの情報を送受信した場合にどの程度のトラフィックが発生するかを検証する必要がある．

- PRS の動作環境は，今回はほぼすべて Solaris 2.6 Intel Edition であるが，このプラットフォーム用の Java Virtual Machine (JVM) で PRS のプログラムを動作させると非常に不安定である (ハングアップ，コアダンプなど)．この問題を回避するため，PRS のプログラムは毎日 0 時ごろに起動し，朝 9 時 (土日は夜 11 時) にプロセスを終了するように設計した．ハングアップなどでプロセスが動き続けているときは，0 時の起動時にプロセスを強制的に終了させる．

1.3 PRSM プログラムの利用方法

インストール

1. Java2 v1.2[5] が動作する Servlet 動作環境を用意する．

2. 作業用ディレクトリとプログラムディレクトリを作成しておく．作業用ディレクトリは，動作中に必要なデータなどを読み書きする場所なので，大きな容量が必要である．プログラムディレクトリは，配布パッケージが展開できればよい．以下の説明で，作業用ディレクトリを'/data/prsm'，とする．

3. 配布パッケージ prs-2000mmddn.zip もしくは prs-2000mmddn.tar.gz を，プログラムディレクトリで展開する．

4. プログラムディレクトリの org/hauN/kent/prsm/prsm.properties の'/data/prsm/'の部分を作業用ディレクトリに変更する．

[5] *JDK*1.2.1_03 以降

5. Servlet 動作環境において，パッケージを展開したプログラムディレクトリが CLASSPATH に含まれるように設定する．

6. Servlet 動作環境において，以下のクラスが指定の名前の Servlet として起動できるように設定する．

クラス名	Servlet 名
org.hauN.kent.prsm.IsRestart	/servlets/isrestart
org.hauN.kent.prsm.GetList	/servlets/list
org.hauN.kent.prsm.ListRobot	/servlets/listrobot
org.hauN.kent.prsm.RegisterServer	/servlets/record
org.hauN.kent.prsm.Restart	/servlets/restart
org.hauN.kent.prsm.RegisterLog	/servlets/submitlog
org.hauN.kent.prsm.Subscribe	/servlets/subscribe
org.hauN.kent.prsm.Unsubscribe	/servlets/unsubscribe

運用

1. 担当サーバの設定
   作業ディレクトリ下の'assign/' ディレクトリに，PRS の名前で担当サーバリストのファイルを置く．登録されている PRS の名前は，作業ディレクトリの robothost.list ファイルに羅列されている．担当サーバリストは，プロトコルの (M1) そのものである．ただし，'http://' で始まらない行があると，(M1) の送信時に自動的に'http://' を補完する．

2. ログ
   作業ディレクトリ下の'logs/' ディレクトリに，(M3) で送られた各 PRS の転送ログが格納される．ファイル名は，＜PRS 名＞.yyyymmddhhMM である (yyyymmddhhMM は転送された日付と時刻)．PRSM の動作のログが作業ディレクトリの'prsm.log' に記録される．このファイルは問題が発生したときの原因究明用に存在するので，必要なければ削除して構わない．

3. PRS の監視と制御
   'http://PRSM のマシン /servlets/listrobot' にアクセスすると，PRSM が把握している各 PRS の状況を表示する．具体的には，各 PRS について，

   ・リスタート要求を処理したかどうか
   ・最後にリスタートチェックをした時刻
   ・最後に担当リストを送った時刻

   を表示する．
   PRS が動いていれば PRS から PRSM にリスタートチェックが 1 時間に 1 回あるはずなので，リスタートチェックが長い時間なければ，この PRS は動作していないことがわかる．'http://PRSM のマシン /servlets/restart' では，

・次に送る担当サーバリストが50ページ限定なのかページ数無制限なのか
・各PRSをリスタートさせるかどうか

を設定することができる.

1.4 PRSプログラムの利用方法

インストール

1. Java2 v1.2[6] が動作する環境を用意する.

2. 作業用ディレクトリとプログラムディレクトリを作成しておく. 作業用ディレクトリは, 動作中に必要なデータなどを読み書きする場所なので, 大きな容量が必要である. プログラムディレクトリは, 配布パッケージが展開できればよい. 以下の説明で, 作業用ディレクトリを'/data/prs', プログラムディレクトリを'/data/class'とする.

3. 配布パッケージ prs-2000mmddn.zip もしくは prs-2000mmddn.tar.gz を, プログラムディレクトリで展開する.
'/data/class'以外のディレクトリに展開する場合は, 'robot.conf' および'prsinit.sh'を書き換える必要がある.

```
% mkdir -p /data/class
% cd /data/class
% /path/to/gzip -dc /path/to/prs-xxxx.tar.gz | tar xvf -
```

4. プログラムディレクトリにある'prs.conf'のファイル名を変更し, 編集する.
ファイル名を変更するのは, プログラムのバージョンアップの際に上書きされるのを防ぐためである. 以下の説明では'prs.conf'を'robot.conf'に変更したとする.

変更を最小に抑えるには, 必ず'/data/class'で配布パッケージを展開し, 作業ディレクトリを'/data/prs'にする.

```
% cp prs.conf robot.conf
% mkdir -p /data/prs/html
% vi robot.conf
```

して, prs.myfqdn (と, 必要なら prs.proxy) を設定
以下に, 'robot.conf'の各項目の意味を説明する.

[6] JDK1.2.1_03以降

項目名	説明
prs.myfqdn	ロボットプログラムを動かすホストのホスト名(ドメイン名込み). DNS の名前を持っていないときに限り，IP アドレスでも構わない.
prs.logfilename	ロボットプログラムの動作中のログを記録するファイル名.
prs.transferlogfilename	ロボットプログラムの動作中の転送記録を保持するファイル名.
prs.serverlistfilename	担当情報を記録しておくファイルのファイル名.
prs.datadirectory	ロボットで集めてきたファイルを格納するディレクトリ名. 大容量が必要.
prs.proxy	ロボットを動かすホストがファイアウォールの中にある場合や, キャッシュを通したい場合などに proxy.foo.co.jp:8080 のようにして proxy を指定する. proxy がいらない場合は，「prs.proxy=」とする.

他に，PRSM の各用途用の URL が記述されている．PRSM が既定値と異なる場合は，各 URL も変更する．

5. crontab を編集する．
   % crontab -e
   10 0 * * * /data/class/prsinit.sh start 2 > /dev/null
   と入力してエディタを終了する．

6. ロボットプログラムの準備ができたことを PRSM に通知する．
   % /usr/java1.2/bin/java -cp /data/class RobotSubscribe /data/class/robot.conf

運用

1. **ホストの電源を落としたいとき**

   ps コマンドで java のプロセスが動いていなかったら，そのまま shutdown して構わない．java のプロセスが動いているときは，プログラムディレクトリ('/data/class/') の prsinit.sh を
   "% /data/class/prsinit.sh stop"
   のように実行してから shutdown する．

   なお，ロボットは，毎日 cron により 0 時 10 分に起動される．また，平日は，午前 8 時まで収集，土日は午後 10 時まで収集し，その後ログが転送される．このため，平日は午前 11 時半～午前 0 時，土日は，午後 11 時半～午前 0 時であれば，java のプロセスは動作していないため，収集に影響を与えずにホストの電源を落すことができる．

2. **ロボットプログラムのバージョンアップ**

   上記の方法でプログラムを停止，新しい配布パッケージを，以前展開したディレクトリと同じところに展開して上書きする．また，自動化するには，本章で説明する自動バージョンアッププログラムを用いる．

3. ログファイル

作業ディレクトリにある‘robot.log’は，PRS動作の状況を記録したものである．基本的に不必要であるので，ファイルが邪魔になったら削除しても構わない．‘transfer.log’は転送の状態を記録したファイルで，これは定期的にPRSMに送られる．PRSMに送られると‘transfer.log.yyyymmddhhMM’という名前に変更されるので，名前が変更された後は削除して構わない．ただし，ネットワークの問題等によりPRSMへの転送が正常に終了しない場合のために1週間程度のログを残すのが望ましい．

2　分散エリア決定

2.1　利用方法

手順 1

まず，Perl が実行できる環境を用意する．以下の説明に登場するディレクトリは全て作業ディレクトリの下にあるものとする．

手順 2

'log/' ディレクトリには各PRS のログが格納されている．ただし容量が大きいため gzip で圧縮されている．log.pl を実行すると，これらのログをPRS ごとにまとめたファイル（名前はPRS 名と同じ）が作成される．このファイルも'logs/' ディレクトリに置かれる．ファイル形式は「URL ドキュメント量 収集時間」で，URL 順にソートされている．

手順 3

get_distance.pl を実行すると，上で得られたログファイルをもとにPRS から各サーバまでの距離を計算し，'distance/' ディレクトリにPRS の名前でファイルが出力される．ファイル形式は「サーバ名 距離」である．

手順 4

get_document.pl を実行すると，ログファイルをもとに各サーバのドキュメント量が作業ディレクトリの'document.txt' という名前のファイルに出力される．ファイル形式は「サーバ名 ドキュメント量」である．

手順 5

get_cost.pl を実行すると，'distance/' ディレクトリの距離ファイルと document.txt を用いて，PRS ごとに各サーバに対するコストを計算し，'cost/' ディレクトリにPRS の名前でファイルが出力される．ファイル形式は「サーバ名 コスト」である．

手順 6

make_random_list.pl はランダムにサーバを割り当てるために実行するもので，'random/' ディレクトリにPRS の名前で担当サーバリストのファイルが出力される．

手順 7

distribution.pl はコスト均等化アルゴリズムを実行するプログラムで，上で得られた各情報をもとに，コスト均等化を試みる．なお，プログラム上でアルゴリズムの実行回数を任意に設定することが可能である．実行結果は「PRS 名: コスト: 担当サーバ数」の形式で標準出力に出力される．

3 PRS 自動バージョンアップ方式

PRS 自動バージョンアッププログラムは，分散型 WWW ロボットのクライアント側，すなわち PRS のバージョンアップを自動化するものである．これによって，ソフトウェアのバージョンアップに伴う作業が大幅に短縮される．

3.1 プロトコル仕様

毎日一回[7]，HTTP プロトコルを使い[8]，指定された URL にアクセスし，そこに存在するプログラム (PRS の tar.gz ファイル) が現在ローカル (クライアント) に使っている PRS よりもバージョンが新しい時，自動的に取得し，プログラムを更新する．

本プログラム (`PRS_Update.java`) は，Java(`JDK1.2.1\_03` 以降) で動作する．

1. 設定ファイル 1(update.conf)
   update.conf には以下の内容を記述する．

   prs-update.source PRS のプログラムが格納してある URL を記述

   prs-update.id アクセスのための ID を記述

   prs-update.pass アクセスのためのパスワードを記述

   prs-update.gzip クライアントマシンの gzip のパスを記述

   例

   ```
   prs-update.proxy=proxy.dwr.com:8888
   prs-update.source=http://www.dwr.com/prs/
   prs-update.id=access_id
   prs-update.pass=passwd
   prs-update.gzip=/usr/local/bin/gzip
   ```

2. 起動方法

   ```
   /bin/sh prs-update.sh /data/class/ /data/class/update.conf
   ```

   - prs-update.sh は起動用スクリプトファイル
   - 第 1 引数は JAVA プログラムの CLASSPATH
   - 第 2 引数は 1の設定ファイル

3. 設定ファイル 2(update.date)
   現在クライアントにインストールされている PRS が，最新かどうかをチェックするために，2の第 2 引数で指定されているディレクトリに update.date というファイルを置き，その中で現在使われている

[7] cron により毎日起動する
[8] Proxy 経由も利用可能とする

prs- 西暦 (4 桁)/ 月 (2 桁)/ 日 (2 桁)/ バージョン (1 桁).tar.gz

のファイル名を格納しておく.

例: prs-200001011.tar.gz

4. ログファイル (`prs_update.log`)
   プログラム実行中の一連のログを記録するために, 2の第 2 引数で指定されているディレクトリに`prs_update.log`というファイルを作成し, そこにログを出力する.

5. 起動後の処理

   (a) update.conf から設定内容を読み込む.

   (b) (5a) で, proxy の記述が存在した場合には, proxy の設定をする. proxy サーバーにアクセスするデフォルトの port 番号は 80 である.

   (c) update.date から現在の PRS のバージョンを読み込む. update.date が存在しない場合には, (5d) の動作を行なわずに (5e) で最新のプログラムをダウンロードする.

   (d) prs-update.source に記述されている URL にアクセスし, 最新のバージョンのプログラム名を調べる. これが, (5c) で読み込んだバージョンより新しければ (5e) で, 最新のプログラムをダウンロードする. (5c) で読み込んだバージョンと, 最新のバージョンが一致したときは, (5e) 以下の作業を行なわずに, 処理を終了する.

   (e) prs-update.source に記述されている URL にアクセスし, 最新のバージョンのプログラムをダウンロードする.

   (f) (5e) でダウンロードしたプログラムを gzip で解凍, インストールする. ダウンロードした prs-xxxx.tar.gz ファイルはインストール後, 削除する.

   (g) update.date の内容をダウンロードしたプログラム名に更新する.

   (h) `prs_update.log` にログを書き出す.

   (i) 処理を終了する.

3.2 利用方法

以下の説明では,

$WORK 2.(9ページ) の第2引数で指定されているディレクトリ

$START prs-update.sh の置かれているディレクトリ

$CLASSPATH PRS_Update.class の置かれているディレクトリ

とする.

1. 手動で実行する場合
   以下のコマンドを実行する.

   ```
   /bin/sh $START/prs-update.sh $CLASSPATH $WORK/update.conf
   ```

   例

   ```
   $WORK=/data/class/
   $START=/data/
   $CLASSPATH=/data/class/
   ```
   の場合

   ```
   /bin/sh /data/prs-update.sh /data/class/ /data/class/update.conf
   ```

2. cron を使用して自動で実行する場合
   crontab に以下を記述する.
   毎日0時に起動する場合

   ```
   0 0 * * * /bin/sh $START/prs-update.sh $CLASSPATH $WORK/update.conf
   ```

4 PRS 収集データ再配布システム

再配布システムはPRS(以下では「WWW データ収集クライアント」と呼ぶ) からデータを一カ所に収集 (このプログラムを「データ収集システム」と呼ぶ) し，第三者に配布する (この配布するためのプログラムを「データ配布システム」，データを受け取るためのクライアントを「第三者クライアント」と呼ぶ).

4.1 プロトコル仕様

システム構成図を図1に示す.

本システムは，WWW データ収集クライアント (PRS) の特定のディレクトリにおかれたファイル (仕様は以下参照) をデータ収集システムに定期的に（1 日 1 回）集め，データ収集システムから第三者クライアントへデータを再配布する.

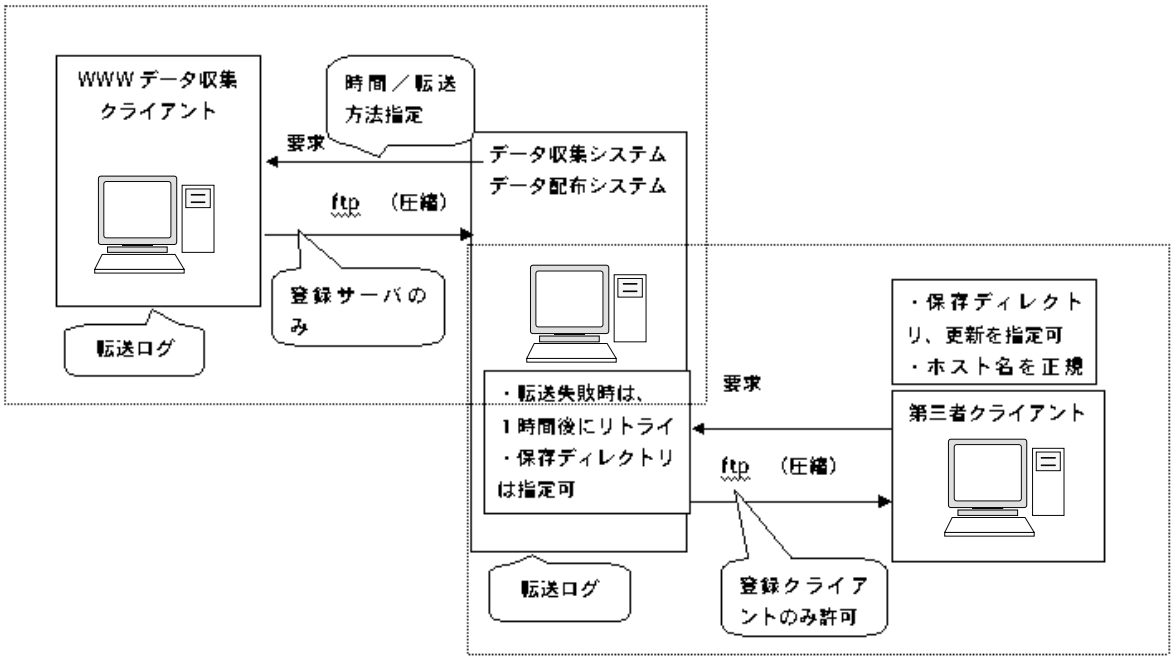

図 1: PRS 収集データ再配布システム

WWW データ収集クライアントは，収集した WWW データ (Web ページ) をデータ収集システムに配送する．WWW データ収集クライアントが Proxy 外 (インターネットに直結) の場合，データ収集システムは WWW データ収集クライアントに直接，接続することができる．データ収集システムからの要求時，WWW データ収集クライアントはデータ収集システムにデータを転送する.

一方，データ収集クライアントが Proxy 内 (ファイアウォール内) の場合，WWW データ収集クライアントはデータ収集システムからスケジュール設定ファイルを取得し，該当のデータをデータ収集システムに転送する.

データ収集システムはスケジュール設定ファイルを読み込んで WWW データ収集クライアントからデータを取得する．スケジュール設定ファイルには，どの WWW

データ収集クライアントに接続するか，収集するデータの条件(日付等)が入る．

データ配布システムは，要求によって，第三者クライアントにデータを配布する．第三者クライアントの環境ファイルに要求する情報が入れる．

第三者クライアントはデータ配布システムから，WWWデータを要求するエンドユーザである．要求はデータ配布システムに送信される．第三者クライアントのデータ管理機能は本システムの対象外である．

「WWWデータ収集クライアント」→「データ収集システム」

1. 「データ収集クライアント」から「データ収集システム」へは，「データ収集システム」からの要求に基づいてファイルを転送する．セキュリティ確保のため，あらかじめ「データ収集クライアント」側に登録してある「データ収集システム」にしかデータを送れない．

2. 転送時には，データをzipにより圧縮して転送する．

3. ログ（転送開始時間，終了時間，転送速度，転送ファイル容量，総数）を「データ収集クライアント」側に残す．ログは，１週間毎別ファイルとして保存する．

「データ収集システム」

1. 「データ収集クライアント」毎に，収集開始の要求を出せる．

2. 上記のスケジューリングをファイル(クライアント名，開始時間，転送ファイル条件)を指定可能．開始時間は，UNIXのcrontabの書式に準拠する．転送ファイル条件には，全転送，差分転送，特定日指定転送（特定日を指定した場合は，その特定日以降のデータのみを転送），指定期間転送（いつからいつまでを指定）をクライアント毎に指定可能．

3. 「データ収集システム」側が持っている転送ログと突き合わせ，差分転送であれば，差分のみの転送要求を「WWWデータ収集クライアント」に出す．この時，クライアント側には，XXXX/XX/XX～YYYY/YY/YYというように更新日の日付を送る．

4. 「WWWデータ収集クライアント」からのデータ転送ログ（クライアント名，転送開始時間，終了時間，転送ファイル容量，総数）を「データ収集システム」側に残す．ログは１週間毎別ファイルとして保存する．

5. 転送にfailした場合は，１時間後にリトライを行い，それでも失敗した場合には，上記でスケジューリングされる日に再トライする．

6. 転送データは，「データ収集システム」側の指定されたディレクトリに保管される．

7. 再配布効率化のために，一日毎の差分ファイルは１週間分保存する．

「データ配布システム」→「第三者クライアント」

1. 第三者クライアントからの要求に基づいてデータを転送する.
2. 転送時には，データを zip で圧縮して転送する.
3. 認証を行い「データ配布システム」側へ登録してある「第三者クライアント」のみにしか転送できない.

「第三者クライアント」→「データ配布システム」

1. 転送要求は，(ホスト名, 更新日, 指定期間) の指定が可能．また，ホスト名は正規表現で記述可能.
2. 「第三者クライアント」側では，取得データを格納するディレクトリを指定できる．指定ディレクトリ以降の構造は，以下のディレクトリ構成と同様であり，hostname:port 番号.zip ファイルで保存.

以下に，WWW データ収集クライアントのデータ格納ディレクトリの仕様を示す.

WWW データ収集クライアントのデータ格納ディレクトリ

```
html/[sumh]/[suml]/[servername:port]/
```

[sumb] [suml] は，servername:port に対して

```
static String createDataDirectory(String base, String server) {
    int sum = 0;
    for (int i = 0;  i < server.length();  i ++) {
        sum = sum*13+server.charAt(i);
    }
    String sumh = Integer.toString(sum & 0xff00, 16);
    String suml = Integer.toString(sum & 0x00ff, 16);
```

という計算をした結果.

ディレクトリ内のファイル構成

1. **extserver.list**
   このサーバから外のサーバへのリンク情報.
   1 行が「[リンク先 URL] [TAB] [リンク元 URL]」になっている.

2. **transferedurl.list**
   このサーバ内の，転送したURLのリスト．1行に1エントリ．サーバ名は書かない．http://www.foo.jp/ だったら，「/」と書く．

3. **transferedurldate.list**
   transferedurl.list の1エントリが 24 Byte に対応する．

   最初の 8Byte: 転送した日付 (1970年1月1日00:00 GMT からの経過ミリ秒)

   次の 8Byte: データの最終更新日 (情報がなければ0) (1970年1月1日00:00 GMT からの経過ミリ秒)

   次の 8Byte: データサイズ [Byte]

4.2 利用方法

インストール

1. **WWW データ収集クライアントのインストール**

   Redistribution.zip をハードデイスクの任意デイレクトリに解凍し，

   ```
   sh makeFileDataCollectionClient.sh
   ```

   を実行する．

2. **データ収集システムとデータ配布システムのインストール**

   Redistribution.zip をハードデイスクの任意デイレクトリに解凍し，

   ```
   sh makeFileaCollectDistributeServer.sh
   ```

   を実行する．

3. **第三者クライアントのインストール**

   Redistribution.zip をハードデイスクの任意デイレクトリに解凍し，

   ```
   sh makeFileThirdParty.sh
   ```

   を実行する．

システムの起動

1. **WWW データ収集クライアントの起動**

WWW データ収集クライアント (Proxy なし) を起動するためには，下記のいずれかのコマンドを使用する.

```
1. java -DENV_FILE_PATH= 環境ファイルパス名
        Redistribution.prsClient.PRSClient
例:
java -DENV_FILE_PATH=(environment file path name)
        Redistribution.prsClient.PRSClient

2. java Redistribution.prsClient.PRSClient
```

WWW データ収集クライアント (Proxy あり) を起動するためには，下記のいずれかのコマンドを使用する.

```
1. java -Dhttp.proxyHost = IPAddress  -Dhttp.proxyPort=PortNo
        -DENV_FILE_PATH=(environment file path name)
        Redistribution.prsClient.PRSClient
例:
java -Dhttp.proxyHost = 10.7.5.121  -Dhttp.proxyPort=1080
     -DENV_FILE_PATH=/japan/datacollection/WWWCollectionClient.rc
     Redistribution.prsClient.PRSClient


2. java -Dhttp.proxyHost = IPAddress  -Dhttp.proxyPort=PortNo
        Redistribution.prsClient.PRSClient
例:
java -Dhttp.proxyHost = 10.7.5.121  -Dhttp.proxyPort=1080
     Redistribution.prsClient.PRSClient
```

”環境ファイルパス名”は絶対パス，または相対パスである (クラスがインストールされたデイレクトリに相対する). 2. で，環境ファイルパス名が指定されていない場合，ファイル名”WWWGatherClient.rc”(デフォルトの WWW データ収集クライアントの起動用環境ファイル) が必要.

2. **データ収集システムの起動**

下記のいずれかのコマンドを使用して，WWW データ収集システムを起動する.

1. java -DENV_FILE_PATH=環境ファイルパス名
   Redistribution.collectDistribute.DataCollectionServer

   例

   java -DENV_FILE_PATH
   JAPAN/myProjects/NTTSoft/CollectionServerFiles/CollServerEnv.list
   Redistribution.collectDistribute.DataCollectionServer

2. java Redistribution.collectDistribute.DataCollectionServer

“環境ファイルパス名“は絶対パス，あるいは相対パスである (クラスがインストールされたデイレクトリに相対する)．2. で環境ファイルパス名が指定されていない場合，ファイル名“WWWGatherServer.rc ”のファイルが必要となる．これはデータ収集システム起動用のデフォルト環境ファイルである．

3. **データ配布システムの起動**

下記のいずれかのコマンドでデータ配布システムを起動する．

1. java -DENV_FILE_PATH=environment file path name
   Redistribution. collectDistribute.DataDistributionServer

例

java -DENV_FILe_PATH
JAPAN/myProjects/NTTSoft/CollectionServerFiles/distributeEnv.list
Redistribution. collectDistribute.DataDistributionServer

2. java Redistribution. collectDistribute.DataDistributionServer

“環境ファイルパス名“は絶対パス，あるいは相対パスである (クラスがインストールされたデイレクトリに相対する)．2. で環境ファイルパス名は指定されていない場合，“WWWDeliveryServer.rc ”(デフォルトのデータ配布システム起動用の環境ファイル) が必要．

4. **第三者クライアント**

下記のいずれかのコマンドを使用して，第三者クライアント (Proxy なし) を起動する．

1. java -DENV_FILE_PATH=環境ファイルパス名
   Redistribution.thirdParty.ThirdPartyClient

例

```
java  -DENV_FILE_PATH
      JAPAN/myProjects/NTTSoft/CollectionServerFiles/distributeEnv.list
      Redistribution.thirdPartyClient ThirdPartyClient
```

```
2. java Redistribution.thirdParty.ThirdPartyClient
```

下記のいずれかのコマンドを使用して，第三者クライアント (Proxy で) を起動する.

```
1. java -Dhttp.proxyHost = IPAddress  -Dhttp.proxyPort=PortNo
        -DENV_FILE_PATH=(environment file path name)
        Redistribution.thirdParty.ThirdPartyClient.
```

例

```
java -Dhttp.proxyHost = 10.7.5.121  -Dhttp.proxyPort=1080
     -DENV_FILE_PATH=/japan/redistribution/WWWDeliveryClient.rc
      Redistribution.thirdParty.ThirdPartyClient.
```

```
2. java -Dhttp.proxyHost=IPAddress  -Dhttp.proxyPort=PortNo
        Redistribution.thirdParty.ThirdPartyClient
```

例:

```
java -Dhttp.proxyHost = 10.7.5.121  -Dhttp.proxyPort=1080
     Redistribution.thirdParty.ThirdPartyClient.
```

“環境ファイルパス名“は絶対パス，あるいは相対パスである (クラスがインストールされたデイレクトリに相対する). 2. で環境ファイルパス名が指定されていない場合，“WWWDeliveryClient.rc”(第三者クライアント起動用のデフォルトの環境ファイル) が必要.