〈一般研究課題〉　大規模ソフトウェア解析とWeb応用のための研究開発

助 成 研 究 者　中部大学　　前田　和昭

大規模ソフトウェア解析とWeb応用のための研究開発

前田　和昭
(中部大学)

A Study on Analysis of Large-Scale Software Systems and the Web Applications

Kazuaki Maeda
(Chubu University)

Abstract：

In this study, software analysis of Web browsers and the performance evaluation were investigated. SVG is widely used to draw vector graphics image on web pages. It's designed as an application of XML so that it's possible to express various graphical representations with animations using JavaScript programs. The experience of SVG based application development indicates that different web browsers have different performance to draw the user interface. The author tried to search research papers and articles on the web about the performance issues, but there are no research papers to clarify the difference of web browsers to the best of my knowledge.

Therefore, major six web browsers, those are Chrome, Firefox, Vivaldi, Opera, Internet Explorer and Edge, were chosen, and three JavaScript programs were created for performance comparison by the author. After that, some experiments to measure the performance were done. Three types of laptop PCs were prepared to examine the performance of the web browsers to display SVG images. As a result, two web browsers, which are IE and Edge, show poor performance. Moreover, Chrome, Vivaldi and Opera show very similar value. One reason of this is that they include same implementation code. Based on another experience, a specialized compiler to analyze source code was built using a popular free and libra open source compiler. The author analyzed source code of Vivaldi so that the code includes a part of Chrome.

Corporate Vice President in Microsoft announced that next version of Edge will change drastically. It will become a Chromium based browser. The reason is that Edge should observe web standards and improve the performance. This study should continue because this year is a good timing to evaluate enhancements of web browsers. The results will be published in a future paper.

1．はじめに

インターネットの爆発的な普及は，Webページを記述するための言語HTMLとそれを表示するWebブラウザがきっかけとなって始まった．世界初のWebブラウザ WorldWideWeb は，1990年にCERN (European Organization for Nuclear Research)で開発された[1]．このWorldWideWebは，Tim Berners-Leeが書いた提案書[2]にもとづいて，彼自身が実装したアプリケーションである．それから30年が経ったことを記念して，当初と同じ見た目・使い勝手を実装するプロジェクトが立ち上がり，CERN 2019 WorldWideWeb Rebuildとして公開されている[3].

WorldWideWebを起動させてから，中部大学トップページを開いたときのスクリーンショットを図1に示す．中部大学のトップページ部分には文字しか表示されていないことから分かるように，当時はWebページに画像を埋め込むことができなかった．最近のWebブラウザでは，画像だけでなく音声や動画を扱うことができる．さらには，動的なコンテンツを提供するためにプログラムを埋め込む技術としてFlashやJavaアプレットなどが使えるように機能拡張が行われてきた．その結果，初期のWorldWideWebに比べて最新のWebブラウザのサイズは膨れ上がり，巨大なアプリケーションになった．それとともに，この30年間で各Webブラウザの市場シェアは大きく変わっていった．

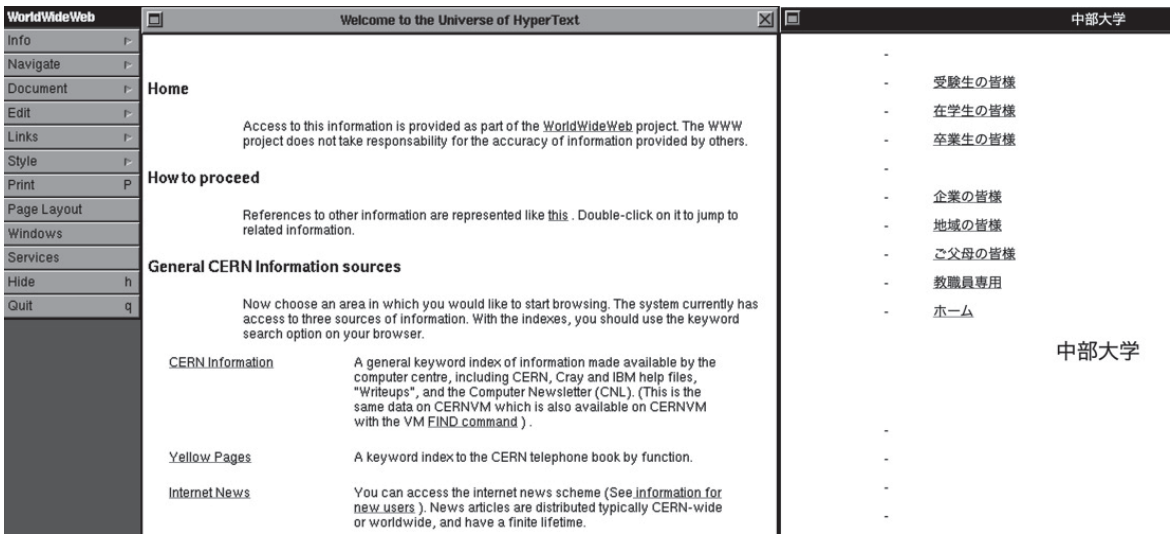

図1. CERN 2019 WorldWideWeb Rebuildの実行画面の一部

2009年から2015年までの市場シェア変動グラフをWikipedia

https://en wikipedia.org/wiki/File:Usage_share_of_web_browsers_(Source_StatCounter).svg

から抜粋し加工して図2に示す．白黒印刷で比較可能とするため，元画像ではカラー表示だったグラフ内のInternet Explorerの線を太く　Chromeの線を太い破線になるように，テキストエディタを使って筆者が直接データを書き換えた．2009年頃には70%近くを占めていたInternet Explorerが2015年頃には10%ぐらいまで減少し　その一方 Google Chromeが急速に普及したことが分かる．

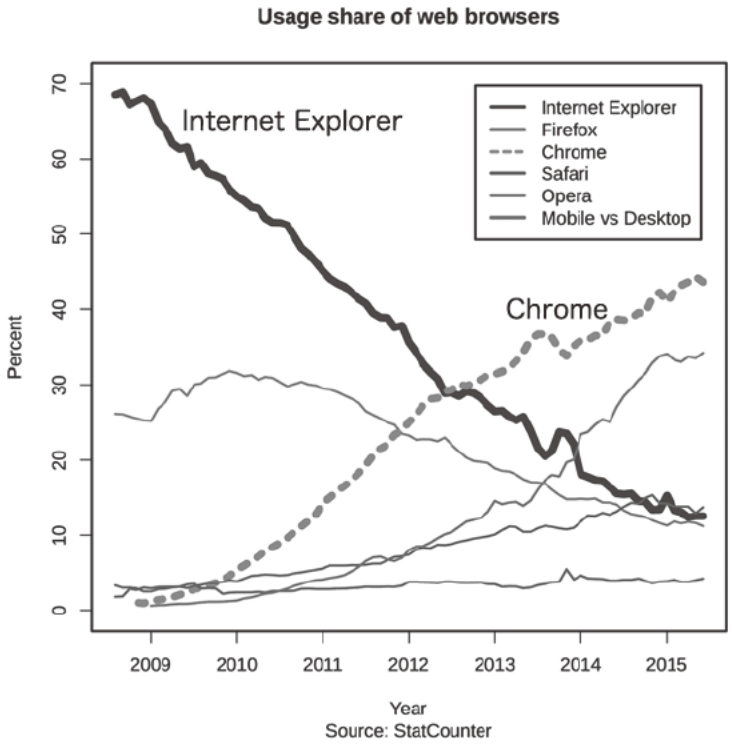

図2．2009年以降のWebブラウザ市場シェア

2019年1月のWebブラウザ市場シェアを，StatCounter社が提供するデータ[4]から抜粋して表1に示す　この表では，日本・世界　米国・欧州の地域ごとにWebブラウザ市場シェアの違いが分かるようにした．日本ではInternet Explorerの割合が他と比べて高めであり，また，欧州ではFirefoxとOperaが他と比べて高めになっている．地域によって細かな違いがあるものの，Chromeは50%以上を占め，Internet Exploreは10%前後であることが分かる．

表1　デスクトップPCでのWebブラウザ市場シェア（2019年1月 各地域での割合：単位 %）

Web ブラウザ	日本	世界	米国	欧州
Google Chrome	58.46	70.88	63.19	63.17
Internet Explorer	15.19	5.74	10.1	6.08
Mozilla Firefox	9.47	9.5	8.43	13.58
Safari	8.3	5.15	8.31	6.52
Microsoft Edge	6.69	4.14	8.6	5.4
Opera	0.84	2.37	0.82	3.49

最近のWebブラウザには，SVG(Scalable Vector Graphics)による画像を扱う機能が実装され，２次元グラフィックスをWebページ上に描くために広く使われている．図3に示すように，図2で示したグラフはSVGで提供されていたため，テキストエディタを使ってデータを書き換えることが可能であった．SVGは，ベクター画像をXMLで記述するように設計されているため，プログラムから操作することで多様な表現が可能となる．本稿では，SVGを使ってWebアプリケーションを構築してきた経験を踏まえ，SVGの概要を説明し，品質の良いWebアプリケーションを作り上

げるために行った性能評価実験とその結果について述べる．なお本稿は，発表済みの論文[5]の内容を書き改め，新しく作り直した論文である．

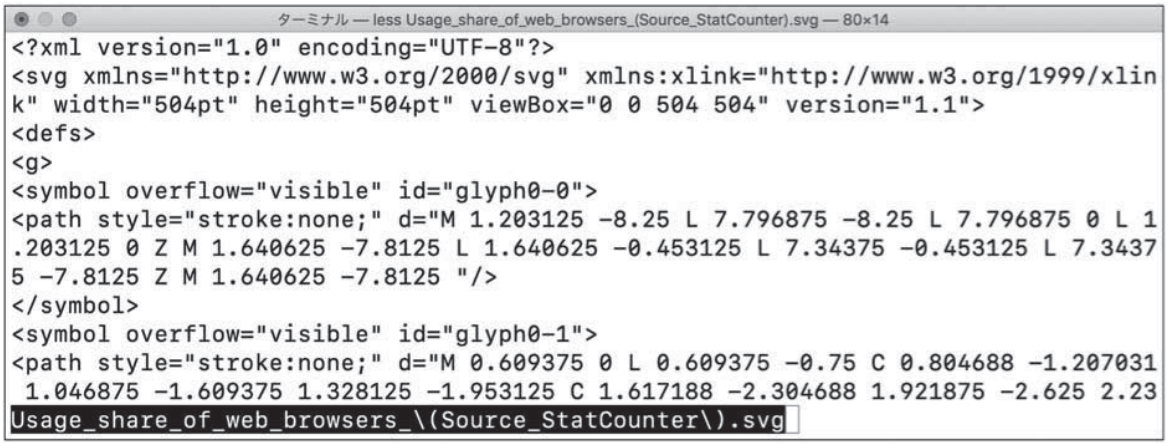

図3.Usage_share_of_web_browsers_(Source_StatCounter).svg の内容

2. XMLと画像

WWW普及の立役者となったHTMLは，文書中に印刷命令を表現するタグを埋め込むために設計された．これに対してXMLは，データに意味を付記できるようにしたマークアップ言語である．コンピュータでの処理やデータベースに格納するなど，ビジネスで広く活用できるように工夫がなされている．例えば，文献データや注文伝票などを記述したものがそれにあたり，アプリケーション間でデータを交換したり，またデータベースからデータを取り出して他の製品で活用する場面など多くの応用分野がある

```
<inproceedings mdate="2017-05-21" key="conf/bionetics/Maeda14">
    <author>Kazuaki Maeda</author>
    <title>Parser Development with an Internal Domain-Specific Language and
           an Aspect Weaver.</title>
    <year>2014</year>
    <booktitle>BICT</booktitle>
    <ee>https://doi.org/10.4108/icst.bict.2014.257977</ee>
    <crossref>conf/bionetics/2014</crossref>
    <url>db/conf/bionetics/bict2014.html#Maeda14</url>
</inproceedings>
```

図4. DBLP論文データの一部

XMLデータを処理する場合，データの容量が膨れあがることに注意が必要である．例えば，コンピュータ科学の論文文献データを提供するDBLP[6]の中から筆者執筆による論文の部分を抜粋して図4に示す．XMLは，人間が読むことができ，また，プラットフォーム非依存にするため，タグによって構造化された文字データを使って表現されている．そのため，バイナリ形式や簡易テキスト形式に比べて，3倍から20倍もの容量が必要だといわれている[7]．DBLPをXMLデータとして書き出したところ約2.5GBの容量となった(2019年5月26日現在)．

コンピュータ科学辞典[8]には，画像について以下のような定義が述べられている．

(以下，Oxford Dictionary of Computer Science[8]より抜粋)

Computer graphics: Computer graphics is the creation of, manipulation of, analysis of, and interaction with pictorial representations of objects and data using computers. There are two quite different ways in which the information is stored and manipulated, usually referred to as raster graphics and vector graphics.

Raster graphics: This is the system used for scanned images and for photographs produced by a digital camera. It is suitable for complex pictures with a large number of different colors and shades. The image is stored as chunks of data, each representing the properties (e.g. color, tint, transparency) of small element of the picture.

Vector graphics: This system is used for diagrams, graphs, flow charts, engineering drawings, etc. It is suitable for relatively simple line drawings. The image is stored and manipulated as mathematical formula producing lines geometrical figures, and curves (e.g. Bezier curves).

本稿では，Raster graphicsのことを「ラスター画像」，Vector graphicsのことを「ベクター画像」と記すことにする．図5左側にラスター画像(PNG形式)を，右側にベクター画像(AI形式)を表示したときの様子を示す．これらは，同じ絵を描いてから各画像を10倍に拡大したときの表示結果を並べたものである．左側のラスター画像はギザギザが目立つのに対して，右側のベクター画像は滑らかな曲線になっている．これらの差は，ピクセルでデータを持つか，図形命令でデータを持つかの違いに由来する．

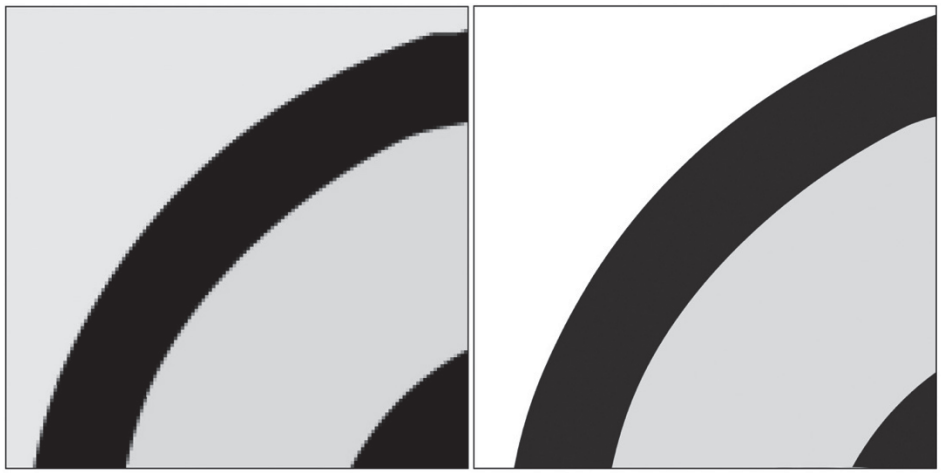

図5. ラスター画像(左)とベクター画像(右)

最新のWebブラウザでベクター画像を扱うにはSVGを使うことが多い．SVGは，ベクター画像を表現するためのXMLデータ形式であり，W3Cにより仕様が作成されている．SVG 1.0が2001年に，SVG 1.1が2011年に勧告として公開された[9,10]．開発者らによって保守されているWebサイト[11,12]によれば，代表的なWebブラウザには SVG 1.1を満たす機能が実装済みとなっている．

SVGを説明するために，SVGによる簡易な記述を図6に示す．この記述をWebブラウザで読み込むと，図7に示す円・線分・テキストの組み合わせが表示される．SVGは，Webブラウザで表示されるためだけではなく，グラフィックスツールに読み込んで加工し保存できる．最近では，グラ

フィックスツール間でデータを交換するためにも使われている.

```
<svg width=“400” height=“200” >
  <circle cx=“200 cy=“100” r=“80” fill=“none” ></circle>
  <line x1=“200” y1=“20” x2=“200” y2=“180” ></line>
  <text x=“40” y=“160” >晴れ</text>
</svg>
```

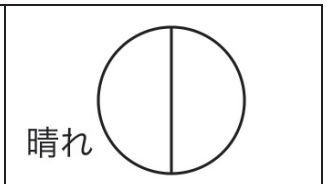

図6. 円・線分・テキストをSVGで記述した例

図7. Webブラウザでの表示結果

3. SVGの3通りの使い方

筆者のこれまでの経験から，SVGは

1. 静的に使う
2. アニメーションに使う
3. 動的に使う

の３通りの場合があると考える.「静的に使う」ときは，Adobe社Illustratorなどのグラフィックツールを使って絵を描き，SVGで保存する場合である　例えば，図8に，橋本氏が描いた東京近辺の鉄道路線図の一部を示す[13]. この路線図はパブリックドメインで公開されているため，SVGを説明するためのデモや，JavaScriptプログラムで操作する素材として使うことを検討した　しかし，Adobe社Illustratorを使って丹念に生成されたベクター画像をSVGで保存したファイルは，ファイル１つが約9,000行の記述からなり，これをJavaScriptプログラムから操作することは困難と判断した.

図8. 鉄道路線図の一部

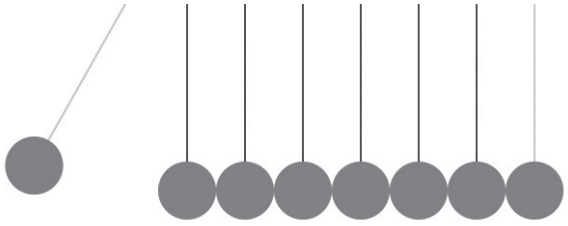

図9. アニメーションの例

「アニメーションに使う」ときは，例えばanimateTransform要素を使い，SVGで記述された特定の要素に対して，移動や回転，変化の始点・終点，変化の開始・終了タイミングを指定して，動きを表現する場合である. 図9に，書籍[14]で紹介されているニュートンのゆりかごの例を参考にして，筆者が新たに描いた例を示す. このアニメーションでは，SVGで表現されたXMLのみで動きを記述し，左端の円とそれに接続する線分と，右端の円とそれに接続する線分を左右に30度ずつタイミングを合わせて交互に回転させている.

図10. 毎秒ごとに針の表示を更新する時計

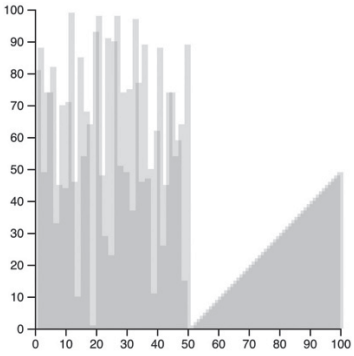

図11. 毎秒データを更新する棒グラフ

３つめの「動的に使う」ときは，定期的(例えば毎秒ごと，または，ユーザの要求時)に表示を変化させる場合である．性能評価のために筆者が作成した，毎秒ごとに針の表示を更新する時計の例を図10に，毎秒データを更新する棒グラフの例を図11に示す．

次節では，３つめの「動的に使う」ときのみを対象とし，SVGで作成されたベクター画像を，JavaScriptを使って決められたタイミングで更新していくプログラムを使って性能評価を行った結果について報告する．

4. Webブラウザの性能評価

これまでSVGを使ってWebアプリケーションを開発してきた経験では，Webブラウザを変えると動作速度が異なることを感じることが多かった．この動作速度について調査をしたものの，SVGで作られたベクター画像を表示する場面で，Webブラウザによる違いを示した研究論文または解説記事が見つからなかった．そこで，各Webブラウザでの実行時間を計測するためのJavaScriptプログラムを作成し，代表的なWebブラウザを使って性能評価実験を実施することにした．

4.1 JavaScriptプログラムによるWebブラウザ毎の性能評価

実験を行うハードウェアとして，次に示す仕様のノートブックPCを３台準備した．これらの購入時期が2018年，2016年，2014年であることから，それぞれ PC 2018，PC 2016，PC 2014と名付けることにする．まずは，これら全てのPCに内蔵されたHDDまたはSSDを初期化した後，Windows 10をインストールし，その後ウィルス対策ソフトウェアなど実験に関係ないソフトウェアは一切インストールしないことにした．

- PC 2018 - Intel Core i7 (7820HQ) 2.90GHz，主記憶：16GB，SSD：2TB，
  Windows 10 Pro, Version 1703
- PC 2016 - Intel Core i5 (6200U) 2.30GHz，主記憶: 8GB，HDD：500GB，
  Windows 10 Home, Version 1803
- PC 2014 - Intel Core i5 (4308U) 2.80GHz，主記憶: 16GB，HDD：1TB，
  Windows 10 Pro, Version 1803

さらに，以下に示す代表的な６つのWebブラウザをインストールして，それらの上で筆者が作成したSVGデータを含むWebページを表示しJavaScriptプログラムを動かして，実行時間を測定した．

- Google Chrome, Version 70.0.3538.110
- Mozilla Firefox, Version 63.0.3
- Vivaldi, Version 2.1.1337.51
- Opera, Version 56.0.3051.116
- Internet Explorer (IEと略記する), Version 11.1418.15063.0
- Microsoft Edge, Version 40.15063.674.0

性能測定のために作成したJavaScriptプログラムは以下の３種類である.

・時計(SVGを主)と時計(D3を主)

２つの乱数を生成し，その値にしたがって図10で示す時計の長針・短針を回転させる．この時計を同じWebページ上に９個配置して，長針・短針の回転処理を100回行い，プログラム実行開始前と終了後の時間の差から実行時間を計算する．さらに，３秒に１回Webページ全体を更新する処理を100回実行して平均値を求める．実装の違いによる性能を評価したかったので，ベクター画像の大部分をSVGで描くことでJavaScriptのプログラム量を最小限にしたもの(時計(SVGを主)と記す)と，SVGによる記述を最小限にして大部分をD3[15]のAPIを使ってJavaScriptで記述したもの(時計(D3を主)と記す)の２つを準備した.

・棒グラフ

乱数50個を生成し，図11で示した棒グラフを更新する．各データを左側(棒グラフの原点)から順番に挿入し，表示されているデータを100個に限定するため，右側(棒グラフのx座標値100)から削除している．この棒グラフを同じWebページ上に３個配置して，データの更新処理を50回行い，プログラム実行開始前と終了後の時間の差から実行時間を計算する．さらに，３秒に１回Webページ全体を更新する処理を100回実行して平均値を求める.

4.2　性能評価の結果

性能評価実験で求めることができた実行時間を表2に示す．この結果から，IEとEdgeのときが他に比べて処理時間が大きいことが分かる．また，Chrome，Vivaldi，Operaの実行時間が似た値を示していることは，Webブラウザでの表示機能を実装しているレンダリングエンジンが同じである

表2. Webブラウザ毎のJavaScriptプログラム実行時間(単位：ミリ秒)

		Chrome	Firefox	Vivaldi	Opera	IE	Edge
PC2018	時計(SVG を主)	17.77	29.79	18.54	20.09	102.19	126.10
	時計(D3 を主)	22.03	31.13	21.88	29.76	108.05	147.66
	棒グラフ	108.86	233.64	110.56	97.88	433.12	360.06
PC2016	時計(SVG を主)	13.78	16.24	15.46	15.13	90.91	103.42
	時計(D3 を主)	15.20	16.07	15.29	15.85	84.74	100.49
	棒グラフ	108.08	197.7	111.30	103.08	447.88	517.08
PC2014	時計(SVG を主)	28.31	37.40	33.80	33.23	97.73	126.10
	時計(D3 を主)	28.58	41.46	35.94	40.60	100.12	132.06
	棒グラフ	171.42	315.64	163.10	164.80	510.70	586.94

という事実を示唆しているものと思われる．表2で示した実験結果を求める前段階で，同じWebページに複数のSVGデータを置いてから，１秒に１回の頻度で表示を更新していた．しかし，Webブラウザによっては更新のタイミングに追いつくことができず異常な値を示していたため，３秒に１回の頻度で実験を行うことになった．これら結果として出てきたWebブラウザ間での優劣は，筆者が経験したことに近い感触を得ている．

4.3　Webブラウザのソースコード解析

Webブラウザのソースコードを解析するにあたり，一般に公開されているGoogle ChromeのオープンソースChromiumとFirefoxのソースコードを対象とした．ChromiumやFirefox に代表されるWebブラウザは，大規模ソフトウェアの代表例であり，大部分がC++を使って開発されている．IEEE Spectrumによるプログラミング言語のランキング(2018年)[16]によれば、C++のランキングは第2位で，多数の開発者が利用していることが分かる．

しかし，国際会議にて本研究者が研究者たちと討論したところ，C++の言語仕様が複雑すぎて解析のためのツールを自前で構築するには膨大な時間(1人で構築する場合で数ヶ月程度)が必要となり無謀であるとの結論に至った．そこで，これまでに逆エンジニアリングツールを開発した本研究者の経験をもとに，解析ツールの中心となる構文解析プログラムを構築し，現在，応用実験を進めている．

ChromiumとFirefoxの入手可能な版をダウンロードして解析したところ，

- Chromiumは，ファイル数が約13万，ソースコード行数が約2,600万
- Firefoxは，ファイル数が約10万，ソースコード行数が約1,500万

であった．ファイル数とソースコード行数以外の解析結果については，今後の研究活動で活用する．

4. おわりに

本稿では，SVGを使ってWebアプリケーションを構築してきた経験を踏まえ，SVGの概要を説明し，SVGに基づくベクター画像を表示するときのWebブラウザ毎の性能評価について述べた．現段階で示すことができる性能評価実験の結果は，各Webブラウザの限定した特徴を示しているにすぎない．Microsoftからの発表[17]によれば，Edgeの次バージョンはChromiumベースに変更され，劇的な性能向上が期待されているようで，今年はWebブラウザの性能測定を行い解析を進める非常に良い機会となるであろう．今後さらなる実験・評価と解析を続けていく予定である．

参考文献

[1] James Gillies and Robert Cailliau, How the Web Was Born, Oxford University Press (2000).

[2] Tim Berners-Lee, Information Management: A Proposal (1989), https://www.w3.org/History/1989/proposal.html (2019年5月26日参照).

[3] CERN 2019 WorldWideWeb Rebuild, https://worldwideweb.cern.ch/ (2019年5月26日参照).

[4] StatCounter Global Stats - Browser, OS, Search Engine including Mobile Usage Share, http://gs.statcounter.com/ (2019年5月29日参照).

[5] 前田和昭, SVGによるユーザインタフェース構築の検討, 第14回情報システム学会全国大会・研究発表大会 (2018).
[6] Zap Think, The "Pros and Cons" of XML, Zap Think Research Report (2001).
[7] dblp: computer science bibliography, https://dblp.uni-trier.de/ (2019年5月26日参照).
[8] Oxford Dictionary of Computer Science 7th edition, Oxford University Press (2016).
[9] Scalable Vector Graphics (SVG) 1.0 Specification, https://www.w3.org/TR/2001/REC-SVG-20010904/ (2019年5月26日参照).
[10] Scalable Vector Graphics (SVG) 1.1 (Second Edition), https://www.w3.org/TR/SVG11/(2019年5月26日参照).
[11] Can I use... Support tables for HTML5, CSS3, etc, https://caniuse.com/#search=SVG (2019年5月26日参照).
[12] HTML svg tag, https://www.w3schools.com/tags/tag_svg.asp (2019年2月26日参照).
[13] 東京の鉄道路線図SVGを作りました&パブリックドメインで配布します - Liner Note, https://note.openvista.jp/2014/svg-rail-map (2019年5月26日参照).
[14] Damian Dawber, Learning Raphael JS Vector Graphics, Packt Publishing (2013).
[15] D3.js - Data-Driven Documents, https://d3js.org/ (2019年5月26日参照).
[16] The 2018 Top Programming Languages, https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages (2019年5月26日参照).
[17] Microsoft Edge: Making the web better through more open source collaboration, Windows Experience Blog,https://blogs.windows.com/windowsexperience/2018/12/06/microsoft-edge-making-the-web-better-through-more-open-source-collaboration/ (2019年5月26日参照).