回線切断に頑健なWebエージェントMiSpiderについて

A Robust Web Agent MiSpider against Disconnection

深萱裕二郎 † 大囿忠親 † 新谷虎松 †

Yujiro FUKAGAYA Tadachika OZONO Toramatsu SHINTANI

† 名古屋工業大学大学院 情報工学専攻

Graduate School of Engineering, Nagoya Institute of Technology

{fukagaya, ozono, tora}@ics.nitech.ac.jp

Web アプリケーションはクライアント PC へのインストール作業が不要であることから，様々なアプリケーションが Web アプリケーションとして再実装されている．Web アプリケーションは本質的にネットワーク依存しているにも関わらず，利用中の回線切断に対する対策は十分に研究されていなかった．本研究では，Web エージェント MiSpider を拡張することで，回線切断に頑健な Web アプリケーションを実現する．MiSpider は，Web ブラウザを実行基盤とするエージェントシステムである．MiSpider に，回線切断時におけるデータ保存機能を付加することで，実行中の Web アプリケーションにおける実行状態の保存を可能にした．本論文では，MiSpider における回線切断時のデータ保存機能について述べ，回線切断に頑健な Web アプリケーションの実装を示す．

1 はじめに

Web アプリケーションはクライアント PC へのインストール作業が不要であることから，様々なアプリケーションが Web アプリケーションとして再実装されている．Web アプリケーションは本質的にネットワーク依存しているため，データ入力中に回線が切断し，データが失われる，回線切断時に Web アプリケーションが利用不能になる等の恐れがある．

本研究では，Web エージェント MiSpider[1] を拡張することで，回線切断に頑健な Web アプリケーションを実現する．MiSpider は，Web ブラウザを実行基盤とするエージェントシステムである．MiSpider によって，サービスをエージェントしてカプセル化し，既存の Web ページ，Web アプリケーションに付加することが可能である．MiSpider に，回線切断時におけるデータ保存機能を付加することで，実行中の Web アプリケーションにおける実行状態の保存を可能にした．本論文では，MiSpider における回線切断時のデータ保存機能について述べ，回線切断に頑健な Web アプリケーションの実装を示す．

第 2 章では，Web エージェント MiSpider の概要について述べる．第 3 章では，MiSpider の回線切断時の対処機構について述べる．第 4 章では，MiSpider の実装方式について述べる．第 5 章では，MiSpider を利用した回線切断に頑健な Web エディタの試作について述べる．最後に第 6 章で本論文をまとめる．

2 Web エージェント MiSpider

MiSpider におけるエージェントは，Web ページ上で動作するページエージェント，及び Web サーバ上で動作するベースエージェントから構成される．MiSpider は，ページエージェントとベースエージェントが協調してサービスを提供する．ここでは，Web サーバと Web ブラウザの通信，資源に関連して次のような制約がある．1) ページ上のエージェントとサーバ上のエージェントの通信には制約がある．2) ページ上のエージェントは計算資源が限られている．3) サーバ上のエージェントは計算資源はあるがユーザと直接インタラクションできない．

2.1 実装言語

Web ブラウザ側の処理の記述には，JavaScript 言語を用いている．JavaScript 言語は，Web ブラウザ上において利用可能な最も一般的なスクリプト言語である．最近では，JavaScript の非同期通信を利用した技術に Ajax という名前がつけられたことにより，高機能な Web アプリケーションの開発言語の一つとして注目されている．JavaScript を利用することで，ユーザとのインタラクションの高いシステムが実現できる利点がある．

ここでは，MiSpider に関連が深い Ajax について説明する．Ajax とは，Web ブラウザに実装されている JavaScript の HTTP 通信機能を使い，Web

ページのリロードを伴わずにサーバと XML 形式のデータのやり取りを行い，処理を進めていく対話型 Web アプリケーションの実装形態である．実際には，Ajax という技術が存在しているわけではなく，DHTML(JavaScript + CSS + DOM) や XMLHttpRequest などのクライアントの技術にサーバ側の Web アプリケーションとを加えた実装形態のことを指す．Ajax では，指定した URL から XML ドキュメントを読み込む機能を使い，ユーザの操作や画面描画などと平行してサーバと非同期に通信を行うことで，サーバの存在を感じさせないシームレスな Web アプリケーションを実現する．

Ajax と MiSpider を比較すると，アプローチとして，JavaScript，XML，DOM，XMLHttpRequest などの技術を組み合わせて利用する点が共通点である．MiSpider と Ajax の差異は，MiSpider は，サービスを Web ブラウザ上にプッシュし，そのサービスの継続的な実行を実現する点である．Ajax ではデータの送受信を対象としており，MiSpider ではデータだけでなくサービスの送受信を対象にしている．

2.2 システムの構成

図 1 は，MiSpider におけるシステムの構成を表す．上記の制約を考慮して，ページエージェント，及びベースエージェントは次の役割を担当する．ページエージェントには，ユーザとリアルタイムにインタラクションを行えるという利点がある．例えば，ベースエージェントは，Web ページ内の情報を表示中に変更したり，表示されている情報の表示時間を計測することができない．ネットワークの遅延により，ベースエージェントはユーザとのリアルタイムのインタラクションに関して不利である．以上の特徴から，ページエージェントは，ユーザの行動の取得などのセンサとしての役割，及びユーザの行動に基づいた Web ブラウザの表示内容の追加，変更を行うアクチュエータとしての役割を果たす．ベースエージェントは，ページエージェントのタスク実行をサーバ側から支援する．ベースエージェントは，サーバ上で動作しているので，計算機資源に関する制約がページエージェントよりも緩い．例えば，CPU の実行速度，ネットワーク通信帯域，または 2 次記憶装置などに関して，ページエージェントよりも圧倒的に豊富な資源を用いて処理を実行可能である．ベースエージェントは，複数のページエージェントと連携するが，あるページエージェントと連携するベースエージェントは一意に定まる．

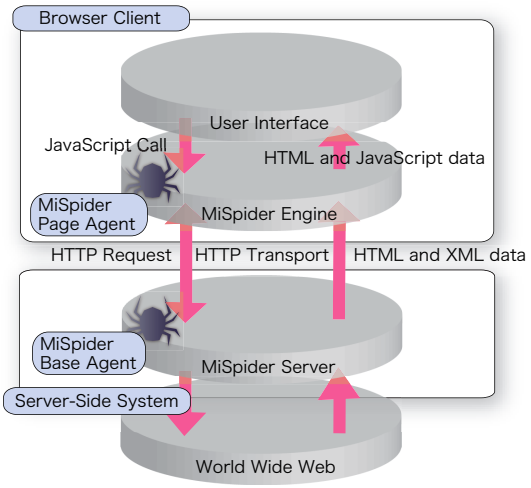

図 1: システム構成図

2.3 ページエージェントの定義

ページエージェントの定義は，ページエージェント内の 4 つのモジュールを定義することで実現される．すなわち，センサ定義，アクチュエータ定義，イベント定義，そして推論定義から構成される．センサ定義とアクチュエータ定義は，ページエージェントの低レベルの処理を定義する．センサ定義は，ブラウザ上の情報の変化やユーザからのインタラクションをイベントに変換する．アクチュエータ定義は，高レベルの処理を低レベルに変換し，ページエージェントの外界に対して変化を与える手続きを定義する．イベント定義は，マウスクリックのような低レベルのイベントを，ユーザからの情報要求のようなより高レベルなイベントに変換する手続きを定義する．推論定義では，ページエージェントの推論機構を決定するための定義をする．

3 回線切断に頑健な Web エージェント

Web アプリケーションは，ネットワークの不具合により，ローカル上のアプリケーションに比べて可用性，信頼性が低下する．以上の問題を解決するために，MiSpider の機能として，前章で述べた Web エージェントの機能に加え，Web アプリケーション利用中の回線切断に対処するための機能を持たせる．本章では，回線切断に対して頑健な Web エージェントを実現するための機構について述べる．

回線切断時にはサーバ側の資源を利用できないため，ローカル上の資源を利用する必要がある．しか

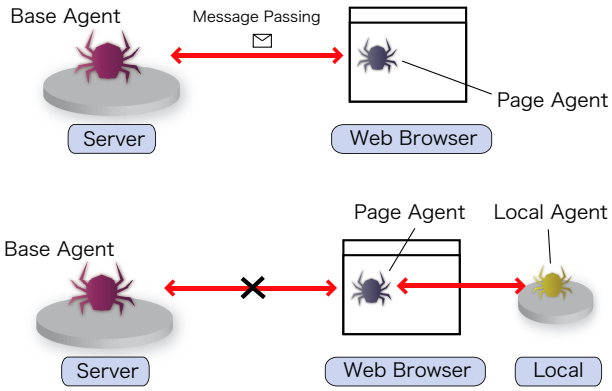

図 2: 回線接続時の通信 (上図) 及び回線切断時の通信 (下図)

し，セキュリティ上の問題から，Web ブラウザから自由にローカル上の資源にアクセスできない．Web ブラウザからのアクセスが許可されている，ローカル上の限られた資源を有効に利用することを考える．

そこで，ローカル上の資源とページエージェントの間の橋渡しを行う役割を持つローカルエージェントを定義する．ローカルエージェントはドメインごとに存在しており，Web ブラウザからアクセス可能なローカル上の領域のデータ管理を行う．ローカルエージェントはデータをインデックス管理している．ページエージェントが指定したキーに対応したデータにアクセスし，データの書き換え，参照を行う．

回線切断に対処するために，ページエージェントは，回線接続状況を確認し，回線接続状況に応じてページエージェントが動作する必要がある．また，ベースエージェント，ページエージェント，ローカルエージェントが連携して動作する必要である．

3.1 エージェント間の連携

図２は，回線接続時のエージェント間の通信，及び回線切断時のエージェント間の通信を示す．ページエージェントは回線切断に対応するため以下の手順で動作する．ページエージェントはサーバ上のベースエージェントと通信を試みる．ページエージェントは，ベースエージェントとの通信に成功した場合は，ベースエージェントとデータのやりとりを行う (図２の上図)．ベースエージェントとの通信に失敗した場合は，回線が切断されたと判断し，ローカルエージェントとデータのやりとりを行う (図２の下図)．回線切断中にローカルエージェントが得たデータは，回線切断状態から回線接続状態に遷移した時に，ページエージェントは，ベースエージェント及び，ローカルエージェントの持つ情報の更新時間を問い合わせ，いずれのエージェントの持つ情報が最新であるかを判断する．ローカルエージェントの持つ情報が最新であれば，ページエージェントを介してローカルエージェントからベースエージェントへデータを受け渡される．

3.2 回線切断時の起動

通常，Web アプリケーションは回線切断時に利用できない．そのため，回線切断時はローカルのアプリケーションを代用することになる．しかし，Web アプリケーションとローカルのアプリケーションで環境が異なる，回線切断時に行った作業を回線接続後に Web アプリケーションに反映する作業の手間，バージョン管理の煩雑さなどの問題が生じる．Web アプリケーションを回線切断時でも利用できると有用である．

回線切断時は Web アプリケーションを設置した URL へのアクセスによって，Web アプリケーションを利用できない．回線切断時にはサーバ側の資源を利用できないため，ローカル側に Web アプリケーションを起動するための何らかの資源が必要となる．ローカル側への資源の設置に関して，サーバからダウンロードしてインストールする手段ではユーザの負担が大きい．ブラウザの持つ保存機能を用いて保存したファイルから Web アプリケーションの起動を可能にする．

Web アプリケーションを起動した時に，ページエージェントを復元される仕組みを実現することで，回線接続時に起動する場合と同等の機能を Web アプリケーションで利用できるようにする．また同一のベースエージェントと通信を行うことで，同一のサーバ上の資源を利用する．ローカルエージェントは性質上，ドメイン別に存在し，ページエージェントは現在のドメインに対応するローカルエージェントのみと通信できる．サーバ上の Web アプリケーションを利用する場合と，ローカル上で起動する場合ではドメインが異なるため，異なるローカルエージェントと通信することになる．サーバ上の Web アプリケーションを利用して，回線切断時にローカルエージェントが得たデータは，ローカル上で起動した場合のローカルエージェントからは利用できない．またローカルエージェント同士の間で通信を行うことはできない．

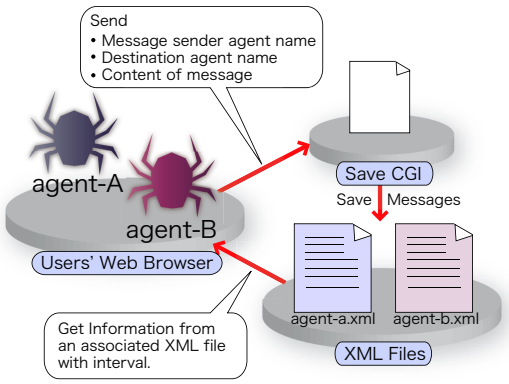

図 3: MiSpider におけるメッセージパッシング

4 実装方式

4.1 メッセージパッシング

Web ブラウザ上では，セキュリティの向上を目的として，さまざまな制約が課せられている．通信に関しても制約されており，自由な通信をすることができない．ある程度，自由な通信を実現する方法として，フォームを利用可能であるが，フォームを用いた場合，ページが変わる，すなわち，現在表示中のページが破棄されてしまい，エージェントが存続できない．また，ユーザインターフェイスの観点からも，エージェントの通信毎にページが読み変わることも好ましくない．ページの更新を伴わない，エージェント間通信の実現が必要である．

図 3 にメッセージパッシングの仕組みを示す．agent-A が agent-B へメッセージを送る場合，agent-A は CGI に agent-B の名前とメッセージ内容を送信する．そして，CGI では，送信先のエージェントの名前のファイル（agent-B.xml）に XML 形式で保存する．各エージェントは，対応する名前の XML ファイルにアクセスして，定期的に情報を更新する．このときに，agent-B は agent-A からのメッセージを取得する．

4.2 永続性

Web ページを閉じた時や，検索サイトなどでフォームを送信した後も，Web ページが変わる前の状態で，ページエージェントが継続的に動作することが可能になれば，より効果的な支援が行える．本研究では，このような性質を，エージェントの永続性と呼ぶ．新たに Web ページが読まれた時に，エージェントの状態を再構築するためには，Web ページが閉じられる，またはフォームを送信する直前のページエージェントの状態を保存し，関連のページが新たに開かれた時に，その状態を再構築するための仕組みが必要である．

MiSpider では，プロキシを通して Web ページにアクセスする．プロキシでは，JavaScript の埋め込みやリンク先の書き換えなどを行う．埋め込まれた JavaScript ファイルには，ページ読み込み時の処理と，閲覧ページ移動時の処理を記述する．そのための手続きは次の通りである．Web ページを読み込む時に，エージェントの内部状態を記述した XML ファイルを同時に読み込む．取得した XML 文書を構文解析して，JavaScript オブジェクト *obj* に変換する．*obj* から得られた実行状態をエージェントの実行状態として再設定した後に，エージェントの処理を再開する．ページを閉じる時には，*obj* を XML 文書に変換し，内部状態を XML ファイルとして保存する．エージェントの内部状態の表現に XML を用いることで，JavaScript のオブジェクトや配列などのデータ構造を表現できる．

4.3 ローカル領域

回線切断時に利用するローカル領域は，各々の Web ブラウザから利用可能な領域を活用する．例として，Internet Explorer では，DHTML behavior の userData 機能 [5] がサポートされており，Cookie と同じようにクライアントの PC にデータをセーブすることができる．1 ページあたり 64KB，1 ドメインあたり 640KB の情報を保存することができるため，Cookie と比べて保存できるデータ量が大きい．利用できるデータ量はできるだけ多い方がよいため，userData 機能を利用している．

4.4 環境とのインタラクション

ページエージェントのセンサとアクチュエータは，ページエージェントの外界，すなわち，Web ページとユーザからのインタラクションを扱う．ページエージェントは，HTML 文章をテキスト，もしくは Document Object Model(DOM) におけるオブジェクトとしても操作可能である．DOM は，W3C により標準化された HTML 文章のオブジェクトとしての表現である．ユーザからのインタラクションは，Web ページの変化として処理することも可能であるが，文字入力やマウスの移動，クリックなどのより低レベル

表 1: 回線切断状況に応じたセーブ，ロード

	セーブ	ロード
回線接続中	サーバ側	タイムスタンプにより決定
回線切断中	ローカル上	ローカル上

な処理としても処理できる．

5 応用アプリケーション

本章では MiSpider を利用した回線切断に頑健な Web エディタについて述べる．Web エディタは Web ブラウザ上で動作するエディタである．Web エディタでは入力したデータが失われないことが非常に重要となる．MiSpider の回線切断対処機構によって，回線接続状態から回線切断状態に移行した場合においてもユーザの入力データが失われず，継続的にシステムを利用できるようにする．また，回線接続状況をユーザが意識しなくても利用できるようにする．本システムにおいて，ページエージェントは記事編集機能，及び回線接続状態に応じた入力データの管理機能を保持している．回線接続状態に応じたデータのセーブ，ロードを表 1 に示す．回線接続中にデータのセーブを行った場合，ページエージェントはベースエージェントへデータを送信し，サーバ側にデータが保存される．回線切断後にデータのセーブを行った場合，ページエージェントはローカルエージェントと通信を行い，ローカル上にデータが保存される．データの保存時にタイムスタンプをつける．回線切断後にロードを行った場合，ページエージェントはローカルエージェントからデータを受け取る形式で，ローカル上からデータがロードされる．回線切断状態から再度接続した後にロードを行った場合，ローカル上のデータ，またはサーバ側のデータを比較して，最新のデータをロードする必要がある．ページエージェントは，ベースエージェント及び，ローカルエージェントにタイムスタンプを問い合わせ，最新の方からデータをロードする．

回線が切断された状態からシステムを起動するには，Web エディタを開いた状態で，ページの保存 (Web ブラウザの機能) を行い，保存したファイルを開くことで，ページエージェントが復帰し，Web エディタを提供する URL に接続した場合と同一の Web エディタを起動することができる．回線切断時にシステムを起動できることで，回線切断時であっても回線接続時と同様の環境で作業することができる利点がある．また，回線接続状態に切り替わった時にシームレスに回線接続時の動作へ移行できれば，ユーザに負担をかけない．回線切断状態から起動したシステムは，回線接続後はページエージェントがサーバ上のベースエージェントと通信が可能になり，回線接続状態で Web アプリケーションが設置された URL へアクセスして起動したシステムと同一のデータを参照することができる．以上の機能により，回線切断時にローカル上で編集したデータを，回線接続後にユーザは意識せずにサーバ側へ反映することができる．

6 まとめ

本論文では，回線切断に頑健な Web エージェントの実現に関して，必要となる機能について議論し，その実現方法を提案した．回線切断への対策として，Web アプリケーションを利用中に回線接続した状態から回線切断状態になった場合，及び，もとより回線が切断された状態から Web アプリケーションを利用する場合の 2 つの場合を考え，各々に応じた対策について述べた．MiSpider が提供する回線切断時の対処機能によって，信頼性の高い Web アプリケーションの開発が可能となる．ユーザの入力を多く必要とする Web アプリケーション，及び Web ベースの試験や書類提出といった信頼性が求められるケースに本研究は有用であると考える．

参考文献

[1] Y. Fukagaya, T. Ozono, T. Ito and T. Shintani, ”MiSpider: A Continuous Agent on Web Pages ,” In the Proc. of the 14th International World Wide Web Conference(WWW2005), pp.1008-1009, May. 2005.

[2] SAHAI A. et MORIN C., ”Mobile Agents Enhanced Thin Client Approach to Network Management, ” In Proc. of the IEEE Singapore International Conference on Networks (SICON’98), Kent Ridge, Singapore, June 1998.

[3] A. Vahdat, P. Eastham, ”WebOS: Operating system services for wide area applications, ” Technical Report UCB CSD-97-938, U.C. Berkeley, 1997.

[4] Ajax: a new approach to web applications http://www.adaptivepath.com/publications/essays/archives/000385.php

[5] userData Behavior http://msdn.microsoft.com/workshop/author/behaviors/reference/behaviors/userdata.asp