Vol.2016-IOT-32 No.41
2016/3/4

P2P構成に基づく耐障害性を備えたWebシステムの構築

中村 秀丸[1,a)]　田中 久治[1,b)]　堀 良彰[1,c)]　大谷 誠[1,d)]

概要：クライアントサーバモデルに基づく従来の Web システムでは，利用者がサーバから提供されるコンテンツを利用するのみであった．しかし，Web システムの利用者がコンテンツの提供と利用の両方を行う近年のトレンドにおいては，サーバに負荷が集中し障害が発生しやすくなる．そのため，障害や負荷へのより高度な対策が要求される．そこで，本研究では P2P 構成のメリットである耐障害性を備えた Web システム構築についての提案とプロトタイププログラムを作成した．

キーワード：Web システム，P2P，Web アプリ，クライアントサーバモデル，ファイル共有

Development of Web system with fault tolerance based on P2P architecture

Hidemaru Nakamura[1,a)]　Hisaharu Tanaka[1,b)]　Yoshiaki Hori[1,c)]　Makoto Otani[1,d)]

Abstract: In traditional Web system based on client server model, user is simply to use contents provided by server. However, server failures tends to occur by server loads are concentrated in recent trend that user of Web system has both providing and using contents. Therefore, more advanced measures for fault and load is required. In this research, we considered development of Web system with fault tolerance based on P2P architecture, and we have developed prototype program.

Keywords: Web system, P2P, Web application, Client/server architecture, File sharing

1. はじめに

スマートフォンやタブレット，PC 等のデジタルデバイスとインターネット利用環境が普及したことによって，利用者自らコンテンツを作成し提供するといった機会が増加している．今日では YouTube[1] のような動画共有サービスをはじめとした Web サービスで，そのようなコンテンツの利用と提供が盛んに行われている．このような Web サービスはクライアントサーバモデルに基づいた Web システム上で動作しているが，クライアントサーバモデルではサーバと呼ばれるコンピュータに負荷が集中しやすく，クライアント側の利用者がコンテンツの提供と利用の両方を行う近年のトレンドおいては，これまでのサーバが一方的にクライアントへサービスを提供するといった状況に比べて負荷が集中し障害が起きやすくなるため，障害や負荷へのより高度な対策が要求されている．

一方で，P2P 構成は全てのコンピュータ (ピア) がサーバやクライアントといった区別をせず，対等な立場で通信を行うという構成であり，クライアントサーバモデルと比べて負荷が各ピアに分散しやすく，障害に強いという特徴がある．そこで，本研究では P2P 構成に基づいた耐障害性を備える Web システムの構築を提案し，プロトタイププログラムを作成した．

[1] 佐賀大学
Saga University
[a)] hidemaru@ai.is.saga-u.ac.jp
[b)] kangaroo@ai.is.saga-u.ac.jp
[c)] horiyo@cc.saga-u.ac.jp
[d)] otani@cc.saga-u.ac.jp

2. クライアントサーバモデル

クライアントサーバモデルとはネットワーク上のコンピュータをクライアントとサーバに分離する構成のこと

である．中心となって処理を行うサーバと処理結果を受け取るクライアントにコンピュータの役割を分離し，処理をサーバへ集中させることでクライアントの処理能力が低くても高度な処理を必要とするサービスを提供できる他，サーバによるクライアントの一元管理がやりやすく，セキュリティを高めやすいというメリットもある．

2.1 クライアントサーバモデルの問題点

一方で従来の Web システムは以下のような問題を抱えている．

2.1.1 負荷集中による過負荷

クライアントサーバモデルに基づいた Web システムではサーバに処理を集中させるため，アクセス過多等により負荷が集中すると過負荷を起こしてしまう．サーバが過負荷状態になるとシステムが不安定になったり利用できなくなってしまうことがある．

2.1.2 単一障害点

サーバが過負荷等によって停止してしまうとサーバに接続していた全てのクライアントがサーバの提供していたサービスを利用できなくなってしまう．クライアントサーバモデルに基づいた Web システムではこのようにサーバが単一障害点になってしまう．様々な対策も存在しているが，一般的にコストがかかるものが多い．

2.1.3 リソースの消費

クライアントサーバモデルに基づいた Web システムで利用者（クライアント）がコンテンツを提供する場合，コンテンツはサーバに保存される．即ちサーバの記録媒体というリソースが消費されることとなる．近年の利用者が動画のような大容量のコンテンツを配信するタイプの Web サービスでは，サーバに保存される情報量が多くなり，サーバのリソースの消費が激しくなってしまう．

3. Peer to Peer

Peer to Peer(P2P) とはコンピュータ（ピア）が対等な立場で通信をする構成（以下 P2P 構成）のことである [2]．P2P 構成ではピア同士が対等な立場で通信をするため，サーバのような中心となるコンピュータが存在しないという特徴がある．

3.1 P2P 構成によるメリット

P2P 構成ではサーバのような中心となるコンピュータが存在しないことによる以下のようなメリットがある．

3.1.1 負荷分散

P2P 構成では中心となるサーバが存在しないため，負荷のかかる処理を複数のピアに分散させることができる．

3.1.2 耐障害性

P2P 構成では複数のピアが対等の立場で通信を行っているため，例えネットワークの内でいずれかのピアが障害を起こしたとしても，他のピアが代わりに引き続き処理を行うことができる．

3.1.3 リソース消費の分散

P2P 構成では動画等のコンテンツを特定のピアが全て持っている必要はない．そのため，大容量のコンテンツを配信するような場合においてもリソースの消費はそれを必要とするピアに分散されることとなる．

3.2 P2P ファイル共有

P2P ファイル共有とはピアにファイルのアップロード機能とダウンロード機能を持たせ，P2P 構成によるファイル共有を行うことである．P2P ファイル共有を行うソフトウェアを P2P ファイル共有ソフト（サーバントやクライアントと表現されることもある）と呼ぶ．代表的なものに Napster[3], Gnutella[4], Winny[5], BitTorrent[6] 等がある．本システムでは Web コンテンツの提供に，この P2P ファイル共有の機能を使用した．サーバントとしては BitTorrent クライアントである QuartzTorrent[7] を利用した．

4. P2PWeb

P2PWeb は本研究で構築した，P2P 構成に基づく耐障害性を備えた Web システムである．P2PWeb は P2P ファイル共有によって Web コンテンツを共有し，さらに取得した Web コンテンツを Web ブラウザから利用できるようにすることで P2P 構成に基づく耐障害性を備えた Web システムを構築する．

4.1 システム構成

図 1 は本システムのシステム構成図である．通常の Web ブラウザはサーバントを制御したり，サーバントが取得した Web コンテンツを利用することはできない．そこで，本システムではピア内で Web サーバソフトを起動させ，その上で利用者が Web ブラウザから Web コンテンツを利用したり，サーバントを制御できるようにする制御用 Web アプリを動作させ，これを Web ブラウザから利用することで Web ブラウザからサーバントの操作やコンテンツの管理と閲覧を可能にする．このような構成にすることによって既に広く利用されている Web サーバソフトや Web ブラウザに柔軟に対応できるようにした．

なお，本システムの実装は全て Ruby[8] で行っており，**表 1** の環境で動作確認を行った．

表 1 動作環境

OS	Mac OS X 10.11	Windows 10	Ubuntu 14.04	FreeBSD 10.2
Ruby	Ruby 2.3	Ruby 2.2	Ruby 2.0	Ruby 2.1

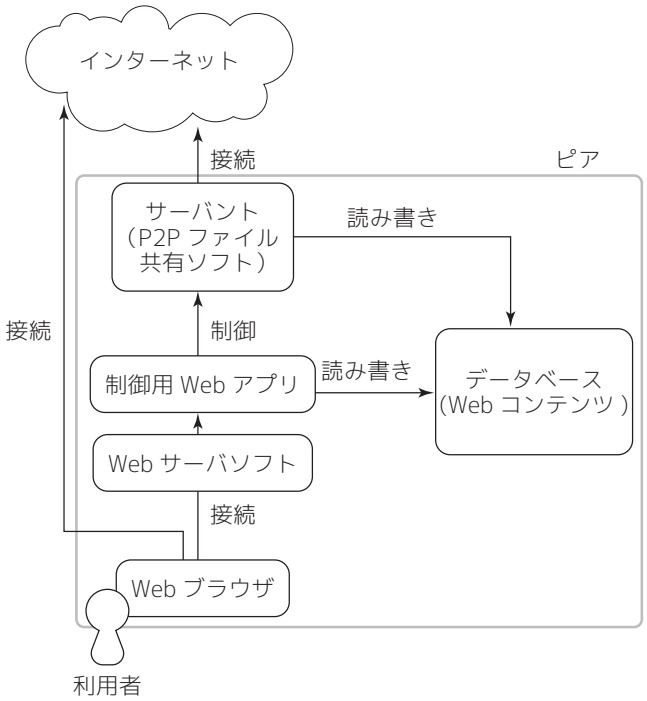

図 1 システム構成図

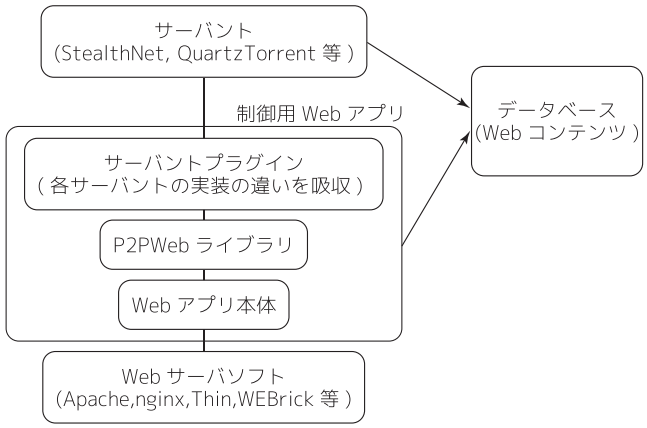

図 2 制御用 Web アプリの構成図

4.2 制御用 Web アプリ

制御用 Web アプリは Web ブラウザからサーバントを制御したりコンテンツの閲覧及び管理を行うための Web アプリである．**図 2** は制御用 Web アプリの構成図である．制御用 Web アプリは大きく 3 つの要素から成り立っている．1 つ目はサーバントプラグインである．これについては第 4.3 節で述べる．2 つ目は P2PWeb ライブラリである．このライブラリは Web コンテンツの読み込みや書き込みをはじめとした内部的な操作の API を提供するものである．そして 3 つ目は Web アプリ本体である．Web アプリ本体は Web ブラウザからのリクエストに対してレスポンスを返す処理を担当する．本システム利用者が Web ブラウザから利用する全ての機能は Web アプリ本体が P2PWeb ライブラリを内部で利用し提供される．実装には Ruby 用の Web アプリケーションフレームワーク Sinatra[9] を利用した．Sinatra を利用した Web アプリは Apache,nginx,Thin,WEBrick といった Web サーバソフトで実行できる．第 4.2.1 項から第 4.2.6 項では Web アプリ本体が提供する機能について述べる．

4.2.1 Web コンテンツ共有機能

Web コンテンツ共有機能は利用者がコンテンツを他の利用者に共有するための機能である．利用者は Web コンテンツを一般的な Web ページと同じ方法であらかじめ作成しておく．そして作成した Web コンテンツを本システムに読み込ませることで自動的に共有に適した形式へ本システムが変換を行い，変換が終われば利用者は任意のタイミングで Web コンテンツを共有することができる．

4.2.2 Web コンテンツ追加機能

Web コンテンツ追加機能は利用者がメタデータ（第 4.2.5 項）を用いて Web コンテンツを追加する機能である．Web コンテンツを追加すると同時に，サーバントによるファイル共有も行われる．

4.2.3 Web コンテンツ閲覧機能

Web コンテンツ閲覧機能はデータベース（第 4.4 節）に保存されている Web コンテンツを Web アプリ本体が Web ブラウザからのリクエストに応じて読み出し，レスポンスとして返すことで利用者が閲覧できるようにする機能である．

4.2.4 Web コンテンツ削除機能

Web コンテンツ削除機能は利用者がデータベース（第 4.4 節）に保存されている Web コンテンツを削除する機能である．Web コンテンツの削除後はサーバントによるファイル共有も停止する．

4.2.5 メタデータ作成機能

メタデータ作成機能は利用者が Web コンテンツ共有機能（第 4.2.1 項）によって読み込まれた Web コンテンツのメタデータを作成する機能である．メタデータとは共有に必要な Web コンテンツの情報を持ったデータで，サーバントプラグイン (第 4.3 節) によって形式が変化する．

4.2.6 RESTfulAPI

制御用 Web アプリは外部の何らかのプログラムが本システムと連携できるよう RESTfulAPI を提供する．一例として以下のようなリクエストによってメタデータによる Web コンテンツの取得が可能である．

```
/api/servent/donwload
メソッド：POST
パラメータ名：src
値：メタデータ
```

RESTfulAPI を用いることで本システムを用いた様々な機能を作成することが可能である．

4.2.7 独自 URL スキーム

本システムでは制御用 Web アプリも Web コンテンツもローカルの Web サーバソフトを介してアクセスするため，Web コンテンツにアクセスするための URL は以下のようになってしまう．

`http://localhost:ポート番号/contents/コンテンツ名`

localhost やポート番号の部分は利用者の環境によって異なる可能性があり，本システム向けに Web コンテンツを作成する際に，他の本システム上の Web コンテンツへのリンクを上記のような形式で記述してしまうと環境によってはリンクが機能しないことが考えられる．そこで，以下のようにリンクの記述が可能な独自 URL スキームを提供する．

`p2pweb://コンテンツ名`

4.3 サーバントプラグイン

サーバントプラグインは様々なサーバントを本システムに組み込むためのプラグインである．あらかじめ決められたインターフェースに沿ってプラグインプログラムを組むことによって StealthNet[10] や QuartzTorrent をはじめとした様々なサーバントを本システムに組み込むことが可能である．

4.4 データベース

本システムは P2P ファイル共有によって Web コンテンツを取得し手元に保存する都合上，Web コンテンツ名の重複や同じファイルがいくつもシステム上に追加されたりファイルの改ざんが行われたりする可能性がある．

そのため，Web コンテンツをハッシュ値で管理するデータベースを備える．データベースにはインデックスとオブジェクトの二種類のデータがあり，それぞれ別々のディレクトリに保存される．オブジェクトは Web コンテンツのファイル一つ一つそのものである．インデックスは Web コンテンツの持つ全てのオブジェクトのファイル名とハッシュ値の他 Web コンテンツ名や作者名等の Web コンテンツの情報が書き込まれたものである．どちらも自分自身のハッシュ値の先頭 4 桁の 2 桁ずつ 2 つのディレクトリ名に，残りの桁をファイル名として使用し保存する．この保存方法は分散バージョン管理システムの Git[11] を参考にしたものであり，大量のファイルを扱え，しかも高速に特定のファイルを探し出すことが可能である．

4.4.1 拡張 SHA2 構文

本システムのデータベースでは Web コンテンツの管理で使用するハッシュ関数に SHA2 を採用している．URL から Web コンテンツのハッシュ値を直接指定して Web コンテンツを閲覧することが可能であるが，SHA2 のハッシュ値は 16 進数表記で 64 文字もあり，覚えておくことが困難である．そこで Git の拡張 SHA1 構文 [12] を参考に，最小でハッシュ値の先頭 5 桁から 64 桁のハッシュ値を逆算できる拡張 SHA2 構文を実装した．

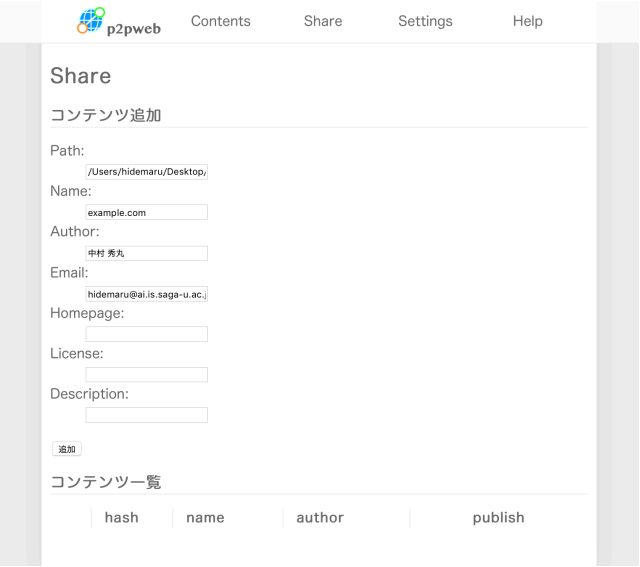

図 3 Share 画面

4.5 システムの利用

本節では本システムの利用について第 4.5.1 項から第 4.5.4 項で実際に動作させた結果の画像を参照しながら解説する．画像の画面は全て Web ブラウザのものである．

4.5.1 Web コンテンツの追加と確認

共有したい Web コンテンツを追加するにはまず，本システムの制御用 Web アプリに Web ブラウザからアクセスする．すると画面の上の方にナビゲーションバーが表示されるので，ナビゲーションバーの Share をクリックする．クリックすると Share 画面へ遷移する（**図 3**）．この画面のコンテンツ追加フォームに以下の内容を入力し追加ボタンを押すとコンテンツ一覧に Web コンテンツが追加される (**図 4**).

- Path: Web コンテンツのあるディレクトリへの絶対パス
- Name: Web コンテンツ名（ここでは仮に example.com とする）
- Author: 作者名（省略可）
- Email: 作者のメールアドレス（省略可）
- Homepage: 作者のホームページ（省略可）
- License: Web コンテンツのライセンス（省略可）
- Description: Web コンテンツの説明文（省略可）

Web コンテンツがコンテンツ一覧に表示されている時に表示を確認したい Web コンテンツの行をクリックすると追加した Web コンテンツのプレビューを確認することができる．

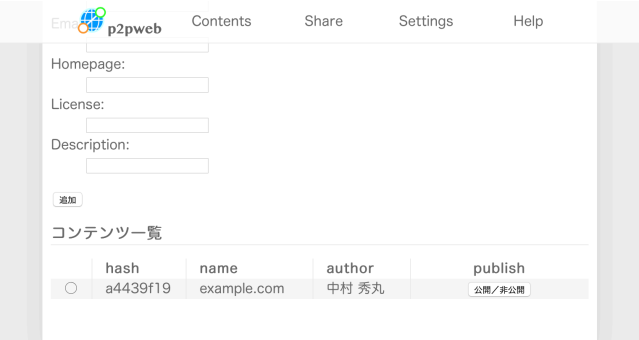

図 4 Share 画面のコンテンツ一覧に Web コンテンツが追加された様子

図 5 公開ボタン

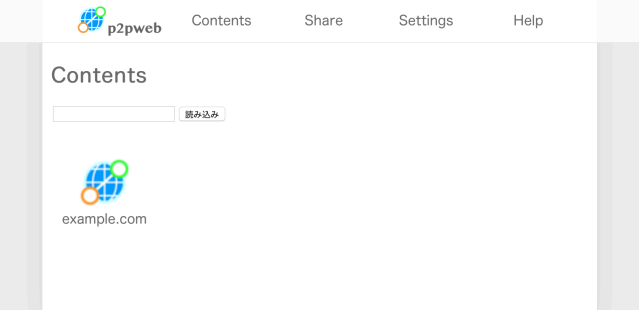

図 6 Contents 画面

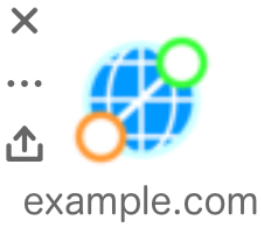

図 7 Web コンテンツのアイコンとボタン類

4.5.2 Web コンテンツの共有

第 4.5.1 項で追加した Web コンテンツを共有するには**図 5** の右側に白く表示されている公開ボタンをクリックする．すると，Contents 画面に Web コンテンツがアイコンと共に表示される (**図 6**)．Contents 画面はナビゲーションバーの Contents をクリックすることで表示できる．また，アイコンは favicon があればそれが自動的に使われ，なければダミーのアイコンが表示される． アイコンにカーソルを合わせると**図 7** のように左側に小さなボタンが 3 つ出現する．この一番下のボタンを押すとメタデータ作成機能（第 4.2.5 項）が呼び出される．メタデータ作成機能は

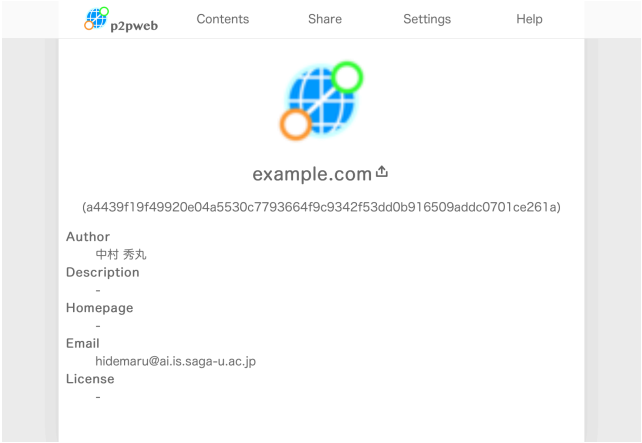

図 8 Web コンテンツの情報表示画面

選択したサーバントプラグインによって動作が変わるようになっている．本研究で実装した QuartzTorrent のサーバントプラグインでは BitTorrent クライアントで使用できるメタデータの Torrent ファイルがダウンロードされる．この時点で Web コンテンツは公開状態であり，ダウンロードされた Torrent ファイルを Web コンテンツを共有したい相手に渡せば Web コンテンツの共有が可能である．

4.5.3 Web コンテンツの閲覧

Contents 画面ではアイコンをクリックすれば Web コンテンツを閲覧することができる．図 6 で表示されているアイコンをクリックすれば Web コンテンツの閲覧画面へ遷移する．また，図 7 の左側に並んだボタンのうち，中央のボタンをクリックすることで**図 8** のような Web コンテンツの情報を確認できる画面を表示することができる．第 4.5.1 項で空欄にした項目は「-」と表示されている．

4.5.4 Web コンテンツの削除

Contents 画面に追加した Web コンテンツは図 7 の左上の×ボタンをクリックすることで削除することができる．

5. 考察

本研究では P2P 構成に基づく耐障害性を備えた Web システムのプロトタイプとして P2PWeb を構築した．本システムは制御用 Web アプリによってサーバントとデータベースを Web ブラウザから利用できるようにするというシステム構成と，サーバントプラグインによって特定の Web ブラウザ，Web サーバ，サーバントに依存せず，柔軟に対応できるように実装した．そして QuartzTorrent を用いたサーバントプラグインを実装し，実際に複数のピアによる負荷の分散，他のピアが障害を起こしてもシステムが動作し続ける耐障害性の実現およびリソース消費の分散が行えることを確認した．

しかし，本システムが実際に運用中の負荷に晒された際に従来の Web システムに対してどの程度耐障害性において優位性があるかという比較がまだできていない．これに

ついては通信障害を意図的に発生させ，その上で動作の違いを比較することが必要である．さらに負荷分散についても実際に複数のクライアント，もしくはサーバントを用いた性能評価が必要である．さらに QuartzTorrent のサーバントプラグインは DHT（分散ハッシュテーブル）を用いたファイル共有に対応しておらず，純粋な P2P 構成となっていないため，P2P 構成に基づく耐障害性を十分に備えているとはいえないため，DHT への対応も必要である．

5.1 従来の Web システムとの比較

また，従来の Web システムでは実現できていたが，本システムでは実現できていない機能として以下のようなものがある．

5.1.1 Web コンテンツの更新や削除

本システムが Web コンテンツの共有に用いている P2P ファイル共有技術では配布した後のファイルは基本的に削除や更新を行うことができない．

5.1.2 インタラクティブな Web コンテンツの提供

現在本システムは，例えサーバントにインタラクティブな P2P 通信を行う機能が実装されていても，それを活用できるようサーバントプラグインを設計できていない．また，データベースもファイルをハッシュ値で管理するため，連続して変化するデータ（例えば掲示板の書き込みログ）を保持するようなファイルを扱おうとすると，変更がある度に別のファイルとして保存してしまうため非効率的である．

5.1.3 PHP 等を用いた動的な Web コンテンツの生成

本システムでは PHP 等を用いた Web コンテンツを閲覧することができない．これは，PHP 等を本システムから呼び出すことで実現できるものと思われる．ただし，環境によって呼び出せなかった場合の処理などシステムによるフォローが必要になると思われる．

5.2 関連研究

さらに，本研究の関連研究として Project Maelstrom が存在する．Project Maelstrom は BitTorrent 社による分散 P2P ブラウザ Maelstrom の開発プロジェクト [13] で，P2P 構成に基づく Web を構築しようという目的が本研究と似ているプロジェクトである．しかし，本研究で開発した P2PWeb が Web ブラウザやサーバントに依存しないのに対して，Maelstrom は Web ブラウザの Chromium に BitTorrent クライアント機能を持たせたものである点が異なっている．また，Maelstrom の動作環境は現在 Windows のみである．

このような点から，柔軟な設計による様々なサーバントや関連技術の検討が可能で，より多くの環境で動作させることができるという点が本研究の利点としてあげられる．

6. おわりに

本研究では P2P 構成に基づく耐障害性を備えた Web システムの構築についての提案とプロトタイププログラム P2PWeb の実装を行った．本研究では特定の Web サーバや Web ブラウザ，サーバントに依存せず，P2P ファイル共有によって Web コンテンツの共有を行うことができるよう制御用 Web アプリとサーバントプラグイン，そしてデータベースを実装し，実際に動作させた．そして本システムが他のピアが障害を起こしてもシステムが動作し続けるという耐障害性を備え，複数のピアによる負荷の分散や，リソース消費の分散が行えることを確認した．今後の課題は運用中の負荷に対する評価や DHT への対応および従来の Web システムでは実現できていたが，本システムでは実現できていない機能の実装である．

参考文献

[1] YouTube, LLC: YouTube(online), 入手先 ⟨https://www.youtube.com⟩ (2016.01.29).
[2] Schollmeier, Rüdiger: *A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications*, Proceedings of the First International Conference on Peer-to-Peer Computing(2001).
[3] Napster, LLC: Napster(online), 入手先 ⟨http://www.napster.com⟩ (2001).
[4] The Gnutella Developer Forum (GDF): Gnutella - Stable - 0.4(online), 入手先 ⟨http://rfc-gnutella.sourceforge.net/developer/stable/index.html⟩ (2016.01.29).
[5] 金子勇：Winny の技術，アスキー・メディアワークス (2005).
[6] BitTorrent, Inc.: BitTorrent(online), 入手先 ⟨http://www.bittorrent.com⟩ (2016.01.29).
[7] jeffwilliams: quartz-torrent: Ruby bittorrent library.(online), 入手先 ⟨https://github.com/jeffwilliams/quartz-torrent⟩ (2016.01.29).
[8] Ruby コミュニティ：オブジェクト指向スクリプト言語 Ruby(online), 入手先 ⟨https://www.ruby-lang.org/⟩ (2016.01.29).
[9] Blake Mizerany: Sinatra(online), 入手先 ⟨http://www.sinatrarb.com⟩ (2016.02.01).
[10] Lars Regensburger & Planet Peer Team: StealthNet - Anonymes Filesharing(online), 入手先 ⟨http://www.stealthnet.de⟩ (2016.02.02)
[11] Git: Git(online), 入手先 ⟨https://git-scm.com⟩ (2016.01.29).
[12] Git: git-show-branch Documentation(online), 入手先 ⟨https://git-scm.com/docs/git-show-branch⟩ (2016.01.29).
[13] BitTorrent, Inc.: Project Maelstrom(online), 入手先 ⟨http://project-maelstrom.bittorrent.com⟩ (2016.01.29).