M-64

組込ブラウザにおけるプラグイン競合実行制御*

山中 弘　マット モラン　泊 陽一郎　齋藤 正史†
三菱電機株式会社　情報技術総合研究所§

1. はじめに

パソコン向けのブラウザでは、新しい形式のデータ処理をユーザが必要に応じて付け加えることのできるプラグイン機構を提供している。このプラグイン機構により、ユーザはブラウザ自体を変更することなく、新しいデータサービスを随時受けることが可能となる。

昨今、携帯電話を中心とする組込端末を用いたインターネットアクセスサービスが普及してきているが、現状の組込端末向けのブラウザでは、端末の制約上、プラグイン機構を実現することが難しい。プラグイン機構を実現できないことは、現在の組込ブラウザの問題点である「表現力の乏しさ」を引き起こす要因の一つと考えられる。

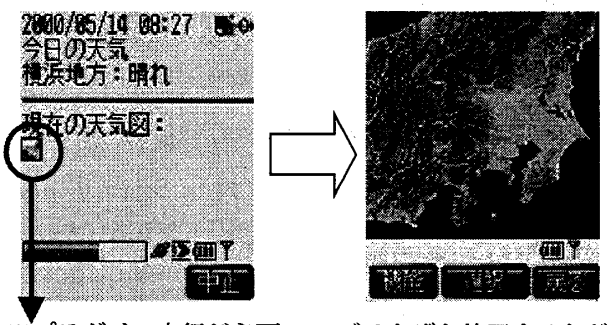
我々は、組込端末向けのブラウザにおいてプラグイン機構を実現する上での問題点と解決策について検討した。本発表で、問題点の一つであるプラグイン競合実行の問題と解決策について述べる。

2. 組込端末でのプラグイン実現上の問題点

パソコン端末に比べ、組込端末ではCPU性能が低いこと、メモリが乏しいことなどの小リソース性から、以下のようなプラグイン実現上の問題点が生じる。

1) プラグインの保存
プラグインモジュールの保存領域が、パソコンと比較して小さく抑えられてしまう。このため、プラグインを単に追加していくだけでは、新規のプラグイン追加が不可能になりやすい(文献[1]参照)。

2) プラグインの競合可能性
新しいプラグインを追加して実行するときに、CPUに与える負荷や必要なメモリなどが分からない。このため、ブラウザや現在実行中のプラグインと、新規に実行するプラグインが競合実行可能かどうかが分からず、ブラウザ内のインライン表示が不可能である。新規プラグインを実行するとすれば、図1に示すようにブラウザを終了してから、別途プラグインを実行しなければならない。



※プラグイン実行が必要だが、必要なメモリやCPU負荷が不明

ブラウザを終了するなどして別画面でプラグインを実行

図1. ブラウザとプラグインの個別実行

3. プラグインモジュール側の実装

本発表では、問題点2)に焦点を当て解決策を述べる。プラグイン競合可能性に関する問題解決のため、全てのプラグインモジュール側に以下のインタフェースを設ける。

3.1. リソース決定状態の通知

映像データ処理プラグインを例に説明する。映像データの処理では、[再生状態]／[停止状態]などの動作状態が存在する。[再生状態]と[停止状態]を比較すると、[停止状態]ではCPU負荷がかからないのに対し、[再生状態]では多くのCPU負荷がかかる。また、[停止状態]でブランク表示を行う場合は、処理に要するメモリも必要ない。

*Race Control of Embedded Browser's Plug-in Execution
†Hiroshi Yamanaka, Matthew Moran, Yoichiro Tomari, Masashi Saito
§Information Technology R&D Center, Mitsubishi Electric Corporation

このように、プラグインモジュールの動作状態に応じて、メモリ／CPU 負荷などの必要リソースは変化する。必要リソースが変化する動作状態を「リソース決定状態」と呼び、プラグインモジュール側には自分が持つ「リソース決定状態」の一覧、および現在の「リソース決定状態」を通知するインタフェースを設ける。

3.2. プラグイン状態遷移情報の通知

プラグインモジュールが「リソース決定状態」を複数持つ場合は、各々の状態同士の間に遷移可能関係が存在する。プラグインモジュール側には、自分が持つ「リソース決定状態」の遷移可能関係情報を通知するインタフェースを設ける。

3.3. プラグイン必要リソースの算出

プラグインモジュール側には、個々の「リソース決定状態」それぞれについて、モジュール実行に必要なメモリ／CPU 負荷などのリソース情報を計算し通知するインタフェースを実装する。

3.4. リソース決定状態変化の通知

プラグインモジュール側には、モジュールの実行中に自発的に「リソース決定状態を変化」させる動作を行った場合は、これをブラウザに通知するインタフェースを設ける。

3.5. プラグイン実行状態の変更

プラグインモジュール側には、遷移可能な「リソース決定状態」間の状態変化を引き起こすインタフェースを設ける。

4. ブラウザ側の実装

プラグインの呼び出しを行うブラウザ側では、プラグイン側に設けたインタフェースを用い、インライン競合実行判定・制御を行う。競合実行判定・制御の基本ロジックを図 2 に示す。

図 2 において、呼び出すプラグイン集合を初期設定したり、追加するプラグインを選択する処理が存在するが、ここで選ばれるプラグインの優先順位は以下のように決定される。

・表示／非表示関係
ブラウザの表示位置に応じて、現在表示されているプラグインを優先実行する。

・ユーザ設定
ユーザにプラグインごとの起動優先順位を指定してもらう。

・優先順情報のダウンロード
インターネット上にプラグイン毎の起動優先順位情報を配置し、ブラウザ側からアクセスする。

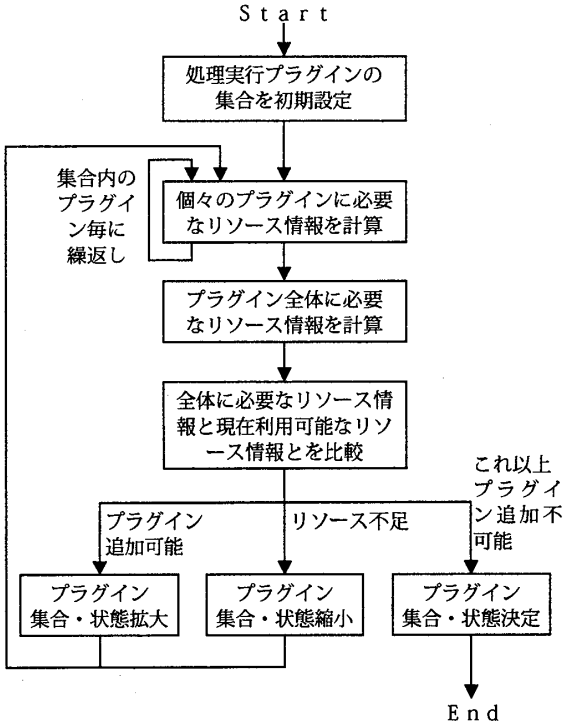


図 2. 呼び出しプラグイン決定基本ロジック

5. 効果

以上のプラグイン競合実行機構を設けることにより、次の効果を得ることができる。

・インライン実行の実現
新規に追加したプラグインをブラウザ内にインライン実行可能となり、より表現力に優れた画面表示を行うことができる。

・プラグイン同士の動的リソース配分
多数の画像を含むコンテンツ表示において、表示位置に応じた画像展開・展開解除が実現されるなど、状況に応じた最適なリソース配分が可能となる。

参考文献

[1] モラン,山中,齋藤,"組込みシステムにおけるプラグイン自動再構成",FIT2002(2002/9)