様式 C-19

科学研究費補助金研究成果報告書

平成 23 年　6 月　3 日現在

研究種目：基盤研究（C）
研究期間：2008～2010
課題番号：20500028
研究課題名（和文）　web アプリケーションにおける処理分担の半自動決定手法

研究課題名（英文）　Semi-automatic Client/Server Partitioning of Web Applications

研究代表者
小宮　常康（KOMIYA TSUNEYASU）
電気通信大学・大学院情報システム学研究科・准教授
研究者番号：80283638

研究成果の概要（和文）：Web アプリケーションプログラムが実行する処理のうち，サーバ／クライアントのどちらでも実行可能な処理群の適切な処理分担を半自動的にあらかじめ何通りか求めておき，実行環境に最も合う処理分担を動的に選択することで Web アプリケーションの高効率実行を達成する手法を提案し，そのための言語等を設計／実装した．また，処理分担の動的変更に関連する技術等の研究を行った．

研究成果の概要（英文）：In this research, we proposed a method of semi-automatic client/server partitioning of Web applications. To achieve this, we also designed and implemented a programming language for that. In our language, a programmer annotates parts of a program that are able to be executed on either side. Partitioning these parts between the client and server is performed semi-automatically in advance of the execution of the program. Our system can adapt to changes in the load of a Web environment by dynamically choosing an appropriate partition pattern among several partition patterns that are prepared in advance of the execution.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
2008 年度	1,200,000	360,000	1,560,000
2009 年度	900,000	270,000	1,170,000
2010 年度	1,000,000	300,000	1,300,000
年度			
年度			
総　計	3,100,000	930,000	4,030,000

研究分野：総合領域
科研費の分科・細目：情報学・ソフトウエア
キーワード：ソフトウエア開発効率化，プログラミング言語，Web アプリケーション，負荷分散，継続

１．研究開始当初の背景

ワープロ，表計算ソフトウェアのようなデスクトップアプリケーションや更には OS さえも Web アプリケーションとして作成する事例が増えている．これは，Web 環境を事実上のプラットフォームと見なしているとも言える．このような Web 環境の捉え方は，計算機の新しい利用形態を生み出す可能性

を秘めている．こうした動きの技術的背景には，デスクトップアプリケーションに匹敵する操作性を実現する Ajax 技術の影響が大きい．

クライアントとサーバの処理分担という観点でみると，Ajax による Web アプリケーションでは，クライアント側でもデータ処理等のある程度大きい処理を実行することができる．ただし，そのような処理は，サーバ／クライアントのどちらでも実行可能な場合も多く，分担の仕方には自由度がある．そのため，高速な実行を達成するには，分担の仕方が肝要となる．

しかしながら，適切な分担を見出す過程では，サーバ／クライアント両側のプログラムの書き直しが繰り返し必要となり，極めて煩雑な作業を強いられる．また，適切な分担量は Web サーバと Web ブラウザ（クライアント）の性能やネットワークの通信性能に依存し，更にこれらは動的に変化するので，分担の仕方を完全に静的に決める方法には問題が残る．

２．研究の目的

Web アプリケーションにおけるサーバ／クライアントの処理分担を，適切かつ効果的に行うための手法を研究する．具体的には，サーバ／クライアント両側のプログラムを，一体化した１つのプログラムとして記述でき，サーバ／クライアントのどちらでも実行可能な処理はアノテーションによって明示できる言語を設計・実装する．そして，Web アプリケーション開発段階において，いくつかの想定される実行環境のそれぞれに適切な処理分担をプロファイリングによって半自動的に探る方法を研究する．また，計算機／ネットワーク性能の動的な変動に適応できるように，実行時モニタリングによって実行環境の状況を把握し，その時点で最も適切な処理分担を選択する仕組みを研究する．

３．研究の方法

(1) 研究の目的で述べた言語を，JavaScript 言語を拡張することにより実現する．まず，プロトタイプ言語仕様を設計し検討する．また，様々な実行環境において，適切な処理分担をプロファイリングによって半自動的に求める方法を設計する．

(2) (1) で設計した言語仕様の実装を行う．また，設計した機能を用いた多くのサンプルプログラムを作成し，仕様の評価・検討を行う．特にサーバ／クライアントのどちらでも実行可能な処理の指示方法と処理対象となるデータの配置について検討する．

(3) (2) で検討した結果を仕様に反映させて再設計／実装する．実際にその機能によるサンプルプログラムを記述・評価し，問題点を洗い出して仕様の改良を積み重ねる．

４．研究成果

(1) Web アプリケーションのための動的適応可能な処理分担機構の設計と実装：

処理分担の問題を低減するために，サーバ／クライアントのどちらでも実行可能な処理をプログラム中にアノテーションによって明示し，開発段階において，適切な処理分担を半自動的に探ることができるプログラミング言語／言語処理系を設計・実装した．こうして完成させたプログラムは，原則，静的に処理分担を行うが，計算機の負荷やネットワーク性能の変動に適応できるように，動的に処理分担の仕方を変える機構を導入した．本機能利用の流れを図１に示す．

処理分担の決定は，想定するいくつかの実行環境において実行時間等のプロファイリングを行い，その結果と関数のコールグラフ等から，最適な処理分担を自動的に求める．実行環境の表現は，計算機性能及びネットワーク性能をパラメータ化し，これを開発者が調整することで想定する実行環境を表すことにした．完成させた Web アプリケーションの実行中は，定期的に実行環境の性能（パラメータ化された各性能の数値）をモニタリングし，最も適切な分担を随時選択することができる．

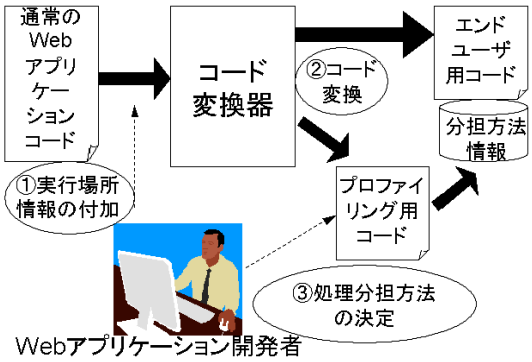

図 １本機能利用の流れ（5.主な発表論文等〔学会発表〕⑤より抜粋）

(2) サーバ・クライアント処理の動的分割・再配置機能を備えた Web アプリケーション用言語の設計：

研究成果 (1) によって，動的に処理分担を変更する機構を導入したものの，サーバ／クライアントのどちらでも実行可能な処理の対象となるデータをどのように扱うかという大きな問題が残った．データをサーバあるいはクライアントに固定して配置するのが望ましい場合もあれば，処理を実行する計算機のもとに配置するのが望ましい場合もあ

る．後者では，処理分担を動的に変更する際，データの配置も動的に変更しなければならない．しかし，そのようなデータ配置の指示及び動的な再配置コードの記述を全てアプリケーション開発者が行うのは負担である．一方，一般的なリモートポインタの類いによってこの問題を解決しようとすると，プログラムの記述性は高くなるが適切なデータ再配置・処理分担の見積りはむしろ難しくなる．あるいはシステムが必要以上に複雑になる．

そこで，次のような言語機能を設計した．まず，トップレベルで定義される変数や関数の配置・実行場所を，キーワード@server，@client，@migratableによって指示できるようにした．@server と@client は，それぞれサーバ，クライアントに固定して配置，@migratable は，配置の動的な変更が可能であることを示す．なお，マイグレータブルな関数／変数の実際の配置場所は個別に設定できる．マイグレータブルな関数／変数間の参照は全て暗黙のリモートポインタとして扱うことでマイグレータブルなオブジェクト群はあたかも全て同じ場所に存在するかのように見え，明示的な遠隔参照は不要である（場所の移動に対して透過）．暗黙のリモートポインタが，マイグレータブルな関数／変数にのみ現れる言語仕様としたことで，暗黙のリモートポインタ導入に伴うオーバヘッドが生じる範囲を限定できる．

Hilda という言語をベースに実行時の自動処理分割に対応できるプラットフォームの研究[Yang et al. WWW2007]がある．これに対して我々の研究は，JavaScript をベースにし，既存の Web アプリケーション開発になじみ易い利点がある．

今後は，本言語の実装を行い，(1)の研究成果と融合させる予定である．

(3)ページ遷移を考慮した Web アプリケーション記述言語の設計と実装：

本研究が対象とする Web アプリケーションは，基本的には Ajax スタイルとなるが，Web ページを遷移させるスタイルの方が適切な場面もなお多い．しかし，アプリケーション開発者の想定しないページ遷移が原因となってバグが発生することがありその発見は難しい．そこで，Ajax との親和性には更なる研究の余地はあるものの（研究成果(1)や(2)との融合方法の研究が必要），ページ遷移を行う堅牢なプログラムの開発を支援するプログラミング言語機能を設計・実装した．提案した機能は，「Web ページ」の定義及びページ遷移の制御構造である．開発者は，許可するページ遷移を簡潔にプログラムとして表記することができる．また，提案機能は，意図する遷移の把握に有益な遷移解析が静的に行えるように設計されている．それを実証するために遷移解析ツールも設計・実装した．

(4)継続の共有化による継続ベース Web サーバのメモリ使用量削減：

研究成果(3)では，継続ベース Web サーバと呼ばれる Web サーバを使用する．CGI プログラムでは，Web ブラウザ経由でユーザからの入力を得る際に，CGI プログラムを一旦終了させる必要があるため１つの CGI アプリケーションを複数の CGI プログラムに分割して構成する必要があった．継続ベース Web サーバは，Web ブラウザへ制御を渡す前に残りの計算（継続）を保存し，入力を得た後にその再開を行うことでこの問題を解決している．

しかし，継続ベース Web サーバはメモリを非効率的に使用する恐れがある．１つのサーバ側プログラムを同時に多数，実行した場合，内容のほとんどが同じ大量の継続が生成・保存されるからである．そこで，メモリ使用量の削減手法を研究した．提案手法は，まず，プログラムの制御フローを解析し，継続の保存地点までの実行経路を調べる．もし，実行経路が１通りであれば，この地点で保存される継続の（少なくとも）リターンアドレス等の制御情報は，いつも同じであるとわかる．そこで，継続の内容のうち，いつも同一となる部分を同プログラムを実行するプロセス／スレッド間で共有できる継続生成法を提案した．

複数のユーザがアドレス管理アプリケーションを利用する状況（各ユーザが３回の対話により名前とアドレスをサーバに登録し，２回の対話により名前から対応するアドレスを検索）をシミュレーションした際のヒープ使用量の遷移を図２に示す（セッション数は 50000）．黒線が従来法，赤線が提案法である．メモリ使用量が上下するのはごみ集めによるものであり，メモリ使用量が下がった時点の値にはごみが含まれない．その値を比較すると，従来法がおよそ 25,000KB に対して，提案法は 10,000KB の使用量となった．

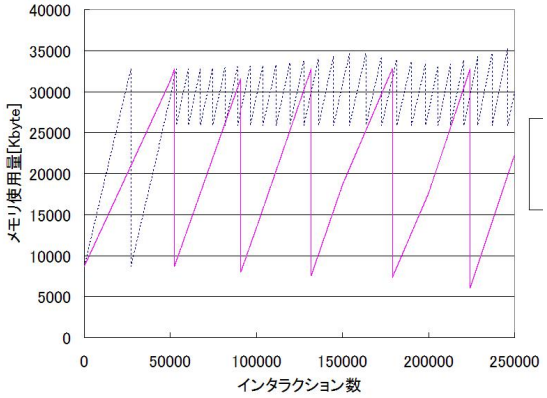

図 2 メモリ使用量の比較（5.主な発表論文等〔学会発表〕⑥より抜粋）

１つのプロセス内において，継続によるメモリ使用量を抑える研究は存在するが，プロセス／スレッド間でそれを実現する研究は新規のアイデアである．

(5)データの更新を漸次的に行う動的ソフトウェア更新機構：
Web アプリケーションの実行中に処理分担を効率よく変更する方法に結びつく研究として，実行中のプログラムの動的更新機構の研究を行った．提案手法は，更新に伴う処理の連続停止期間を短縮するために，更新処理を細分化し漸次的に更新処理を進める．
提案手法を実装し評価した結果，本機構の導入によるオーバヘッドは，非更新時はほぼゼロであった．更新対象データを 10^5 個含むプログラムの動的更新では，細分化された１回の更新処理による停止時間を 0.18ms におさえた場合で，細分化をしない方式（更新のために 839.7ms の連続停止）と比べて 8%程度のオーバヘッドとなった．

(6)スタックの実時間ごみ集め：
実行環境の変化に応じて処理分担を動的に変更する際，実行途中の関数が存在する場合はその実行状態（継続）も他方へ移送する機能が必要となる．その効率的な実現に向けて研究を行った．
継続を OS や CPU に依存せずに表現できる方式[Baker95]がある．この方式では制御スタックのごみ集めが必要となるが，ごみ集めによる中断時間の影響を小さくすることができる実時間化はされていなかった．本研究では，ヒープを対象とする従来の実時間ごみ集め手法を単純にスタック用として流用するのは不十分であること（細分化されたごみ集め処理の間隔が狭く制御できない問題）を指摘し，その問題の解決法を提案した．また提案法を実装し，評価を行った結果，細分化されたごみ集め処理の間隔が制御でき，なおかつオーバヘッドは極めて小さいことを示した．

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔学会発表〕（計７件）

①石橋　崇，サーバ・クライアント処理の動的分割・再配置機能を備えた Web アプリケーション用言語，情報処理学会第 83 回プログラミング研究発表会，2011 年４月 26 日，京都大学 楽友会館
②荻山温夫，データの更新を漸次的に行う動的ソフトウェア更新機構，情報処理学会第 73 回全国大会講演論文集，2011 年３月３日，東京工業大学 大岡山キャンパス
③三島　航，ページ遷移を考慮した Web アプリケーション記述言語の設計と実装，情報処理学会創立 50 周年記念（第 72 回）全国大会，2010 年３月 10 日，東京大学　本郷キャンパス
④石橋 崇，Web アプリケーションにおけるサーバ・クライアント処理の分割支援，第 12 回プログラミングおよびプログラミング言語ワークショップ（PPL2010），2010 年 3 月 3 日，琴参閣（香川県）
⑤山之井啓泰，Web アプリケーションのための動的適応可能な処理分担機構の設計と実装，第 71 回情報処理学会全国大会，2009 年 3 月 11 日，立命館大学 びわこ・くさつキャンパス
⑥新宮澄夫，継続の共有化による継続ベース Web サーバのメモリ使用量削減，第 71 回情報処理学会全国大会，2009 年 3 月 11 日，立命館大学 びわこ・くさつキャンパス
⑦ Tsuneyasu KOMIYA, Real-time Stack Garbage Collection, Workshop on Software Science and Technology in China and Japan (WSST/CJ 2008), 2008 年 9 月 16 日,Guilin, China

〔その他〕
電気通信大学のソフトウェア・リポジトリにおいて，開発したソフトウェアを公開することを検討している．

６．研究組織
(1)研究代表者

小宮　常康（KOMIYA TSUNEYASU）
電気通信大学・大学院情報システム学研究科・准教授
研究者番号：80283638

(2)研究分担者
なし

(3)連携研究者
なし