

**Research on the improvement of Web contents  
compatibility conducive to the widespread use of  
OSS desktops**

**Research report**

February 2007

Information-technology Promotion Agency, Japan

Firefox is trade mark of Mozilla Foundation in the United States and other countries. Internet Explorer, Microsoft Office, and Microsoft Windows 98/Me/2000/XP are trade mark of Microsoft Corporation in the United States and other countries. All other names of products and companies are in general property of their respective companies. Please note that <sup>TM</sup>, © and ® marks are omitted in the text of this document.

# Table of Contents

Chapter 1 Introduction .....	1
1.1 Background .....	1
1.2 Objective .....	1
1.2.1 Problem-solving techniques .....	1
1.2.2 A prospective look at use .....	2
Chapter 2 Technical research on factors responsible for web interoperability discrepancy and methods for improvement .....	5
2.1 Collection, classification, and systematic arrangement of TouchUp scripts, and analysis of web interoperability discrepancies .....	5
2.1.1 Research procedure .....	5
2.1.2 Research results .....	6
Chapter 3 Research on actual content conditions responsible for browser non-compatibility .....	19
3.1 Systematic arrangement of rule sets to collect non-compatible content automatically .....	19
3.1.1 Research procedure .....	19
3.1.2 Preparation of modules for determining non-compatibility (Modules) .....	19
3.1.3 General-purpose and systematic arrangement .....	21
3.1.4 Study of crawling results .....	27
3.2 Enhancement of rule sets to collect non-compatible content automatically .....	28
3.2.1 Research procedure .....	28
3.2.2 List of added Modules .....	28
3.2.3 Study of crawling results .....	28
3.2.4 Web interoperability discrepancy identified after research .....	30
3.2.5 Comments from the person in charge of research .....	31
3.3 Development of crawling tools for automatic collection .....	31
3.3.1 Work procedure .....	31
3.3.2 Course of development .....	31
3.3.3 Script for collection of basic data to be processed .....	33
3.3.4 Download script .....	33
3.3.5 Tools for checking .....	35
3.4 Data collection by web crawling .....	36
3.4.1 Work procedure .....	36

3.4.2 Status of crawling implementation .....	37
3.4.3 Results of implementation of crawling .....	48
3.4.4 Analysis of crawling results .....	52
3.5 Research on situations of intranets .....	60
3.5.1 Work procedure .....	61
3.5.2 Organizations on which research was to be performed .....	63
3.5.3 Research results (Example in a company A) .....	64
Chapter 4 Summary .....	77

# Chapter 1 Introduction

## 1.1 Background

Different browsers will often display the same web content differently. As a consequence, users may find themselves growing dependent on a specific browser. The problem is rooted in misinterpretation: though the W3C (World Wide Web Consortium) has defined a standard, users interpret the standard differently or create content that fails to conform with the standard provisions (including defective conditions) of a specific browser. While users have frequently pointed out the non-compatibilities of browsers (we call them “the web interoperability discrepancies”), they have only done so sporadically. Little action has actually been taken to improve the situation and work towards the realization of a universal implementation of multiple platforms or the widespread use of OSS (Open Source Software) desktops.

To improve this situation and enhance the compatibility of web content, developers need to research the problems with web browser compatibility and examine the actual conditions when content with compatibility problems is used.

## 1.2 Objective

### 1.2.1 Problem-solving techniques

The following methods were used to study solutions for the problems described in the “Background” section.

(a) Technical research on factors responsible for web interoperability discrepancy and methods of improvement

- We identified principal factors responsible for non-compatibility among browsers, based on the specifications and behaviors of the latest practical versions of major commercial browsers and open source browsers as the criteria. This step focused on the state of compatibility among web content presented to the general public.
- We investigated typical methods for improving the factors found to be responsible for non-compatibility. Proposed improvements were considered and used as documented recommendations for the vendors of authoring tools and development tools.

(b) Research on actual content conditions responsible for browser non-compatibility

- We conducted exhaustive research on websites open to the Internet when content with the problems pointed out in (a) above were used.
- We researched websites intended for intranets through interviews or automatic techniques

by setting two or more sites found to be both researchable and of high significance in research.

### 1.2.2 A prospective look at use

Based on a list of identified problems and areas found to be in need of improvement (one of the accomplishments in this research), we prepared scripts to correct the problems and to register in the TouchUpWeb system. When used directly in Firefox, an OSS browser, etc., these scripts help users browse non-compatible web content. Concretely, they were designed to correct problems such as incorrect displays and inoperable buttons. The accomplishments of this research can also be used for the development of problem-checking tools or TouchUp scripts themselves.

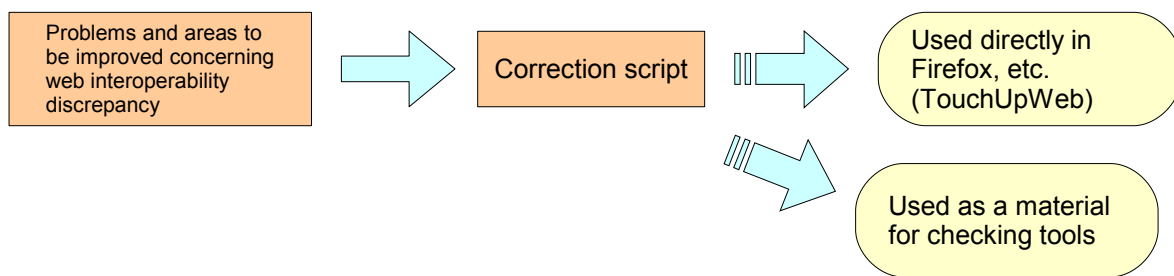


Fig. 1 A prospective look at how research results may be used

We also pointed out other accomplishments and presented documents recommending the proposed improvements to the vendors of web content authoring tools and web system development tools. Thus, the documented recommendations were used as guidelines for the development of these types of tools.

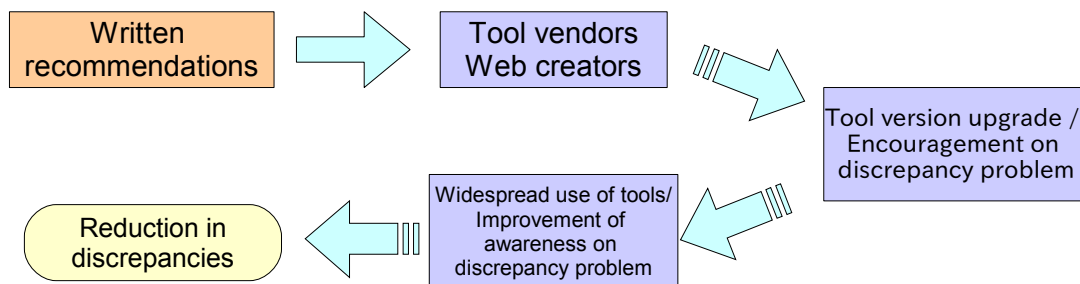


Fig. 2 Effect of documented recommendations

As preliminary preparations, we defined a rule set for use on a general-purpose basis as a tool for checking non-compatibility and prepared a crawling check tool to collect content from the Internet and check it cyclically. To efficiently rotate the processes described below, we publicized the TouchUpWeb system and made it universally known to the community in a manner that would assure its acceptance.

As soon as the preparations were made, we carried out the central work operations in this research, that is, technical research on factors responsible for non-compatibility and methods for improvement, and cyclopedic research on non-compatibility on the Internet. Intranets were also examined, on a separate basis.

Lastly, we analyzed the research, documented the non-compatibility factors and improvement methods in a report on accomplishments, and prepared a set of documented recommendations. The enlightenment for tool vendors and web creators was carried forward through this approach.



## Chapter 2 Technical research on factors responsible for web interoperability discrepancy and methods for improvement

### 2.1 Collection, classification, and systematic arrangement of TouchUp scripts, and analysis of web interoperability discrepancies

#### 2.1.1 Research procedure

By running the TouchUpWeb system, we collected and prepared TouchUp scripts to correct non-compatible conditions and web interoperability discrepancies. Next, the TouchUp scripts collected by the TouchUpWeb system were classified and broken down into patterns. The principal factors were thus picked out and typical methods for improvement were systematically arranged.

During these work operations, non-compatible conditions were arranged systematically into four levels: “ignorable,” “modifiable (automatically modifiable),” “modifiable (requiring manual inspection), and “un-modifiable.” The tendencies of non-compatible conditions and the measures for dealing with non-compatible conditions were analyzed through the foregoing steps.

The analysis included literature research on the following.

- Reports on the Web standardization made in Mozilla Group’s Bugzilla-jp (<http://bugzilla.mozilla.gr.jp/>)
- Japan top website project report (<http://www.mozilla.gr.jp/webtest/finalreport.html>)
- DB registered in Firefox’s “Reporting websites with problems” (Information on results of browsing in URL and Firefox; <http://reporter.mozilla.org/>)
- Feedback information collected by the TouchUpWeb system (Information on results of browsing in URL and Firefox)
- Commercially available books commentating on the preparation of content for the various browsers
  - “Dictionary on HTML & XHTML & CSS Explained in Detail”
  - “Must-have Dictionary on HTML/CSS/JavaScript Web browser compatibility”
  - “Super-Illustrated Dictionary on HTML & CSS”

The results of these studies were also used as the basis for research tools and TouchUp scripts. In concrete terms, we focused our attention on web interoperability discrepancy factors obtained from these studies and used the results of the research as input to further study rule sets for research tools and the development of TouchUp scripts.

Further, results collected by automatic collection tools were also added to the analysis of web interoperability discrepancy factors. We clarified the causes of the web interoperability discrepancy factors by studying web interoperability discrepancy factors attributable to the browsers' own functions, software implementations out of compliance with the standard, differences in interpretation of the standard, and the use of incompatible media.

## 2.1.2 Research results

As a result of the research, 172 web interoperability discrepancy factors were found. These factors can be classified from two points of view shown below.

- a) Areas containing factors
- b) Comparison with the standard

In addition, the web interoperability discrepancy factors in a) and b) can be further classified into the four types shown in Table 1 and the five types shown in Table 2, respectively.

The following table lists the web interoperability discrepancy factors according to the classifications in a) and b). In this case, however, we must note that some of the discrepancy factors may not be covered. An identification of all discrepancy factors would require a complete grasp of the specifications and bugs of all browsers. This is an unrealistic task.

Table 1 Classification of web interoperability discrepancy factors (a)

Classification	Description of classification
HTML/XHTML	Web interoperability discrepancy factors caused by HTML or XHTML tags, attributes, values, etc.
CSS	Web interoperability discrepancy factors caused by CSS contained in HTML or XHTML or by properties, values, etc. contained in external CSS
HTML/XHTML と CSS	Web interoperability discrepancies caused by HTML/XHTML and CSS in combination (not by either independently)
Javascript/DOM	Web interoperability discrepancy factors caused by properties, methods, etc. of Javascript or DOM

Table 2 Classifications of non-compatibility factors (b)

Classification	Description of classification
Browser's proprietary functions	Web interoperability discrepancy factors caused by the browser's extended capability arising from any of the following which isn't included in the W3C standard, etc.: tags, attributes or properties added by the browser on its own, scripts, XML, etc. described in HTML/XHTML files, and so forth.
Implementation in disregard of the standard	Web interoperability discrepancy in which display and operation vary from one browser to another even though the W3C standard, etc. contain tags, attributes and properties, as the browser has been implemented in disregard of the display and operation stipulated in the standard.
Implementation with a different interpretation of the standard	Web interoperability discrepancy in which display and operation vary from one browser to another because the standard is ambiguous and may be interpreted in more than one way, even though the browser is implemented in accordance with the provisions of the standard
Unimplemented functions	Web interoperability discrepancy in which display and operation differ from those of an implemented browser as the functions stipulated in the standard are not implemented
Use of non-compatible types of media	Web interoperability discrepancy arising from the use of media that can only be reproduced in specific browsers and platforms

### 2.1.2.1 HTML/XHTML

There are 59 web interoperability discrepancies caused by tags and attributes included in HTML or XHTML. The number of web interoperability discrepancies caused by the browser's proprietary functions is 41.

#### (1) Browser's proprietary functions

There are 41 web interoperability discrepancies caused by the use of propriety functions of IE or Netscape as tags and attributes in HTML or XHTML. Given that few sites are expected to be specialized for Netscape, the actual source of the problems is the set of IE proprietary functions. Most of the attributes related to the display, such as color and text position, can be replaced by a style sheet.

Table 3 Web interoperability discrepancies caused by browser's proprietary functions (HTML/XHTML)

ID	Tag / attribute as a cause	Type of cause	Correctable or not	Rule set
31	<bgsound>	Proprietary tag if IE	Automatically correctable	x
33	<comment>	Proprietary tag if IE	Automatically correctable	x
35	<marquee>	Proprietary tag if IE	Manually correctable	x
127	width, height, and bordercolor attributes of <frame> tag	Proprietary attribute of IE/NN	Automatically correctable	x
126	border, bordercolor, frameborder, and framespacing attributes of <frameset> tag	Proprietary attribute of IE/NN	Automatically correctable	x
52	src attribute of <applet> tag	Proprietary attribute of IE	Automatically correctable	x

57	galleryimg attribute of <img> tag	Proprietary attribute of IE	Ignorable	x
56	dynsrc attribute of <img> tag	Proprietary attribute of IE	Automatically correctable	x
44	bgproperties attribute of <body> tag	Proprietary attribute of IE	Automatically correctable	x
46	scroll attribute of <body> tag	Proprietary attribute of IE	Ignorable	x
45	topmargin and rightmargin attributes of <body> tag	Proprietary attribute of IE	Automatically correctable	x
53	valign attribute of <caption> tag	Proprietary attribute of IE	Automatically correctable	x
54	valign attribute of <fieldset> tag	Proprietary attribute of IE	Automatically correctable	x
128	hspace and vspace attributes of <iframe> tag	Proprietary attribute of IE	Automatically correctable	x
55	event attribute of <script> tag	Proprietary attribute of IE	Manually correctable	x
58	for attribute of <script> tag	Proprietary attribute of IE	Manually correctable	x
59	align attribute of <script> tag	Proprietary attribute of IE	Automatically correctable	x
60	bordercolorlight and bordercolordark attributes of <table><tr><td> tag	Proprietary attribute of IE	Automatically correctable	x
38	bordercolor, bordercolorlight, and bordercolordark attributes of <tr> tag	Proprietary attribute of IE	Manually correctable	x
62	bordercolor, bordercolorlight, and bordercolordark attributes of <th><td> tag	Proprietary attribute of IE	Automatically correctable	x
61	contenteditable attribute	Proprietary attribute of IE	Ignorable	x
39	disable attribute	Proprietary attribute of IE	Ignorable	x
40	hidefocus attribute	Proprietary attribute of IE	Ignorable	x
41	language attribute	Proprietary attribute of IE	Ignorable	x
42	unselectable attribute	Proprietary attribute of IE	Ignorable	x
43	imagetoolbar attribute	Proprietary attribute of IE	Ignorable	x
47	<blink>	Proprietary tag of NN	Automatically correctable	x
32	<embed>	Proprietary tag of NN	Ignorable	x
114	<ilayer>	Proprietary tag of NN	Automatically correctable	N/A
119	<keygen>	Proprietary tag of NN	Not correctable	x
116	<layer>	Proprietary tag of NN	Automatically correctable	x
118	<nolayer>	Proprietary tag of NN	Automatically correctable	x
117	<multicol>	Proprietary tag of NN	Not correctable	x
112	<nobr>	Proprietary tag of NN	Automatically correctable	N/A
115	<noembed>	Proprietary tag of NN	Ignorable	N/A
120	<nolayer>	Proprietary tag of NN	Automatically correctable	N/A
37	<spacer>	Proprietary tag of NN	Manually correctable	x
113	<wbr>	Proprietary tag of NN	Automatically correctable	x
129	lowsrc and suppress attributes of <img> tag	Proprietary attribute of NN	Automatically correctable	x
125	marginheight and marginwidth attributes of	Proprietary attribute of NN	Automatically correctable	Same as

	<body> tag			45
124	hspace and vspace attributes of <table> tag	Proprietary attribute of NN	Automatically correctable	x

## (2) Implementation in disregard of the standard

There are 7 web interoperability discrepancies caused by a deviation of the implementation of the browser for tags and attributes contained in HTML or XHTML from the standard. The PLEASE INDICATE is mostly implemented in a manner whereby even an inaccurate description from a content creator will still be recognized. For example, “\” can be used in addition to “/” in a URL or the type attribute of <object> tag can be omitted.

Table 4 Web interoperability discrepancies arising from implementation in disregard of the standard (HTML/XHTML)

ID	Tag / attribute as a cause	Type of cause	Correctable or not	Rule set
76	<del>	Disregard of the standard	Automatically correctable	x
99	align attribute value of <caption> tag (left and right)	Disregard of the standard	Automatically correctable	x
88	URL with img or a	Disregard of the standard	Automatically correctable	x
34	type attribute of <object> tag	Disregard of the standard	Automatically correctable	x
90	<col>	Disregard of the standard	Manually correctable	x
68	<object>	Disregard of the standard	Manually correctable	x
74	<table width = “0” >	Disregard of the standard	Automatically correctable	x

## (3) Unimplemented functions

These web interoperability discrepancies are problems caused by tags or attributes which are included in the standard of HTML or XHTML stipulated by W3C, but not supported by Firefox or IE. There were 5 web interoperability discrepancies of these types found. The Firefox browser is incapable of displaying a ruby directly above text as in IE. This is why we list a letter string surrounded by <ruby> is classified as an unimplemented function rather than as a function against the standard.

Table 5 Web interoperability discrepancies arising from unimplemented functions (HTML/XHTML)

ID	Tag / attribute as a cause	Type of cause	Correctable or not	Rule set
36	<ruby> <rp> <rb> <rt>	Not supported by FF yet.	Not correctable	x
65	<acronym>	Not supported by IE yet.	Ignorable	x
50	<col>	Not supported by FF yet.	Automatically correctable	x
49	<colgroup>	Not supported by FF yet.	Automatically correctable	x
48	<basefont>	Not supported by FF yet. (Discontinued in 4.0.1)	Manually correctable	x

#### (4) Difference in interpretation of the standard

There are two web interoperability discrepancies of these types. With regard to quotation marks for the <q> tag, an implementation of IE without quotation marks can be considered to go against the standard, as it has been stipulated that “the content of the Q element is rendered with delimiting quotation marks.” We classify it here, however, as there is no clear description of type of quotation marks.

Table 6 Web interoperability discrepancies arising from difference in interpretation of the standard (HTML/XHTML)

ID	Tag / attribute as a cause	Type of cause	Correctable or not	Rule set
66	<q>	Difference in interpretation of the standard (Difference in implementation)	Ignorable	x
72	Margin for <p> tag	Difference in interpretation of the standard (Difference in initial value)	Manually correctable	

#### (5) Others

The following table lists factors that can cause web interoperability discrepancy arising from historical backgrounds even though both IE and Firefox are implemented. Tags and attributes for these factors shouldn't be used.

Table 7 Other factors that can cause web interoperability discrepancies (HTML/XHTML)

ID	Tag / attribute as a cause	Type of cause	Correctable or not	Rule set
109	align attribute middle, texttop, absmiddle, absbottom, and baseline	NN's own attribute	Automatically correctable (De facto)	N/A
121	<listing>	Tag that was discontinued in HTML4.0	Ignorable (Implemented by any browser)	N/A
122	<plaintext>	Tag that was discontinued in HTML4.0	Ignorable (Implemented by any browser)	N/A
123	<xmp>	Tag that was discontinued in HTML4.0	Ignorable (Implemented by any browser)	N/A

### 2.1.2.2 CSS

There are 50 web interoperability discrepancies caused by CSS properties, etc. The number of discrepancies caused by the browser's proprietary functions is 32

#### (1) Browser's proprietary functions

Most proprietary functions of the browser's CSS are the proprietary properties of IE. There are two kinds of properties. The first is a property to be adopted in CSS3 under current plans, such as word-break. The second is a property completely specific to IE, such as the color designation for the scroll bar. Firefox (Mozilla-based), Opera, and Safari also have their own properties, but all of these properties are positioned as experimental and not recommended for use.

Table 8 Web interoperability discrepancies arising from the browser's proprietary functions (CSS)

ID	Property as a cause	Type of cause	Correctable or not	Rule set
1	background-position-x	IE's proprietary property	Manually correctable	x
2	background-position-y	IE's proprietary property	Manually correctable	x
3	behavior	IE's proprietary property	Manually correctable	x
4	filter	IE's proprietary property	Not correctable	x
5	ime-mode	IE's proprietary property	Not correctable	x
6	layout-grid	IE's proprietary property	Not correctable	x
7	layout-grid-char	IE's proprietary property	Not correctable	x
8	layout-grid-line	IE's proprietary property	Not correctable	x
9	layout-grid-mode	IE's proprietary property	Not correctable	x
10	layout-grid-type	IE's proprietary property	Not correctable	x
11	line-break	IE's proprietary property	Not correctable	x
14	ruby-align	IE's proprietary property	Ignorable	x
15	ruby-overhang	IE's proprietary property	Ignorable	x
16	ruby-position	IE's proprietary property	Not correctable	x
17	scrollbar-3dlight-color	IE's proprietary property	Ignorable	x
18	scrollbar-arrow-color	IE's proprietary property	Ignorable	x
19	scrollbar-base-color	IE's proprietary property	Ignorable	x
20	scrollbar-darkshadow-color	IE's proprietary property	Ignorable	x
21	scrollbar-face-color	IE's proprietary property	Ignorable	x
22	scrollbar-highlight-color	IE's proprietary property	Ignorable	x
23	scrollbar-shadow-color	IE's proprietary property	Ignorable	x
24	text-autospace	IE's proprietary property	Not correctable	x
25	text-justify	IE's proprietary property	Ignorable	x
64	text-overflow	IE's proprietary property	Ignorable	x
26	text-underline-position	IE's proprietary property	Manually correctable	x
27	word-break	IE's proprietary property	Not correctable	x
28	word-wrap	IE's proprietary property	Not correctable	x
29	writing-mode	IE's proprietary property	Not correctable	x

30	zoom	IE's proprietary property	Manually correctable	x
110	Property starting with -moz-	Mozilla's (Firefox's) proprietary property	Not correctable	x
162	Property starting with -o-	Opera's proprietary property	Not correctable	N/A
111	Property starting with -webkit-	WebKit's (Safari's) proprietary property	Not correctable	N/A

## (2) Implementation in disregard of the standard

There are 12 web interoperability discrepancies caused by implementation different from CSS1 or CSS2. In addition to implementations with specifications that appear to be implemented improperly, we can also find implementations that allow the grammatical errors allowing of the page authors.

Three properties, Nos. 130, 131, and 132, the so-called "CSS Hacks," are used to apply CSS only to specific browsers, such as IE 6.0. Each is used on a trial basis to allow browsing in various kinds of browsers equally. At this stage, therefore, we should evaluate them in a forward-looking manner.

Table 9 Web interoperability discrepancies arising from implementation in disregard of the standard (CSS)

ID	Property as a cause	Type of cause	Correctable or not	Rule set
82	background-attachment	Disregard of the standard	Automatically correctable	x
75	color property in <hr>	Disregard of the standard	Automatically correctable	x
70	text-align	Disregard of the standard	Manually correctable	x
92	Excess "{" contained in CSS	Disregard of the standard	Manually correctable	
83	float	Disregard of the standard	Manually correctable	
81	a: link, a: hover	Disregard of the standard	Manually correctable	
80	z-index	Disregard of the standard	Automatically correctable	x
100	"," after "{"	Disregard of the standard	Automatically correctable	x
130	"* html" at the line head of CSS	Disregard of the standard	Automatically correctable	x
131	"html" at the line head of CSS	Disregard of the standard	Automatically correctable	x
132	"_" immediately before CSS property	Disregard of the standard	Automatically correctable	x
63	cursor property	Disregard of the standard / IE's proprietary value	Automatically correctable	x

## (3) Unimplemented functions

There are 3 problems caused by the lack of the implementation of the functions stipulated by CSS1 or CSS2. All of these problems arise because IE does not support some of the properties and values specified as properties.

Problems arise if a user tries to use IE to browse a site prepared for Firefox with these properties. There are actually many cases, however, when these same properties somehow manage to find their way into a site prepared for IE. When such a site is displayed in Firefox, Firefox reflects these properties and the Firefox display differs from that of IE.

Table 10 Web interoperability discrepancies caused by a failure to implement functions (CSS)

ID	Property as a cause	Type of cause	Correctable or not	Rule set
79	font-size-adjust	Not yet supported by IE.	Manually correctable	x
78	position fixed	Not yet supported by IE.	Automatically correctable	x
77	text-decoration blink	Not yet supported by IE.	Not correctable	x

#### (4) Differences in interpretation of the standard

The following table shows the number of web interoperability discrepancies caused by an incomplete description of CSS1 or CSS2. This is a problem that arises when an initial value is not clearly shown in the standard. The initial value therefore varies from one browser to another.

Table 11 Web interoperability discrepancies caused by differences in interpretation of the standard (CSS)

ID	Property as a cause	Type of cause	Correctable or not	Rule set
73	vertical-align	Difference in interpretation of the standard (Difference in initial value)	Automatically correctable	x

#### (5) Others

The following two properties are the proprietary properties of IE. Recently, however, Firefox has begun to support them (starting from version 1.5). Plans are in place for their adoption in CSS3.

Table 12 Others (CSS)

ID	Property as a cause	Type of cause	Correctable or not	Rule set
12	overflow-x	Supported by FF 1.5	Manually correctable	x
13	overflow-y	Supported by FF 1.5	Manually correctable	x

### 2.1.2.3 HTML/XHTML and CSS

The 6 web interoperability discrepancies related to both HTML/XHTML and CSS are shown below.

#### (1) Implementation in disregard of the standard

The 4 web interoperability discrepancy caused by implementation ignoring the standard of HTML/XHTML and CSS are shown below.

Table 13 Web interoperability discrepancies arising from implementation in disregard of the standard (HTML/XHTML and CSS)

ID	Tag, attribute, etc. as a cause	Type of cause	Correctable or not	Rule set
89	"#" at the top of color description	Disregard of the standard	Automatically correctable	x
69	Overflow of block items	Disregard of the standard	Automatically correctable	x
51	padding and border	Disregard of the standard	Manually correctable	
93	Attributes of border and properties of border-width	Disregard of the standard	Automatically correctable	x

#### (2) Differences in interpretation of the standard

The two web interoperability discrepancies caused by differences in interpretation of the standard of HTML/XHTML and CSS are shown below. Nos. 67 and 71 are problems that arise when the settings are inconsistent and there is no clear description of which setting should be reflected.

Table 14 Web interoperability discrepancies caused by differences in interpretation of the standard (HTML/XHTML and CSS)

ID	Tag, attribute, etc. as a cause	Type of cause	Correctable or not	Rule set
67	Color name (gray)	Difference in implementation	Ignorable	x
71	nowrap attributes and width (px designation), or white-space:nowrap; and width (px designation)	Difference in interpretation of the standard	Automatically correctable	x

### 2.1.2.4 JavaScript/DOM

The 45 web interoperability discrepancies caused by Javascript or DOM are shown below.

#### (1) Browser's proprietary functions

Javascript is implemented by both IE and Firefox (Mozilla). Compatibility between them is insufficient, however, as these two browsers were developed independently. The following 24 web interoperability discrepancies are caused by the proprietary functions of IE or Firefox (Mozilla).

Table 15 Web interoperability discrepancies caused by the browser's proprietary functions (Javascript / DOM)

ID	Syntax, method, etc. as a cause	Type of cause	Correctable or not	Rule set
136	Element .innerText	Proprietary function of IE	Not correctable	x
134	Element .outerHTML	Proprietary function of IE	Not correctable	x
135	Element .outerText	Proprietary function of IE	Not correctable	x
165	attachEvent	Proprietary function of IE	Manually correctable	x
166	detachEvent	Proprietary function of IE	Manually correctable	x
163	window.event	Proprietary function of IE	Manually correctable	
164	event.offsetX, offsetY event .pageX pageY	Proprietary function of IE	Manually correctable	x
133	window.scroll	Proprietary function of IE	Not correctable	x
146	document.getSelection	Proprietary function of FF	Not correctable	x
138	navigator.javaEnabled	Proprietary function of FF	Not correctable	x
139	navigator.mimeTypes	Proprietary function of FF	Not correctable	x
140	navigator.plugins	Proprietary function of FF	Not correctable	x
147	arity property of Javascript object	Proprietary function of FF	Automatically correctable	x
148	caller property of Javascript object	Proprietary function of FF	Not correctable	x
151	captureEvents method of Javascript object	Proprietary function of FF	Not correctable	x
149	routeEvent property of Javascript object	Proprietary function of FF	Not correctable	x
142	screen.pixelDepth	Proprietary function of FF	Not correctable	x
150	setInterval	Proprietary function of FF	Not correctable	x
153	toSource	Proprietary function of FF	Not correctable	x
141	valueOf	Proprietary function of FF	Not correctable	x
152	watch/unwatch	Proprietary function of FF	Not correctable	x
145	window.find	Proprietary function of FF	Not correctable	x
144	window.innerHeight, window.innerWidth window.OuterHeight, window.OuterWidth	Proprietary function of FF	Not correctable	x
143	outerWidth and outerHeight of the 3rd argument (attribute) of window.open	Proprietary function of FF	Not correctable	x

## (2) Implementation in disregard of the standard

The following 11 web interoperability discrepancies are caused by implementation in disregard of the standard of ECMAScript or DOM. All of them are in IE. The browser tends to over-interpret the grammar.

Table 16 Web interoperability discrepancies arising from implementation in disregard of the standard (Javascript / DOM)

ID	Syntax, method, etc. as a cause	Type of cause	Correctable or not	Rule set
85	//-->	Disregard of the standard	Manually correctable	x
91	document.URL	Disregard of the standard	Manually correctable	x
102	name and id given to <form> and form element	Disregard of the standard	Manually correctable	
101	name and id given to <form> and form element	Disregard of the standard	Manually correctable	
87	Name of frame	Disregard of the standard	Manually correctable	
103	id given to <frame>	Disregard of the standard	Manually correctable	
105	id given to tag	Disregard of the standard	Manually correctable	
104	id given to tag	Disregard of the standard	Manually correctable	
84	parent.frames	Disregard of the standard	Automatically correctable	x
86	parent.frames	Disregard of the standard	Manually correctable	
106	submit function	Disregard of the standard	Manually correctable	x

## (3) Unimplemented functions

The following 9 web interoperability discrepancies are caused by the failure of the browser to implement certain functions now stipulated in DOM.

Table 17 Web interoperability discrepancies arising from a failure to implement functions (Javascript / DOM)

ID	Syntax, method, etc. as a cause	Type of cause	Correctable or not	Rule set
156	Element.addEventListener	Not yet supported by IE.	Not correctable	x
159	Element.style.borderColor, borderStyle, and borderWidth	Not yet supported by IE.	Not correctable	x
158	Element.type	Not yet supported by IE.	Not correctable	x
155	Element.getAttributeNode	Not yet supported by IE.	Not correctable	x
154	Element.hasAttribute	Not yet supported by IE.	Not correctable	x
160	Element.style.padding	Not yet supported by IE.	Not correctable	x
157	Element.removeEventListener	Not yet supported by IE.	Not correctable	x
161	Element.style.whiteSpace="pre"	Not yet supported by IE.	Not correctable	x
137	Element.height	Not yet supported by IE.	Not correctable	x

#### (4) Others

Though this is not an web interoperability discrepancy, in order to detect web-sites which have contents including a process to check what browser is used, we consider a method to check the browser using Javascript as the target of this survey.

Table 18 Browser check (JavaScript/DOM)

ID	Syntax, method, etc. as a cause	Type of cause	Correctable or not	Rule set
107	Browser discrimination	Browser discrimination	No need to correct	x

#### 2.1.2.5 Other proprietary browser functions

Some browsers support some scripts and functionality expansions other than HTML/XHTML, CSS, and Javascript / DOM. Representative examples are shown below. Cell phones and WebTV pages are not normally accessed from PC browsers, hence there is no problem.

Table 19 Other proprietary browser functions

ID	Function as a cause	Type of cause	Correctable or not	Rule set
94	HTML+TIME	Proprietary function of IE	Not correctable	x
97	Transition	Proprietary function of IE	Not correctable	x
108	JScript	Proprietary function of IE	Manually correctable	x
95	TDC (Tabular Data Control) (Data Binding)	Proprietary function of IE	Not correctable	x
98	VBScript	Proprietary function of IE	Not correctable	x
96	VML (Vector Markup Language)	Proprietary function of IE	Not correctable	x
168	Proprietary tag / attribute of EZweb	HDML	Not correctable	N/A
170	Proprietary tag / attribute of HTML+'	HTML+	Not correctable	N/A
167	Proprietary tag / attribute of i-mode	i-mode extended HTML	Not correctable	N/A
171	IBM WebExplorer's proprietary tag / attribute	IBM WebExplorer	Not correctable	N/A
172	Proprietary tag / attribute of ISO/IEC 15445 Preparation	ISO/IEC 15445 Preparation	Not correctable	N/A
169	Proprietary tag / attribute of WebTV	WebTV	Not correctable	N/A



## Chapter 3 Research on actual content conditions responsible for browser non-compatibility

### 3.1 Systematic arrangement of rule sets to collect non-compatible content automatically

#### 3.1.1 Research procedure

Based on the accomplishments of TouchUp scripts collected in 2.1, we systematically arranged the rule sets used to collect non-compatible content automatically. We also prepared modules for determining non-compatibilities (hereinafter referred to simply as “Modules”) picked out content to be corrected based on universal rules, made the rules universal, and arranged the rules systematically.

#### 3.1.2 Preparation of modules for determining non-compatibility (Modules)

In line with the specifications for crawling tools, we reimplemented rule sets as Modules. As the language implemented in the crawling tools, we adopted JavaScript, i.e., essentially the same language as used for TouchUp script. The Modules took the form of dedicated modules to efficiently detect the presence or absence of web interoperability discrepancies without making any modifications.

##### 3.1.2.1 Examples of Modules

From among the Modules prepared, typical ones are shown below.

###### (1) Detection of CSS properties (IS1: background-position-x)

We obtain an external style sheet and a style sheet described in the header as the text and then check the presence or absence of properties in regular expressions. Given that Firefox parses CSS while reading in pages, we should note that it isn't possible to detect nonstandard properties by simply accessing `document.StyleSheets.cssRules`. `LoadTextFile` is a function implemented in crawling tools to deal with this problem.

```

(
function() {
  try {
    var i, str;
    function checkProp(str) {
      return str.match(/background-position-x(¥s)*:/i);
    }
    var sheets = document.styleSheets;
    for (i = 0; i < sheets.length; i++) {
      if (sheets[i].href == document.URL) continue;
      str = tuwutil.loadTextFile(sheets[i].href);
      if (checkProp(str) != null) return "NG";
    }
    var elms = document.getElementsByTagName("style");
    for (i = 0; i < elms.length; i++) {
      str = elms[i].innerHTML;
      if (checkProp(str) != null) return "NG";
    }
    return "OK";
  } catch (ex) {
    return ex;
  }
}
)();

```

## (2) Detection of HTML elements (ID31:bgsound)

Unlike CSS in (1) above, Firefox does not delete HTML as a result of parsing even if HTML is a nonstandard element. Thus, we can make determinations simply by using the `getElementsByTagName` method and seeing the number of appearances of the element.

```

(
function() {
  try {
    var elms = document.getElementsByTagName("bgsound");
    if (elms.length != 0) return "NG";
    return "OK";
  } catch (ex) {
    return "EXCEPTION";
  }
}
)();

```

## (3) Detection of attributes contained in HTML elements (ID39: contenteditable)

As in (2) above, we know that parsing will not delete nonstandard attributes and attribute values here, as well. We can make determinations by accessing all elements by `document.body.getElementsByTagName("*")` and checking for attributes.

```
(
function() {
  try {
    var elms = document.getElementsByTagName("*");
    for (var i = 0; i < elms.length; i++) {
      if (elms[i].hasAttribute("contenteditable")) return "NG";
    }
    return "OK";
  } catch (ex) {
    return "EXCEPTION";
  }
}
)();
```

#### (4) Detection of JavaScript (ID84: parent.frames)

We also detect JavaScript in regular expressions by reading it in as text, as in the case of CSS in (1) above. External scripts cannot be accessed by the regular method (DOM), hence we read them in using the loadTextFile function.

```
(
function() {
  try {
    var elms = document.getElementsByTagName("script");
    for (var i = 1; i < elms.length; i++) {
      var str = (elms[i].src) ? tuutil.loadTextFile(elms[i].src) : elms[i].innerHTML;
      if (str.match(/parent.frames?([0-9]+?)/i) != null) return "NG";
    }
    return "OK";
  } catch (ex) {
    return "EXCEPTION";
  }
}
)();
```

### 3.1.3 General-purpose and systematic arrangement

While proceeding with the work on the general-purpose and systematic arrangement, we encountered several items that couldn't be included in the Modules. The following lists those items and the various reasons that led to their occurrence.

#### 3.1.3.1 Items related to CSS

Among the items related to CSS listed in the list of web interoperability discrepancies, we encountered several problems that couldn't be judged without actually seeing the page in question. The following shows items that we decided to exclude from the rule sets because of difficulties in collecting them automatically.

Table 20 Excluded items related to CSS

ID	Title	Reason
51	(112) padding, border	This problem is notorious among web developers. The item is also corrected, in some cases, in the so-called “box model hack” or CSS exclusive to Internet Explorer. Thus, we expect it to be difficult to detect universally. Our researchers have to check them individually by directly viewing the page in question.  (Reference) BoxModelHack <a href="http://css-discuss.incutio.com/?page=BoxModelHack">http://css-discuss.incutio.com/?page=BoxModelHack</a>
70	(91) text-align property	The text-align property which causes problems is also used for original application of centering inline elements. Therefore, they cannot be clearly discriminated during automatic collection.
72	(57) <p> margin	This item affects actual display in some cases, but not in others. Therefore, universal detection is difficult. This is a problem dependent upon page markup and CSS.
83	(93) float	Same as the above
92	(92) Excess “}” included in CSS	Same as the above

### 3.1.3.2 Items related to JavaScript

Among the items related to JavaScript, our researchers have to open pages and operate them to check whether the problems caused by IDs or usernames affect the actual performance. We therefore decided that it would be difficult to detect these types of items universally. The following table lists the items we excluded from the rule sets because of difficulties in automatically collecting them.

Table 21 Excluded items related to JavaScript

ID	Title
87	(145) frame name
101	(143) name and id given to <form> and form elements
102	(144) name and id given to <form> and form elements
103	(146) id given to <frame>
104	(147) id given to tag
105	(148) id given to tag
163	(122) event

### 3.1.3.3 Specifications specific to browsers, and non-compatibility

Some have pointed out that the specifications implemented independently by certain browsers have been the main source of web interoperability discrepancies. The “de-facto standard” emerged as the main browsers adopted specifications in common and introduced them into widespread use.

Firefox is based on Mozilla. The source code for Mozilla was developed further by turning Netscape Communicator, originally a proprietary product, into an open source. In response to the web interoperability discrepancies produced as a result of the so-called “browser war,” developers now make it a policy to give up numerous specific implementations and conform to the standard. At the same time, however, support continues for some of the de-facto standards in order to ensure compatibility with existing content.

These de-facto standards cannot be considered desirable from the standpoint of the web standard, as the detailed specifications are sometimes unclear. In light of the purport of this research, however, we exclude from the research items with discrepancies in display or operation, and items that are now considered to be within the permissible range.

Table 22 Specifications specific to browsers, and web interoperability discrepancy

ID	Title	Reason
109	(extended values of align attribute)	This is a nonstandard attribute value implemented by Netscape Navigator by expanding HTML 2.0. This attribute does not affect the display, however, as it is also implemented by Internet Explorer, Firefox, etc.
112	(32) <nobr>	This is a nonstandard element implemented by Netscape Navigator by expanding HTML 2.0. This element does not affect the display, however, as it is also implemented by Internet Explorer, Firefox, etc. For reference, this element is also used on the top page of "Yahoo! JAPAN."
114	(27) <embed>	This is a nonstandard element implemented by Netscape Navigator by expanding HTML 3.0. Company:  This element does not affect the display, however, as it is also implemented by Internet Explorer, Firefox, etc. This element is used widely today in consideration of backward compatibility problems, although the use of standard object elements is recommended.
115	(33) <noembed>	This is a nonstandard element implemented by Netscape Navigator by expanding HTML 3.0. This element does not affect the display, however, as it is also implemented by Internet Explorer, Firefox, etc. This element is used together with embed shown above.
126	(5) border, bordercolor, frameborder, and framespacing attributes of <frameset> tag	Out of the attribute values of frameset element, border, bordercolor and frameborder are implemented by each browser.
127	(4) width, height and bordercolor attributes of <frame> tag	Out of the attribute values of frame element, bordercolor is implemented by each browser.

### 3.1.3.4 Experimental implementation with CSS by each browser's vendor

Each browser vendor may experimentally or independently implement CSS properties. These properties may be in the draft phase in W3C, etc., with the intention of stipulating a browser's user interface. The properties are generally implemented in line with the "vendor-specific extensions" ([http://www.w3.org/TR/CSS21/syn\\_data.html#q4](http://www.w3.org/TR/CSS21/syn_data.html#q4)) shown in the W3C's CSS2.1 specifications. Items enumerated in the list of web interoperability discrepancies are as shown below.

Table 23 Each vendor's extension properties

ID	Title
110	(87) Property beginning with -moz-
111	(89) Property beginning with -webkit-
162	(88) Property beginning with -o-

As stipulated in the specifications, web creators shouldn't use these properties. As things stand, cases where such properties are actually used on websites are expected to be extremely rare. Therefore, we studied the exclusion of these properties from the Modules. In this case, however, we expect that some of the extension properties implemented by Mozilla (Firefox) are likely to be used in consideration of browser compatibility. Therefore, we have decided to include them in the extension properties to be checked. The principal extension properties among them are shown below.

Table 24 Principal extension properties of Mozilla (Firefox)

Property name	Description
-moz-opacity	<p>The rate of transparency of elements is specified numerically. This property is officially supported in Firefox 1.5 as an opacity property, and it is parsed as opacity during page reading. Internet Explorer uses a filter with its own property (refer to ID4). When examining tutorials, etc. on the web, however, we find cases in which these three are written down altogether. This ensures that all will meet the requirements of the principal browsers and remain compatible. This is a prior implementation of CSS3.</p> <p><a href="http://developer.mozilla.org/en/docs/CSS:moz-opacity">http://developer.mozilla.org/en/docs/CSS:moz-opacity</a></p> <p><a href="http://www.w3.org/TR/css3-color/#transparency">http://www.w3.org/TR/css3-color/#transparency</a></p>
-moz-box-sizing	<p>In older versions of Internet Explorer and in the compatible mode, the implementation of the box model doesn't conform with the standard and the handling of border/padding and width/height differs from that of other browsers (refer to ID51). In Firefox, this property realizes a display compatible with IE. This is a prior implementation of CSS3.</p> <p><a href="http://developer.mozilla.org/en/docs/CSS:moz-box-sizing">http://developer.mozilla.org/en/docs/CSS:moz-box-sizing</a></p> <p><a href="http://www.w3.org/TR/css3-ui/#box-sizing">http://www.w3.org/TR/css3-ui/#box-sizing</a></p>

A list of extension properties of Mozilla can be confirmed in Mozilla Developer Center and XULPlanet.

<http://developer.mozilla.org/en/docs/CSS Reference: Mozilla Extensions>

[http://www.xulplanet.com/references/elemref/ref\\_StyleProperties.html](http://www.xulplanet.com/references/elemref/ref_StyleProperties.html)

No collective documentation on the extension properties for Webkit (Safari) and Opera has been produced as of this writing. While this makes it impossible to grasp all of the properties for these browsers, we can list several below as examples.

Table 25 Principal extension properties for Webkit and Opera

Property name	Description
-webkit-border-radius	This is a prior implementation of border-radius in the CSS3 specifications. It is a property in which frame border corners are rounded off. In the case of Mozilla, it is implemented as -moz-border-radius.
-o-link -o-link-source	According to the list of Opera specifications, these extension properties implemented in a manner that lets content creators include links and images in XML documents. <a href="http://www.opera.com/docs/specs/#xml-css-link">http://www.opera.com/docs/specs/#xml-css-link</a>

### 3.1.3.5 Discontinued elements

The following elements were discontinued in HTML 4.0 and are not recommended for use. All of them, however, are still implemented by principal browsers, hence they have no sources of web interoperability discrepancy. As a result of a study, therefore, they were excluded from the elements to be checked.

Table 26 Discontinued elements

ID	Title	Reason
121	(53) <listing>	The content is displayed in equimultiple fonts. As a substitute, pre element is recommended.
122	(54) <plaintext>	Same as the above
123	(55) <xmp>	Same as the above

### 3.1.3.6 Other excluded items

Besides the items explained above, the following table lists the items excluded from the Modules, and the reasons for their exclusion.

Table 27 Other excluded items

ID	Title	Reason
120	(34) <nolayer>	This is an element specific to Netscape Navigator, but it is hardly used at present. Given that this element is used in combination with the layer element (ID118) and ilayer (ID119), we expect that it can be covered by checking such combined use.
125	(38) marginheight and arginwidth attributes of <body> tag	These elements are integrated with ID45.
171	(167) Tags and attributes specific to HTML+  Chapter 1 (<abbrev>, <abbract>, <added>, <byline>, <changed>, etc.)	Although these elements were developed by W3C, they were not standardized. They are only used in extremely rare cases.  <a href="http://www.w3.org/MarkUp/htmlplus_paper/htmlplus.html">http://www.w3.org/MarkUp/htmlplus_paper/htmlplus.html</a>
173	(170) Tags and attributes specific to ISO/IEC 15445 Preparation (<div1>, <div2>, <div3>, <div4>, <div5>, <div6>)	Although these elements are the standard, they are a subset of W3C HTML. They are only used in extremely rare cases.  <a href="https://www.cs.tcd.ie/15445/15445.html">https://www.cs.tcd.ie/15445/15445.html</a>

### 3.1.3.7 Other implementations specific to vendors (Reference)

Customized HTML specifications were developed for the Internet terminal “WebTV” (current MSN TV) (used through connections with home TVs) platform that came into wide use in the latter half of the 1990s, as well as in the “IBM WebExplorer” browser developed for OS/2 during the same years. We exclude these specifications as items to check, however, as they are now very unlikely to be used, especially in domestic websites.

Further, NTT DoCoMo and KDDI each provide specifications (e.g., extension HTML and “HDML,” respectively) to create sites exclusively for their own mobile phone devices. We also exclude these, as they were unlikely to be encountered in the websites crawled.

### 3.1.4 Study of crawling results

A total of 75 modules for checking (Modules) were prepared and used for the crawling work. The Modules prepared here were used for the first phase of the crawling work. After the crawling work was completed, we studied the results of the crawling and corrected the scripts for the modules that ran into problems. Some of the Modules were also clearly unnecessary for the objective of the research and were thus excluded.

Table 28 Study of crawling results

ID	Description
34	This item was considered unacceptable and rejected when no type attribute was specified the object element. The display was unaffected, however, if an embed element was included in object element. The conditions for that were therefore also added.
45, 125	Concerning a margin for the body element, The proprietary attributes of Internet Explorer and Netscape Navigator were judged independently. If the attributes were written down together, however, we deemed this a measure for compatibility and integrated the rule sets.
69	Parent-child element sizes were compared using computedStyle. If the browser window was reduced (a state in which the horizontal scroll bar appears), the width of a child element became larger than the width of a body. Some of the results were therefore unacceptable. If the parent element was a body, the script was corrected in such a way that determinations would be excluded.
42, 71, 80, 88, 89, 93	Exceptions were seen due to description errors in the script, such as omission of processing and variables being not yet defined in a case where attributes could not be obtained.
109	The standard align="center" was erroneously regarded as an item to be determined. As a result of a study that was subsequently conducted, we found that the nonstandard attribute value included in the items to be determined was a de facto standard. Therefore, we judged this rule set to be superfluous and decided to exclude it from the items to be researched (refer to 3.1.3.3).
112	This is was determined to be a proprietary element of Netscape Navigator. Because it was also supported by Internet Explorer, Firefox, etc., we excluded it from the items to be researched (refer to 3.1.3.3).
121, 122, 123	These were discontinued attributes. None were sources of non-compatibility, however, so we decided to exclude them from the items to be researched (refer to 3.1.3.5).
126, 128	These elements were determined to be proprietary attributes of Internet Explorer and Netscape Navigator. But because they were de facto standards (as in 109 shown above), we decided that they would not affect compatibility and excluded them from the items to be research.

## 3.2 Enhancement of rule sets to collect non-compatible content automatically

### 3.2.1 Research procedure

After carrying out the web crawling work, we fed a result with the discovery of a new problematic website back to the TouchUpWeb system and defined a new rule as a rule to be added newly to enhance the rule sets.

We took the following procedures to define and implement the rule sets to be enhanced

- For content newly discovered as non-compatible content to which existing scripts can be applied, we provided feedback to the scripts in a manner that made it possible to apply the scripts to newly discovered content as well.
- With regard to the existing scripts found to have similar problems, we added new scripts to the TouchUpWeb system while re-implementing existing scripts as modules to check the crawling tools (Modules).
- In addition, TouchUp scripts contributed by third parties were newly included among those to be incorporated into crawling tools to enhance the rule sets.

A new rule set was prepared for the second phase of crawling work. As for the definition and implementation of rule sets to be enhanced, the procedures taken included the following items.

- We confirmed the specifications for crawling and prepared Modules for the items that had been kept on hold during the first phase of crawling work, especially the reading of external style sheets and external script.
- We systematically arranged the items newly added to the list of web interoperability discrepancies and turned them into rules.

In addition to the above, we decided that rule sets for the first phase of crawling work, corrected in 3.4, would also be used for automatic collection again in the second phase (11 cases).

### 3.2.2 List of added Modules

A total of 75 Modules were prepared and used for the crawling work.

### 3.2.3 Study of crawling results

The second phase of crawling work covered 10,000 pages, and we studied the results once the crawling was completed.

Table 29 Study of crawling results (upon completion of 10,000 pages)

ID	Description
42, 45, 69, 80, 90, 93, 106, 126, and 127	<p>Exceptions were found in these rule sets. We found, through examination, that the exceptions were caused by one or more of the following causes.</p> <ol style="list-style-type: none"> <li>1. There was no body element on the page in which the frame was used.</li> <li>2. The page was redirected and an HTML file could not be obtained.</li> <li>3. The document.body wasn't read in when a page-reading-completion event took place. (This phenomenon didn't arise during manual execution. In automatic execution over long hours, we confirmed that the number of occurrences tended to increase. The cause of this tendency wasn't clear.)</li> </ol> <p>For this reason, we tried to avoid exceptions by skipping the check when document.body was empty.</p>

The results were studied again when all of the crawling work was completed again.

Table 30 Study of crawling results (When all of the pages had been crawled)

ID	Description
69	<p>This rule set was corrected after the first phase of crawling work was completed. The determination conditions were not yet sufficient at this time, however. Most notably, there were no conditions to adjust for the special table cell display or to carefully determine the size. And because erroneous determinations were also made on pages lacking tables, other factors were also conceivable. Accordingly, we concluded that it was virtually difficult to make correct determinations automatically with rule sets. We therefore tabled the adoption of research results.</p>
12, 13	<p>Prior research on the overflow-x/y properties showed that parts of them were not yet supported by Firefox. In a subsequent reconfirmation, however, we found that the overflow-x/y properties were actually supported. We therefore decided to exclude them from the research.</p>

### 3.2.4 Web interoperability discrepancy identified after research

The following items should have been included in the items for enhancement of rule sets. Because they were identified after the crawling was started, however, they were not included in this research.

Table 31 Web interoperability discrepancy identified after research

Title	Description
CSS extension property of Excel	This is a prefix of mso- contained in HTML files produced by Microsoft Excel. The details are unclear, as the specifications have not been documented.
Canvas element	This is the Web Hypertext Application Technology Working Group (WHATWG) standard supported by Firefox and Safari. This is included in the Web Applications 1.0 draft. The standard isn't yet introduced into practical as of this writing, however, so we have concluded that it can be ignored for the time being.  <a href="http://www.whatwg.org/specs/web-apps/current-work/#the-canvas">http://www.whatwg.org/specs/web-apps/current-work/#the-canvas</a>  <a href="http://developer.mozilla.org/ja/docs/Canvas_tutorial">http://developer.mozilla.org/ja/docs/Canvas_tutorial</a>
Conditional comment	This is a conditional branching comment that can be interpreted only by Internet Explorer. Many now recognized this comment, as the development team of IE7 recommended its use in place of "CSS hack" on blogs. Its application can be helpful to avoid bugs in IE. This comment in itself shouldn't produce any discrepancies, but it seems worthwhile to recognize the tendency of content creators to use it.  <a href="http://msdn.microsoft.com/workshop/author/dhtml/overview/ccomment_oww.asp">http://msdn.microsoft.com/workshop/author/dhtml/overview/ccomment_oww.asp</a>  <a href="http://blogs.msdn.com/ie/archive/2005/10/12/480242.aspx">http://blogs.msdn.com/ie/archive/2005/10/12/480242.aspx</a>  <a href="http://www.quirksmode.org/css/condcom.html">http://www.quirksmode.org/css/condcom.html</a>
Different designation of width/height for nested embed element and object element	This seems to be a bug in IBM Homepage Builder. Mozilla Japan received several reports of discrepancies in the display size of Flash movies in IE and Firefox.

In addition to the above, the research items will increase further if we list the small bugs that each browser faces, the differences in rendering behavior (in the standard-compatible mode and compatible mode), the status of CSS support, etc. Many documents and articles dealing with these types of problems have already been disseminated on the web. Many of these items are expected to require direct visual examination by our researchers, as described in 3.1.3.1 and 3.1.3.2. We should note that due mainly to time constraints, there is a limit on the list of web interoperability discrepancies to be researched at this point.

#### 3.2.4.1 Reference materials

Several reference materials are listed below as examples. Many other materials on "CSS hack" are available on the web or in books. We recommend that interested parties browse through them.

- Position Is Everything  
<http://www.positioniseverything.net/>

- QuirksMode.org: Bug Report  
<http://www.quirksmode.org/bugreports/>
- QuirksMode.org: Quirks mode and strict mode  
<http://www.quirksmode.org/css/quirksmode.html>
- Adobe: CSS Advisor beta  
<http://www.adobe.com/cfusion/communityengine/index.cfm?event=homepage&productId=1>
- Learn Web Standards: Browser Support  
[http://www.westciv.com/style\\_master/academy/browser\\_support/](http://www.westciv.com/style_master/academy/browser_support/)

### 3.2.5 Comments from the person in charge of research

Work on preparation of Modules was relatively simple. We should consider, however, the unexpected exceptions and rejections that have resulted from the insufficiency of prior testing. If similar research is performed in the future, I hope that the researchers will use the Modules from our current study for guidance and reference. When preparing new Modules or Modules for more complicated conditions, it will be safe and passable to pick out actual web pages at random as test cases in order to confirm that there are no problems in advance.

As products of various types emerge, the market for web browsers and authoring tools continues to grow at an accelerating pace. It seems important, therefore, to perform similar research periodically in the future in order to accumulate information, release it into the public domain, and enrich the TouchUpWeb script.

## 3.3 Development of crawling tools for automatic collection

### 3.3.1 Work procedure

Based on the existing web crawling tools, we developed automatic check tools with an added module for determining non-compatibility. We designed this module for the check tools for subsequent upgrades and expansions with the plug-in system, as we expected it to be added later, as described above.

### 3.3.2 Course of development

The first step in the development of the check tools was to select one of the existing crawling tools. We studied several Java-based crawling tools and selected JSpider, a tool with sufficient granularity to enable customization within the period. When selected, JSpider had a plug-in mechanism and was easy to handle. The application had about 20,000 steps (i.e., the scale of the source code was moderate, and it already had an external engine such as Velocity as a standard plug-in). At the initial stage, therefore, we studied the addition of a plug-in with necessary functions based on JSpider.

At the beginning of the development, the specifications for the module to determine non-compatibility were

partially fluid and unclear. Ultimately, however, we decided to adopt JavaScript, as the use of JavaScript would enable us to easily divert the achievements of the TouchUpWeb project and handle page data as DOM. Therefore, the check tool body would need to be equipped with an execution environment for JavaScript as the module for determining non-compatibility, and it would also need to allow continuous execution. To realize these requirements, we initially started with JSpider. After several prototypes were prepared and many studies were completed, however, we decided to provide the check tool as an extended capability of Firefox. It was difficult for us, however, to reach a consensus on how to convert HTML, a description that had not been normalized, into DOM. An outline of the prototypes we prepared is shown below.

Table 32 Comparison of frameworks

Item No.	Prototype	Remarks
1	JSpider + Rhino	The development language was Java. This prototype customized the open source crawler application JSpider and incorporated the open source JavaScript engine Rhino. In the normal XML parser, however, we were not able to convert HTML into DOM.
2	JSpider + Rhino + Jericho	To solve the above problem, this prototype incorporated an open source Jericho parser as a parser capable of handling HTML. But since Jericho used its own API to parse HTML, this prototype needed to acquire knowledge aside from the knowledge of standard DOM. Therefore, the adoption was shelved.
3	XULRunner	The development language was JavaScript + XUL. This prototype adopted a method that used XULRunner, an open-source rich-client-execution environment, and obtained DOM documents from browser objects for XULRunner. XULRunner was equipped with a rendering engine common to Firefox 1.5, hence we expected the prototype to obtain the same DOM objects as those produced by Firefox. While we did succeed in obtaining the DOM objects we expected to obtain, we encountered several unclear points with the access to CSS attributes. Therefore, we went on to study the following idea.
4	Extended capability of Firefox	The development language was JavaScript + XUL. A capability implemented in the XULRunner version was reconfigured as an extended capability of Firefox. As a result, we were able to confirm that the operation was basically the same as that of XULRunner. In addition, this version was also expected to facilitate the use of past accomplishments. Therefore, we decided to adopt this version as a check tool.

In addition to the main check tool described above, we prepared and implemented a script for obtaining from the Yahoo! Japan directory basic data to be used in selecting the websites to be researched. This was based on the use of the web service for directory access provided by Yahoo! Japan.

We also thought, in light of the constraints to the research schedule, that we would have too little time for research if we adopted a method that required us to directly access the pages of 200,000 sites (or thereabouts) for research every time a new module to determine non-compatibility was added. If, for example, the application required 10 seconds to access one site and there were no interruptions or delays in operations, 23 days of round-the-clock operation would be required to complete the task. When we considered the periods required to develop the tools, to develop the modules for determining non-compatibility (Modules), and to add and renew the Modules, we judged that it would be difficult for us to complete the research within the period allotted. Therefore, we adopted a policy of storing the web page data to be researched on a local disk first, then checking the local data later. For this reason, we decided to execute the processing on local files, as in the case of ordinary URLs. In connection with this, we also

prepared a script for download. This judgment seemed to be correct, although we experienced many problems once we actually started the research processes. It seemed that in the fight against time, the data downloaded locally shortened the processing time.

Further, in order to increase the efficiency of counting, we decided to store the research results in RDB and subject them to counting by SQL. To achieve this purpose, we designed a table for counting and prepared a conversion script.

### 3.3.3 Script for collection of basic data to be processed

#### 3.3.3.1 Objective

The object of this research was “200,000 sites registered in the Yahoo! Japan directory.” As our first step, therefore, we developed a script for collecting URLs registered in Yahoo! Japan.

#### 3.3.3.2 API used

In collecting basic data to be processed, we used the Yahoo! category web service released to the public by Yahoo! Japan. (Reference: <http://developer.yahoo.co.jp/category/>) As the use of this service was limited to 50,000 times per 24 hours, we had to take several days to obtain the information.

### 3.3.4 Download script

#### 3.3.4.1 Objective

To increase the efficiency of the crawling work within the period of research, we developed a policy of downloading the web pages to be researched and checking the local files in advance. We therefore prepared a download script for this purpose.

#### 3.3.4.2 Software used

A Wget command attached to Linux was used to carry out the download work. The Wget command is a widely used download tool. While the Wget command is unusable, as is, for the research we are planning, it is equipped with a function for downloading web pages locally. Users can designate various download parameters, such as the range and file types for the downloads. This time we designated the options shown in Table 33.

Table 33 Options of "wget"

Option	Description
-p	With this option, the Wget downloads all of the files necessary to properly display the HTML pages to be researched. This includes images, voice, style sheets to be referenced, and so forth.
-k	After the download is completed, this option converts links within a document into those suitable for local browsing. This affects all portions linked to external content, such as hyperlinks, embedded images, links to style sheets, and hyperlinks to content other than HTML content.
--follow-tags	In recursive download, this option designates tags to be considered. The option is also effective for use in conjunction with -p.

#### 3.3.4.3 Web service used

When downloading, we checked the applicable links and decided that if there was any problem in the links, we would exclude them from links to be checked. For this purpose, several services were studied. As a result, we decided to incorporate Dr. Web's link check service, a service capable of real-time checking, into the script. We explained the purpose of this research to Doctor Web, Ltd. and they consented to our plan. We would like to express our deep gratitude to them for their understanding and support.

### 3.3.5 Tools for checking

#### 3.3.5.1 Architecture

The check tool is an extended capability of Firefox, hence it conforms with the Firefox specifications. That is, the tool is composed of JavaScript (in charge of control), XUL (in charge of the user interface), and XPCOM, a component to provide the basic functions.

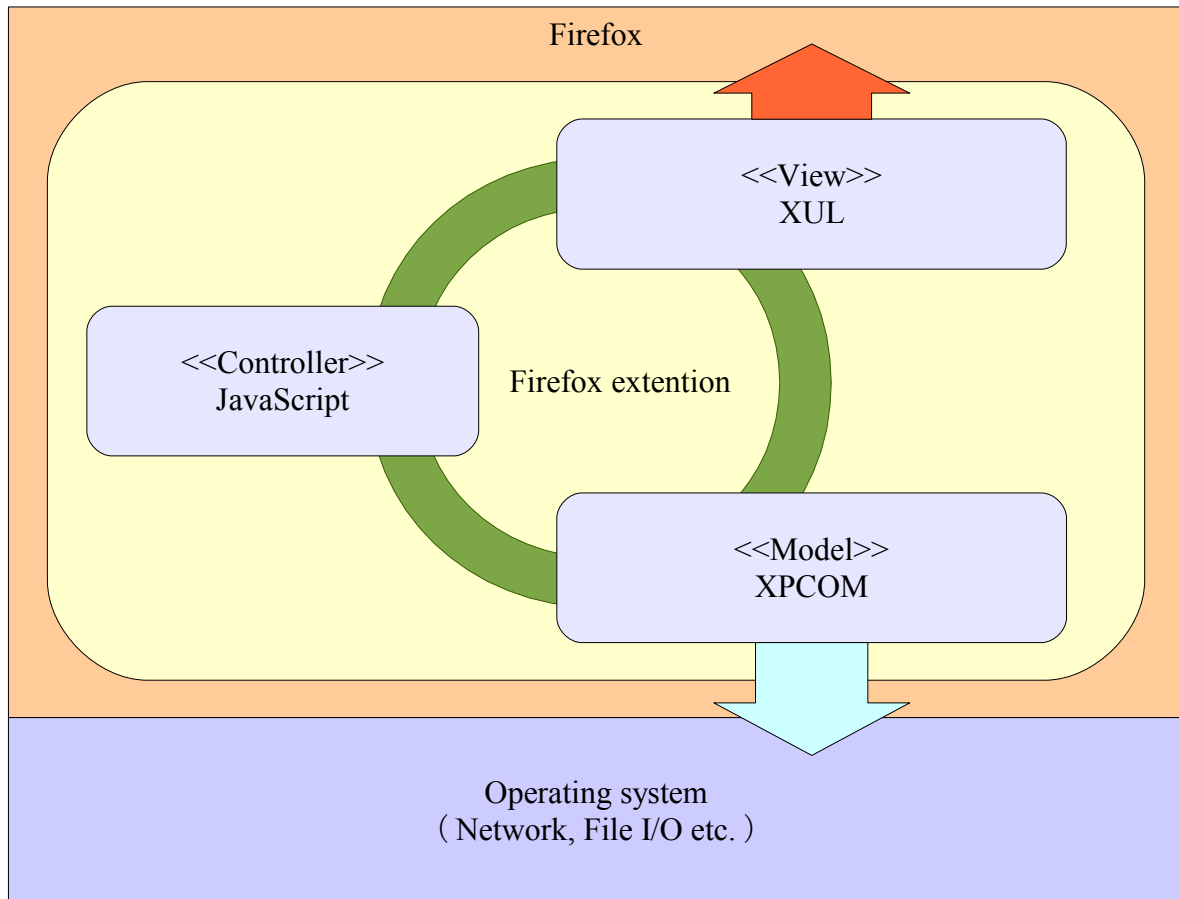


Fig. 3 An overview of Firefox extension

This tool took a form with a sandbox designed for the implementation of a module for determining non-compatibility safely, with necessary API provided to the sandbox. This tool also provides support functions, such as a log function, console function, and source code display function, to facilitate the development of the module for determining web interoperability discrepancy.

#### 3.3.5.2 Storage directory for the module for determining web interoperability discrepancy

This tool prepares a directory of the type shown below in a data directory for each user in Firefox. And under this directory, the tool stores a registered module for determining non-compatibility. The path is written based on the Linux environment, but an equivalent directory exists even in Windows and Mac OS X. Detailed descriptions are avoided, but an equivalent directory exists in %HOME%\Application Data in Windows and in \${HOME}/Library/Application Support in Mac OS X (the directory compositions partially

differ). For reference, xxxxxxxx in the table will have a different value for each user.

Table 34 Storage directory for the module for determining discrepancy

Name of path	Description
<code>\${HOME}/.mozilla/firefox/xxxxxxx/tuwcheck</code>	Registered Modules to determine non-compatibility are stored here. If a directory does not exist, it will be prepared.

### 3.3.5.3 Operating conditions

This tool was developed in the following environment on the assumption that it will operate in major Linux distributions. The tool has been confirmed to operate even in Windows and Mac OS X, but not with any special guarantee. In this research we used the tool with a Linux server setup.

- Fedora Core 4 Linux
- Mozilla Firefox 2.0

## 3.4 Data collection by web crawling

### 3.4.1 Work procedure

Exhaustive research on the actual conditions of the Internet was carried out with a main focus on content in Japanese. We regarded the entire Yahoo! Japan directory as the target for the research, and applied check tools to about 200,000 sites registered in the directory. Because of time constraints, we only researched the top page of each site.

In the first stages of this research we determined the data-collection schedules and the order of sites to be crawled, then designed the formats for the data to be collected. Next, we carried out the non-compatibility check using the automatic check tools described above. The URLs to be checked, check rules, and check results were put into order and organized.

For information, we analyzed and studied the data-collection results by feeding the results back to the aforementioned steps of “classifying and systematically arranging the collected TouchUp scripts, and analyzing the non-compatibility factors.” To put it concretely, we picked out the newly discovered problems from the collected data, classified them, broke them down into patterns, picked out principal factors, and systematically arranged typical methods for improvement. By systematically arranging the non-compatible conditions according to four levels—“ignorable,” “modifiable (automatically modifiable),” “modifiable (needing to be checked manually), and “unmodifiable”—we could analyze the tendencies of the non-compatible conditions and measures for dealing with them.

In addition to the above, we analyzed the tendencies of all of the data crawled and studied countermeasures based on the tendencies of the problems that appeared, etc. (In specific fields, for example, many non-compatibility problems with respect to the use of specific authoring tools tend to arise. With this in mind, we

prepared recommendations and studied the parties for which those recommendations were intended.)

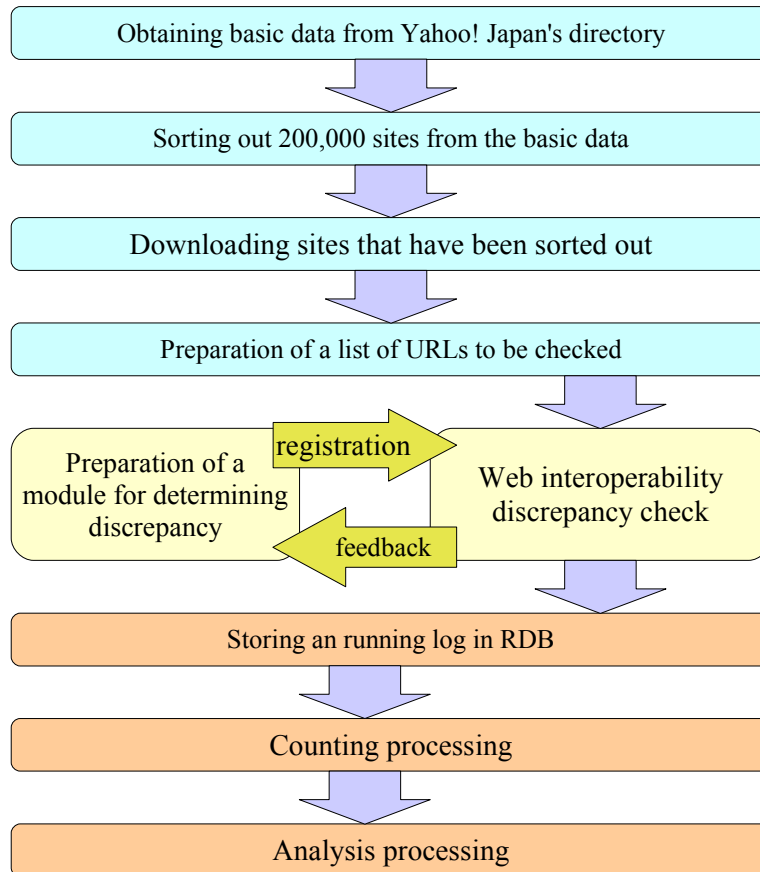


Fig. 4 Sequence of this exhaustive research on the Internet

### 3.4.2 Status of crawling implementation

#### 3.4.2.1 Basic data collection

##### (1) Outline of work

We sorted out the sites to be researched by first collecting the site URLs to be used as the basic data from the Yahoo! Japan directory and classifying Yahoo! Japan directory into the 14 major categories shown below. Some of the subordinate structures in the directory are shared by multiple categories with the use of aliases, hence the directory has a network structure rather than a simple tree structure. To collect information efficiently, it is important to exclude information duplicated by aliases. We prepared a script taking this point into consideration and then carried out the information-collection work.

The information was collected with the Yahoo! Japan web service released publicly. Access from each IP address is limited to 50,000 uses per 24 hours. When the upper limit is reached, the address must wait until the next 24 hour period to resume operation. When accessing via a proxy, it is therefore impossible to take a simple parallelization approach. Therefore, we increased the efficiency of information collection by accessing the service from different sites.

Yahoo!カテゴリ	サイトの登録
<u>エンターテインメント</u> 芸能人, 音楽, 映画, アニメ, 占い ...	<u>メディアとニュース</u> テレビ, ラジオ, 新聞, 雑誌 ...
<u>趣味とスポーツ</u> 車, スポーツ, 旅, アウトドア, ゲーム ...	<u>ビジネスと経済</u> ショッピング, B2B, 雇用, 金融 ...
<u>芸術と人文</u> 写真, デザイン, 演劇, 歴史, 文学 ...	<u>各種資料と情報源</u> 図書館, 辞書, 郵便, 電話番号 ...
<u>生活と文化</u> 住まい, 暮らし, 環境, グルメ, 結婚 ...	<u>コンピュータとインターネット</u> ホームページ, ハード, ソフト ...
<u>教育</u> 大学, 小中高, 資格, 専門学校 ...	<u>政治</u> 行政, 国会, 法, 税, 議員 ...
<u>健康と医学</u> 病院, 病気, 薬, 栄養, ダイエット ...	<u>自然科学と技術</u> 生物, 地球, 天文, 工学, 化学 ...
<u>社会科学</u> 言語, 経済学, 心理学, 社会学 ...	<u>地域情報</u> 都道府県, 世界の国と地域 ...

Fig. 5 Screen shot of Yahoo! Japan's top page

Table 35 Categories and access directories

Item No.	Name of category	Name of access directory
1	Entertainment	/Entertainment/
2	Hobbies and sports	/Recreation/
3	Arts and humanities	/Arts/
4	Life and culture	/Society_and_Culture/
5	Education	/Education/
6	Health and medical science	/Health/
7	Social science	/Social_Science/
8	Media and news	/News/
9	Business and economy	/Business_and_Economy/
10	Various kinds of materials and information sources	/Reference/
11	Computers and Internet	/Computers_and_Internet/
12	Politics	/Government/
13	Natural science and technology	/Science/
14	Regional information	/Regional/

## (2) Work environment

Two machines set up in different sites were used for carrying out the information-collection work, as shown in the following figure.

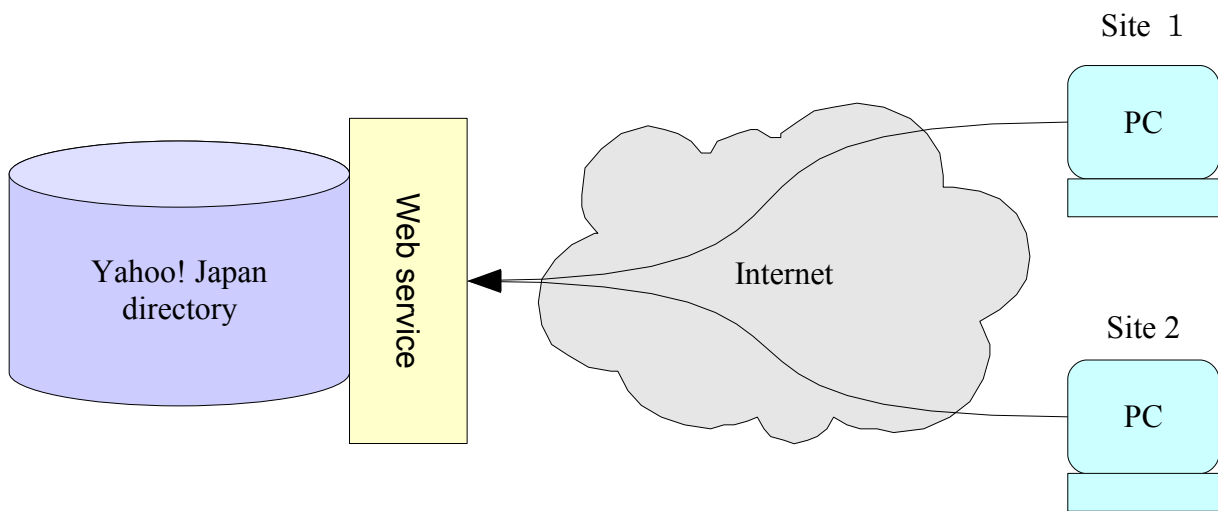


Fig. 6 Working environment

The following shows the specifications for the PCs used for the data collection. At the beginning we used PCs with only mediocre specifications, as we had yet to procure the high-end PCs that were to be used later. For information, the use of the web service has a limitation based on the number of accesses. Therefore, even the specifications shown below fully meet the work requirements. Shell scripts were used for the data collection, and each was implemented on the bash. The paths, etc. for the commands used in the script were partially adjusted for use with the different operating systems of the computers used.

Table 36 Specifications of PCs used in this research

Type of PC	Performance item	Performance value
PC1	CPU	Pentium III
	Memory	512 MB
	Disk	20 GB
	OS	Fedora Core 4
PC2	CPU	PowerPC 4
	Memory	512 MB
	Disk	40 GB
	OS	Max OS X

### (3) Work results

New sites are registered almost every day in the Yahoo! Japan directory, hence the number of registered sites is never static. For this reason, we discontinued the data correction work when the number of newly found sites fell below 100. Basic data was collected for a total of 268,567 sites. Although some margin of error is conceivable, this figure is thought to be an approximate value for the number of sites registered in the Yahoo! Japan directory when the data collection was completed. JP domains make up two thirds of the total, followed by COM, NET, ORG, and INFO. The number of countries represented, including the United States, is 91. This tells us that the JP domains alone are numerous enough to reach the 200,000 sites to be researched.

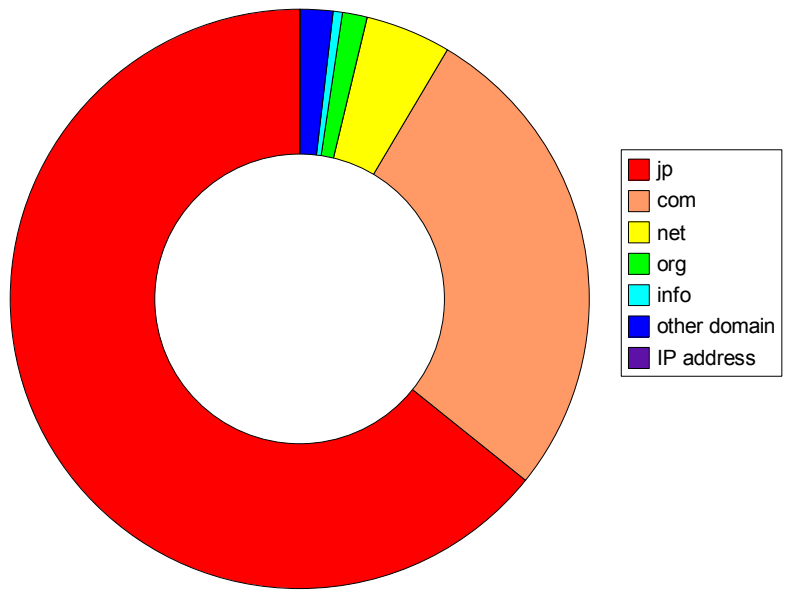


Fig. 7 Domain contents

Table 37 Domain types and the number of sites

Domain	Number of sites included
.JP	172,692
.COM	73,236
.NET	12,936
.ORG	3,642
.INFO	1,343
Others	4,666
IP address directly written	52
<b>Total</b>	<b>268,515</b>

Here, if we look at the rate of domains by organization within the JP domains, we obtain results as shown below. As expected, the rate for the CO.JP domains is the largest, followed by the rate for the NE.JP domains, the domains thought to be used mainly by the Internet service providers. Next came OR.JP, AC.JP, and GO.JP. The rate of NE.JP domains and OR.JP domains combined is comparable to the rate of CO.JP domains, hence we concluded that the number of personal websites included is almost equal to the number of company websites. Further, other JP domains make up nearly one-fourth of all JP domains, which means that a wide variety of domain names, including domains by region, etc., are used.

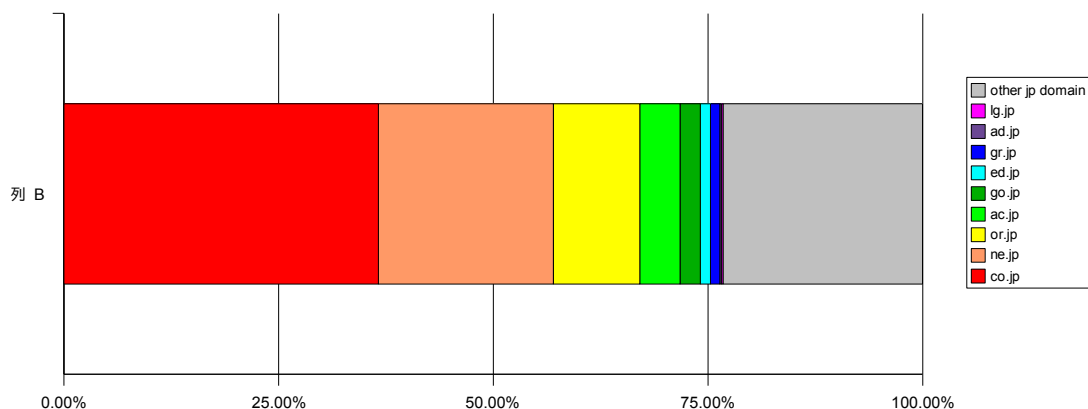


Fig. 8 Type of JP domains

Table 38 Type of JP domains and the number of sites

Domain	Number of sites included
.CO.JP	63,257
.NE.JP	35,153
.OR.JP	17,390
.AC.JP	8,097
.GO.JP	4,101
.ED.JP	2,039
.GR.JP	1,856
.AD.JP	355
.LG.JP	284
Other JP domains	40,160
<b>Total</b>	<b>172,692</b>

### 3.4.2.2 Sorting out the sites to be researched

#### (1) Outline of work

From among the 268,515 sites for basic data collected, 200,000 sites to be researched were sorted out. In sorting out the sites to be researched, the following policies were applied.

1. As the major premise, 200,000 sites should be sorted out as those to be researched.
2. Top priority should be assigned to the AC.JP, GO.JP, and CO.JP domains. To make up for any shortage, selections should subsequently be made from among the remaining JP domains, the EDU domains, the GOV domains, and the COM domains, in that order. In this case, however, sites meeting the following conditions should basically not be included.

3. Authentication may be required in some cases. Thus, https: should be excluded.
4. Host names starting with a number or IP address should be excluded.
5. Sites with suspicious keywords that seem to offend the public order and morals should be excluded. (xxx, porn, etc.)
6. Sites with URLs appended with a GET parameter should be excluded, as these are likely to be requests to call-up CGI forms or web programs.
7. URLs which are certain to be controlled by a web program (such as ,asp, ,jsp and ,php, etc.) should be excluded.

Furthermore, .COM domains to be added to make up for the shortage of JP domains sorted in the lexical order in the second token of domains, in order to represent all of the letters in the alphabet (from A to Z) as evenly as possible in the selections made. In this case, however, there were many domains starting with letters such as A and B, and very few domains starting with such letters as Q. In the case of the latter, other letters were allotted to make up for the shortage.

## (2) Work environment

This work was carried out on PCs which obtained basic data.. The sorting was done mechanically, with frequent use of shell scripts. As a result of this, sites not in conformity with the sorting conditions (for example, sites that seem to offend public order and morals) came to be included in the work operations subsequent to the download phase.

## (3) Work results

As a result of the sorting work, 200,000 sites were sorted out with the following details. Basically, almost all JP domains were included, insofar as possible. A total of 171,937 JP domain sites were selected, which means that 755 sites were excluded during the sorting.

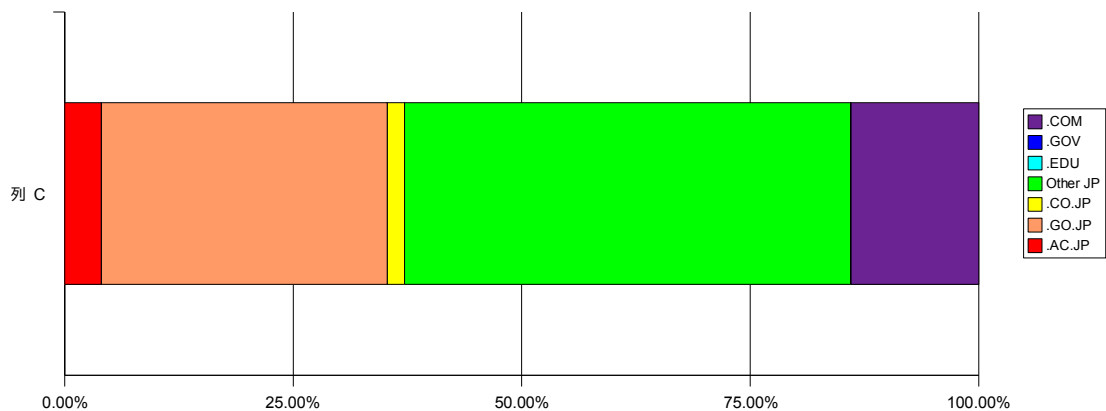


Fig. 9 Type of domains for sites sorted out

Table 39 Type of domains for sites sorted out

Domain	Number of sites included
.AC.JP	7,991
.GO.JP	62,575
.CO.JP	3,800
Other JP domains	97,571
.EDU	18
.GOV	8
.COM	28,037
Total	200,000

### 3.4.2.3 Downloading of, and preparation of a list of, sites to be researched

#### (1) Outline of work

Web pages in 200,000 sites sorted out as sites to be researched were downloaded to local discs. Due to time constraints, we examined only the front page of each site for this research. (In most cases, websites have a hierarchical structure linked from the front page of each site. We think that there are usually many more than 10 links on this front page. Therefore, if the number of hierarchies to be researched is increased by only one, for example, the processing time will increase to “(200,000 + Summation of the number of links of each page) \* Processing time per site”. In the case of 200,000 sites, this corresponds to a calculation dealing with several million pages. If we spend ten seconds to research one page, the processing can be expected to take more than one year. If the number of hierarchies to be researched continues to increase, the time required will increase commensurately.) We should also note that this research was intended to cover web pages, style sheets, and Javascript. Other resources, such as images, were basically not included in the content to be downloaded.

Sites to be researched were obtained from the directory of Yahoo! Japan. Therefore, we expected that some of the URLs would no longer be valid by the time we tried to visit them, for usual reasons such as site changes, extinctions, and so on. Invalid sites were detected only after we tried to actually visit them. We therefore assumed that if invalid sites were replaced and the newly selected sites were found to be invalid again, the replacement work would have to be repeated. This compelled us to discuss how to handle these types of sites, in consideration of the processing efficiency, etc. As a result, the team decided to handle and count these sites as invalid sites, and not to hunt for replacement sites.

Further, web pages were mechanically extracted from the downloaded data, and a list of URLs to be researched was prepared. This list was intended for use in automatic judgments with check tools.

## (2) Work results

The total amount of download reached about 4.5 GB. The number of sites that turned out to be invalid as a result of the download processing was 7,635, which accounted for 3.0% of all of the sites. The table below shows the main reasons explaining why these sites became invalid, according to our investigation. As we expected, many of the sites and pages were apparently nonexistent by the time we tried to access them.

Table 40 List of causes of invalid sites

Reason for becoming invalid	Number of sites
Error at the time of name solution by DNS	1,015
No information on route	6
Connection refusal	42
SSL connection failure	41
Authentication failure	1
The number of redirection was more than 20 times	3
Scheme not yet supported	2
Connection interruption	13
Status 400 (Bad Request)	23
Status 401 (Unauthorized)	10
Status 403 (Forbidden)	1,270
Status 404 (Not Found)	3,257
Status 405 (Method Not Allowed)	79
Status 406 (Not Acceptable)	7
Status 410 (Gone)	1
Status 500 (Internal Server Error)	1,071
Status 501 (Not Implemented)	3
Status 502 (Bad Gateway)	4
Status 503 (Service Unavailable)	13
Others / unclear	774

In addition, we detected several viruses in a series of virus checks. The following table lists the number of

viruses detected for each software used and the results of examination. As shown, Virus Buster from Trend Micro Incorporated detected the highest number. ClamAV, open source anti-virus application, was unable to detect any viruses at all. This does not necessarily mean that ClamAV is totally ineffective. Indeed, ClamAV has been effective in actual use for various purposes, such as the detection of phishing web data, outside the framework of this research.

Table 41 Virus check status

Software name / service name	Number of detections	Virus detected
Online Link Checker / Doctor Web, Ltd.	1	Win32.HLLM.Graz
ClamAV	0	
Virus Buster / Trend Micro Incorporated		

Now, let's take a look at the downloaded data. We applied optional processing to the wget command in the download processing, but in time we came to suspect the effectiveness of this optional processing. According to the wget manual, the `-k` option "converts links in a document into those suitable for local browsing after the downloading is completed. This affects all areas linked to external contents, such as hyperlinks, embedded images, links to style sheets, and hyperlinks to content other than HTML." But when we examined the downloaded data, areas not converted into a relative path were found here and there even in the same site. These areas were infrequent overall, and eventually they were removed from those to be judged when the local data were assessed and judged.

### 3.4.2.4 Web interoperability discrepancy determination

#### (1) Outline of work

We can make web interoperability discrepancy determinations on downloaded web page data. We can execute discrepancy determinations by registering and setting two or more modules to determine discrepancy (hereinafter simply referred to as “Modules”) in a check tool prepared as an extended capability for Firefox. The URLs to be determined are extracted one by one from the list of URLs and processed. The results of the processing are output to the execution log file.

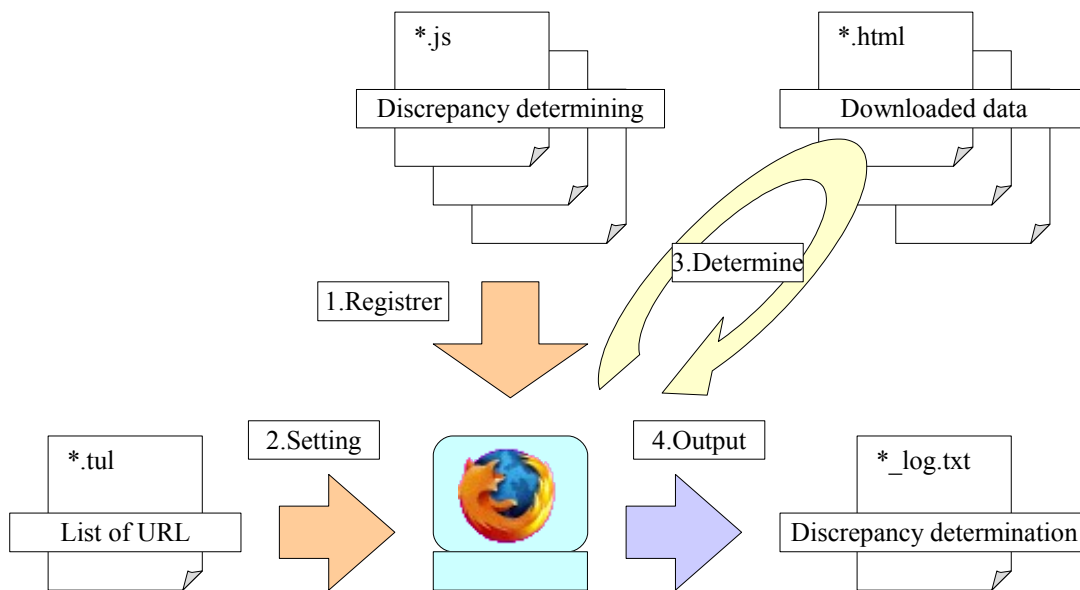


Fig. 10 Web interoperability discrepancy determination environment

For information, discrepancy determination processing was executed twice during the research period in line with the status of preparation of Modules.

In the first execution of discrepancy determination processing, the Modules were inadequate in certain respects and in some cases they were incapable of determining non-compatible properly. For these cases, proper determination results were obtained by correcting defective Modules and executing a second discrepancy determination using the downloaded data that should have been determined in the first determination. About 10,000 pieces of downloaded data were extracted at random in advance in the second execution of discrepancy determination processing to check the Modules. This enabled us to detect inadequacy in the Modules before all of the pieces of data were checked, thereby sparing us from the toil of repeating the same procedures.

A problem arising during the web interoperability discrepancy determination effectively halted the operation of the check tool and made it impossible to proceed with the determination. Apparently an HTML file of downloaded data contained a description of forced redirecting. Based on this description, Firefox attempted

to connect with non-existent local file system paths or URLs that couldn't be accessed. When the check tool stopped in these cases, we excluded the sites involved and proceeded with the work as before. As a result of this problem, there were 363 sites removed from those to be checked, leaving 192,002 sites that could ultimately be researched.

## (2) Work results

The final results of the counting work are shown in the "Results of crawling implementation."

### 3.4.2.5 RDB storage

#### (1) Outline of work

We analyzed an execution log output in the discrepancy determination processing and then compiled a database of our analytical results. This database was to be used to improve the efficiency of the count processing. MySQL4, an open source RDB, was used as the database management system. Ruby, an open source object-oriented script language, was used for preparation of a conversion script.

The conversion work was done twice. This was necessary, as the web interoperability discrepancy determination processing was divided and executed as two separate work operations (hence the conversion work was carried out each time the discrepancy determination processing was performed). In the second conversion work, we used a corrected version of the conversion script that had been used in the first conversion work.

An unexpected log output in the execution log of the discrepancy determination processing posed a problem. Apparently the discrepancy determining module had been inadequate. We addressed the problem during the second discrepancy determination processing by re-executing the processing using a corrected module and then updating the portions of the data that had encountered problems in the first discrepancy determination processing.

## (2) Work results

The following lists the number of records stored in each table.

Table 42 Number of records stored

Table name	Number of records
Site	200,000
Non-compatibility information	166
Corrected script	31
Site evaluation	192,002
Meta information	142,065
Site evaluation by rule	27,432,282

### 3.4.3 Results of implementation of crawling

#### 3.4.3.1 Number of sites to be subjected to the final check

After removing the 7,998 sites that had become invalid due to download conditions or troubles during the non-compatibility determination from the 200,000 sites to be researched, the remaining 192,002 sites were checked with the use of the non-compatibility-determining modules (the Modules).

#### 3.4.3.2 Modules for items to be checked

As many as 166 web interoperability discrepancies were extracted in this research. For some, however, difficulties in automatic detection, etc. prevented us from preparing Modules. There were 24 items removed at this stage. No Modules or checks were prepared or conducted for the remaining 142 items.

Table 43 Status of preparation of the Modules

Upper section: Non-compatibility No. Lower section: Item to be determined (○: Preparation of Module, ×: Excluded from the items to be checked at this stage)									
1	2	3	4	5	6	7	8	9	10
○	○	○	○	○	○	○	○	○	○
<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>	<b>17</b>	<b>18</b>	<b>19</b>	<b>20</b>
○	○	○	○	○	○	○	○	○	○
<b>21</b>	<b>22</b>	<b>23</b>	<b>24</b>	<b>25</b>	<b>26</b>	<b>27</b>	<b>28</b>	<b>29</b>	<b>30</b>
○	○	○	○	○	○	○	○	○	○
<b>31</b>	<b>32</b>	<b>33</b>	<b>34</b>	<b>35</b>	<b>36</b>	<b>37</b>	<b>38</b>	<b>39</b>	<b>40</b>
○	○	○	○	○	○	○	○	○	○
<b>41</b>	<b>42</b>	<b>43</b>	<b>44</b>	<b>45</b>	<b>46</b>	<b>47</b>	<b>48</b>	<b>49</b>	<b>50</b>
○	○	○	○	○	○	○	○	○	○
<b>51</b>	<b>52</b>	<b>53</b>	<b>54</b>	<b>55</b>	<b>56</b>	<b>57</b>	<b>58</b>	<b>59</b>	<b>60</b>
×	○	○	○	○	○	○	○	○	○
<b>61</b>	<b>62</b>	<b>63</b>	<b>64</b>	<b>65</b>	<b>66</b>	<b>67</b>	<b>68</b>	<b>69</b>	<b>70</b>
○	○	○	○	○	○	○	○	×	×
<b>71</b>	<b>72</b>	<b>73</b>	<b>74</b>	<b>75</b>	<b>76</b>	<b>77</b>	<b>78</b>	<b>79</b>	<b>80</b>
○	×	○	○	○	○	○	○	○	○
<b>81</b>	<b>82</b>	<b>83</b>	<b>84</b>	<b>85</b>	<b>86</b>	<b>87</b>	<b>88</b>	<b>89</b>	<b>90</b>
○	○	×	○	○	○	×	○	○	○
<b>91</b>	<b>92</b>	<b>93</b>	<b>94</b>	<b>95</b>	<b>96</b>	<b>97</b>	<b>98</b>	<b>99</b>	<b>100</b>
○	×	○	○	○	○	○	○	○	○
<b>101</b>	<b>102</b>	<b>103</b>	<b>104</b>	<b>105</b>	<b>106</b>	<b>107</b>	<b>108</b>	<b>109</b>	<b>110</b>
×	×	×	×	×	○	○	○	×	○
<b>111</b>	<b>112</b>	<b>113</b>	<b>114</b>	<b>115</b>	<b>116</b>	<b>117</b>	<b>118</b>	<b>119</b>	<b>120</b>
×	×	○	×	×	○	○	○	○	×
<b>121</b>	<b>122</b>	<b>123</b>	<b>124</b>	<b>125</b>	<b>126</b>	<b>127</b>	<b>128</b>	<b>129</b>	<b>130</b>
×	×	×	○	×	○	○	○	○	○
<b>131</b>	<b>132</b>	<b>133</b>	<b>134</b>	<b>135</b>	<b>136</b>	<b>137</b>	<b>138</b>	<b>139</b>	<b>140</b>
○	○	○	○	○	○	○	○	○	○
<b>141</b>	<b>142</b>	<b>143</b>	<b>144</b>	<b>145</b>	<b>146</b>	<b>147</b>	<b>148</b>	<b>149</b>	<b>150</b>
○	○	○	○	○	○	○	○	○	○
<b>151</b>	<b>152</b>	<b>153</b>	<b>154</b>	<b>155</b>	<b>156</b>	<b>157</b>	<b>158</b>	<b>159</b>	<b>160</b>
○	○	○	○	○	○	○	○	○	○
<b>161</b>	<b>162</b>	<b>163</b>	<b>164</b>	<b>165</b>	<b>166</b>				
○	×	×	○	○	○				

Checked: ○ / Excluded: ×

### 3.4.3.3 Rate of detection of web interoperability discrepancies

Among the 192,002 sites to be checked as a population parameter, some sort of web interoperability discrepancy was detected in 63,045 sites. This translated into a 32.8% rate of detection. We can see that nearly a third of the sites contained some sort of web interoperability discrepancy in their web pages.

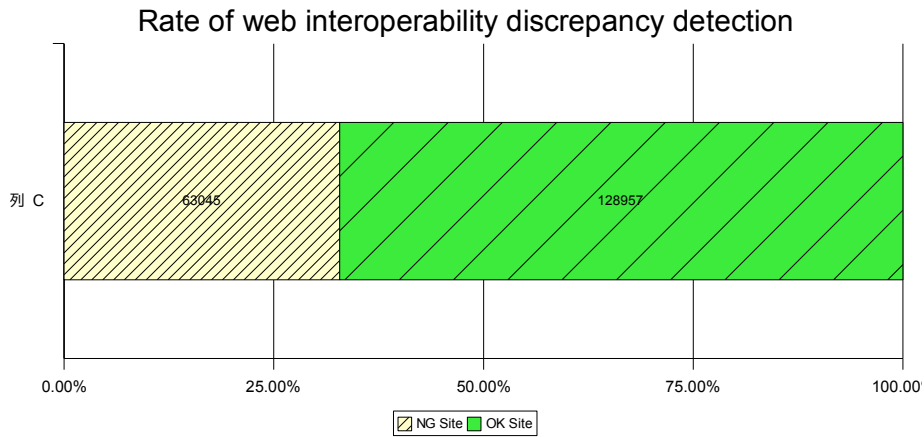


Fig. 11 Rate of non-compatibility detection

Table 44 State of web interoperability discrepancy detection

Number of effective sites	192,002
Number of sites with detected web interoperability discrepancies	63,045
Rate of non-compatibility detection	32.8%
Number of web interoperability discrepancies detected	108,025

### 3.4.3.4 Number of detections by discrepancy ID

There are considerable variations in the number of detected web interoperability discrepancies when the number of detections is examined by discrepancy ID. While there are web interoperability discrepancies in which a considerable number of web interoperability discrepancies were detected, there are also items in which none were detected whatsoever. This table shows only the number of detections. Details are described later based on individual discrepancy IDs. No web interoperability discrepancies were detected in 24 web interoperability discrepancies in total.

Table 45 Number of detections by discrepancy ID

Upper section: discrepancy ID									
Lower section: Number of detected web interoperability discrepancies									
Background (Gray, Not included in the determination; Light blue, The number of detections is 0 (zero); Red, The number of detections is 5,000 or more; White, Others)									
1	2	3	4	5	6	7	8	9	10
9	14	57	217	11	272	0	11	4	0
11	12	13	14	15	16	17	18	19	20
5	46	78	20	0	1	2,103	2,685	1,329	2,248
21	22	23	24	25	26	27	28	29	30
2,532	2,463	2,520	9	410	0	449	12	8	0
31	32	33	34	35	36	37	38	39	40
711	2,752	40	350	5,654	187	1,942	2,047	2	9
41	42	43	44	45	46	47	48	49	50
10	250	7	1,080	9,273	134	2,851	1,422	16	500
51	52	53	54	55	56	57	58	59	60
-	0	17	1	877	55	116	151	152	21
61	62	63	64	65	66	67	68	69	70
3,571	800	766	1	79	17	1,143	0	-	-
71	72	73	74	75	76	77	78	79	80
0	-	2,833	459	4,697	0	2,993	79	4	1,382
81	82	83	84	85	86	87	88	89	90
0	619	-	0	1,443	0	-	414	7,662	1,805
91	92	93	94	95	96	97	98	99	100
2	-	1,755	20	7	0	474	659	409	56
101	102	103	104	105	106	107	108	109	110
-	-	-	-	-	6	1,598	135	-	0
111	112	113	114	115	116	117	118	119	120
-	-	188	-	-	0	2	515	537	-
121	122	123	124	125	126	127	128	129	130
-	-	-	432	-	3,659	24	4,931	499	303
131	132	133	134	135	136	137	138	139	140
1	4	27	2	0	30	8,169	131	938	998
141	142	143	144	145	146	147	148	149	150
1	500	1	1,249	2	0	0	1	9	21
151	152	153	154	155	156	157	158	159	160
331	14	0	0	0	38	2	90	10	8
161	162	163	164	165	166				
0	-	-	1,424	12	1				

### 3.4.4 Analysis of crawling results

#### 3.4.4.1 Trends of web interoperability discrepancy check

The total number of web interoperability discrepancies detected in this web interoperability discrepancy check was tabulated by classification based on the rule set, and the trends by web interoperability discrepancy classification were analyzed. As a result of this analysis, we can see that the rate of detection of web interoperability discrepancies related to Internet Explorer (e.g., the proprietary properties and attributes of IE) is very high. If these web interoperability discrepancies are set aside, however, “web interoperability discrepancy arising from disregard of the standard” is the second most frequently detected type. Although web interoperability discrepancies related to IE account for half of the total overall, these findings provide irrefutable evidence that the factors behind the production of a web interoperability discrepancy description depend on authoring tools. Two findings attest to this: the frequent detection of web interoperability discrepancies arising from disregard of the standard, and the high rate of disregard of the standard by authoring tools (to be described later).

#### 3.4.4.2 Rate of inclusion of Generator descriptions in META information

In this research we extracted Generator descriptions from the META information by focusing our attention on whether specific authoring tools were involved in the mixing-in of web interoperability discrepancies. A Generator description is an important piece of information to identify prepared authoring tools. It became obvious, however, that some of the pages included more than one Generator description. The significance of this multiple description isn't clear, but the phenomenon explains why the total number of Generator descriptions exceeds the number of URLs with Generator descriptions. About one-third of the sites covered in this research included Generator descriptions. This gives a glimpse of how content creators use authoring tools when they create web pages.

Table 46 Rate of inclusion of Generator descriptions

URL with a Generator description	64,823
Total number of Generator descriptions	65,675
Percentage of Generator descriptions	33.8%

#### 3.4.4.3 Breakdown of Generator descriptions

There are a number of different Generator descriptions, including even some which seem to be handwritten. Generator descriptions are roughly classified into the categories shown below. IBM Homepage Builders account for the great majority, or 68% of the total. We can assume that the product itself has a commensurate share of the authoring tool market. We must consider this ratio here to grasp the characteristics of each authoring tool for web interoperability discrepancy factors.

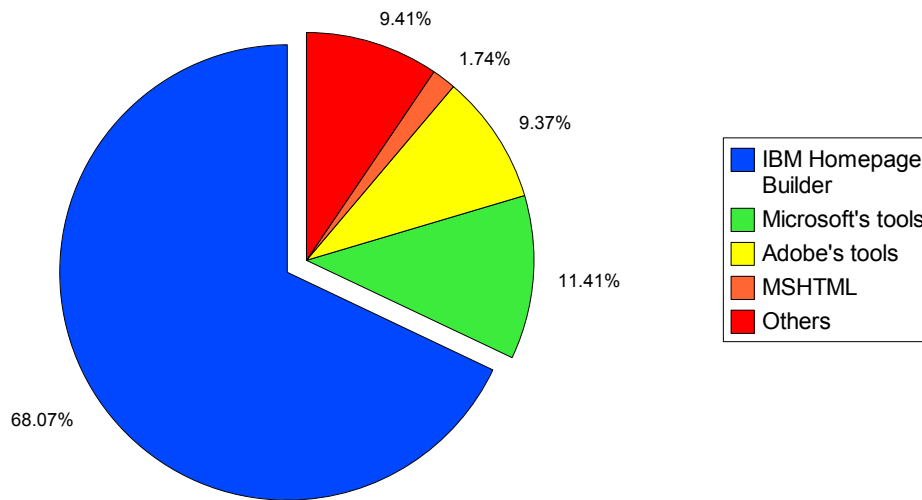


Fig. 12 Number of Generator descriptions

Table 47 Comparison of the number of Generator descriptions

Related to Adobe	6,151
Related to IBM Homepage Builder	44,707
Related to the various authoring tools of Microsoft	7,495
Related to MSHTML	1,143
Others	6,179
Total	65,675

#### 3.4.4.4 Number of detections by discrepancy ID in only sites with Generator descriptions

The number of detections by discrepancy ID is shown in Table 48, by limiting them to sites having Generator descriptions in the META information. When we make a comparison with Table 45, we can see that trends vary by discrepancy ID. Given that generator descriptions make up one-third of the total sites, this finding suggests that the authoring tools make it very easy (or difficult) to embed factors corresponding to discrepancy IDs in sites with ratios very different from the one-third.

To confirm this trend numerically, we determined a ratio to the total number of detections as a population parameter. To do so, however, we had to consider the potentially large influence of errors in such a small sample population. To hold down this influence, we examined a dissociation difference from 33.8%, the rate of inclusion of Generator descriptions, for discrepancy IDs with a population parameter of 1,000 or more. Discrepancy IDs with a dissociation difference of 20% or more are shown in graph form. There are eleven

such discrepancy IDs, and their appearance is thought likely to vary largely, depending on whether authoring tools are used.

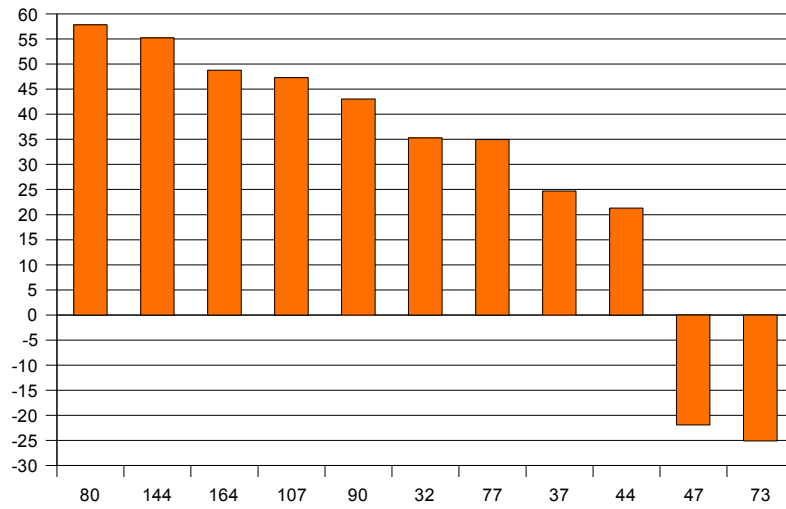


Fig. 13 Rate of dissociation from the rate of detection

Table 48 Number of detections by discrepancy ID in only sites with Generator descriptions

Upper section: discrepancy ID Lower section: Number of non-compatibility detections									
Background (Gray, Not included in the determination; Light blue, The ratio to the total is higher; Red, Higher for the presence of Generator description; White, Others)									
1	2	3	4	5	6	7	8	9	10
4	6	31	83	1	264	0	45	0	0
<b>11</b>	<b>12</b>	<b>13</b>	<b>14</b>	<b>15</b>	<b>16</b>	<b>17</b>	<b>18</b>	<b>19</b>	<b>20</b>
1	14	26	18	0	0	997	1,281	624	1,077
<b>21</b>	<b>22</b>	<b>23</b>	<b>24</b>	<b>25</b>	<b>26</b>	<b>27</b>	<b>28</b>	<b>29</b>	<b>30</b>
1,221	1,161	1,195	7	356	0	44	0	1	0
<b>31</b>	<b>32</b>	<b>33</b>	<b>34</b>	<b>35</b>	<b>36</b>	<b>37</b>	<b>38</b>	<b>39</b>	<b>40</b>
383	1901	11	91	3,006	61	1,136	1,079	1	7
<b>41</b>	<b>42</b>	<b>43</b>	<b>44</b>	<b>45</b>	<b>46</b>	<b>47</b>	<b>48</b>	<b>49</b>	<b>50</b>
4	179	1	595	2,195	29	340	454	3	412
<b>51</b>	<b>52</b>	<b>53</b>	<b>54</b>	<b>55</b>	<b>56</b>	<b>57</b>	<b>58</b>	<b>59</b>	<b>60</b>
-	0	7	0	142	33	14	6	6	2
<b>61</b>	<b>62</b>	<b>63</b>	<b>64</b>	<b>65</b>	<b>66</b>	<b>67</b>	<b>68</b>	<b>69</b>	<b>70</b>
890	116	361	0	15	5	442	0	-	-
<b>71</b>	<b>72</b>	<b>73</b>	<b>74</b>	<b>75</b>	<b>76</b>	<b>77</b>	<b>78</b>	<b>79</b>	<b>80</b>
0	-	247	66	1,510	0	2,058	41	2	1,266
<b>81</b>	<b>82</b>	<b>83</b>	<b>84</b>	<b>85</b>	<b>86</b>	<b>87</b>	<b>88</b>	<b>89</b>	<b>90</b>
0	197	-	0	449	0	-	198	2,338	1,387
<b>91</b>	<b>92</b>	<b>93</b>	<b>94</b>	<b>95</b>	<b>96</b>	<b>97</b>	<b>98</b>	<b>99</b>	<b>100</b>
0	-	699	8	3	0	231	39	267	18
<b>101</b>	<b>102</b>	<b>103</b>	<b>104</b>	<b>105</b>	<b>106</b>	<b>107</b>	<b>108</b>	<b>109</b>	<b>110</b>
-	-	-	-	-	1	1,296	34	-	0
<b>111</b>	<b>112</b>	<b>113</b>	<b>114</b>	<b>115</b>	<b>116</b>	<b>117</b>	<b>118</b>	<b>119</b>	<b>120</b>
-	-	12	-	-	0	0	133	131	-
<b>121</b>	<b>122</b>	<b>123</b>	<b>124</b>	<b>125</b>	<b>126</b>	<b>127</b>	<b>128</b>	<b>129</b>	<b>130</b>
-	-	-	92	-	923	6	1,447	222	17
<b>131</b>	<b>132</b>	<b>133</b>	<b>134</b>	<b>135</b>	<b>136</b>	<b>137</b>	<b>138</b>	<b>139</b>	<b>140</b>
0	0	4	0	0	8	2457	30	156	171
<b>141</b>	<b>142</b>	<b>143</b>	<b>144</b>	<b>145</b>	<b>146</b>	<b>147</b>	<b>148</b>	<b>149</b>	<b>150</b>
1	102	0	1,112	0	0	0	0	3	5
<b>151</b>	<b>152</b>	<b>153</b>	<b>154</b>	<b>155</b>	<b>156</b>	<b>157</b>	<b>158</b>	<b>159</b>	<b>160</b>
147	6	0	0	0	24	0	21	4	4
<b>161</b>	<b>162</b>	<b>163</b>	<b>164</b>	<b>165</b>	<b>166</b>				
0	-	-	1,179	2	0				

### 3.4.4.5 Trend of Generator in individual web interoperability discrepancies

In this research we compared the number of web interoperability discrepancy detections between a case with Generator descriptions and a case without them. As a result, we found that there were specific discrepancy IDs in which sites with Generator descriptions produced more detections than sites without them. We show the counted values for the most conspicuous discrepancy IDs below. Let's begin by looking at the graph shown below. From among the discrepancy IDs in which discrepancy detection was performed, we extracted those with a very frequent detection of sites with Generator descriptions.

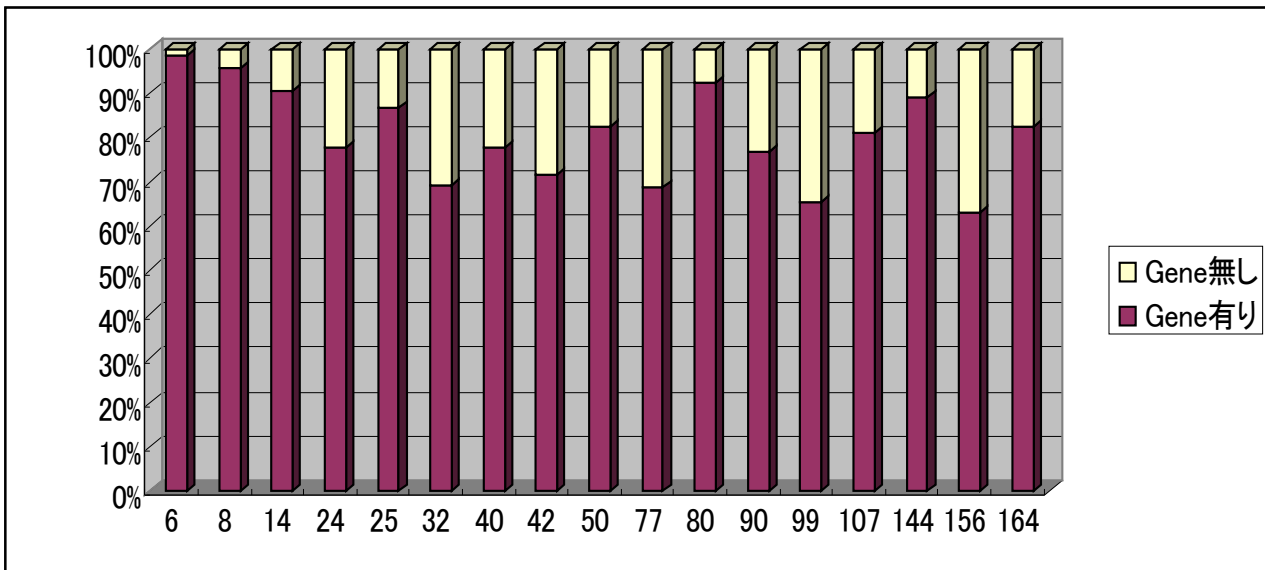


Fig. 14 Generator dependency

□ Upper: Without Generator description  
 □ Lower: With Generator description

Numbers in the axis of abscissas are discrepancy IDs. In these 17 kinds of ID checks out of 142, many non-compatibility detections are seen in Generator descriptions.

#### (1) Trend of discrepancy ID6 and ID8 (Browser and authoring tool dependence)

With regard to discrepancy ID6 and ID8, the total number of detections is small, but when details of Generator are examined we find that the rate of detection was concentrated in Microsoft Word. This seems to suggest that the HTML produced by Microsoft Word includes these web interoperability discrepancies. discrepancy ID6 and ID8 are proprietary properties of IE related to CSS. Each results in a garbled text display arising from a failure to specify grids. The problem cannot be corrected.

#### (2) Trend of discrepancy ID14 (Browser and authoring tool dependence)

Discrepancy ID14 is another example of a non-compatibility dependent on authoring tools. Overall, it can be detected only very rarely. Generators detected in connection with discrepancy ID14 are concentrated in Microsoft Excel, and none are detected in Microsoft Word. discrepancy ID14 has to do with ruby for letters

in CSS. It actually has no effect, so no corrections to the display are required. But given that discrepancy ID14 appears only in specific products, we suspect it to be somehow dependent on the authoring tools.

### (3) Trend of discrepancy ID32 (Browser dependence)

Discrepancy ID32 is related to a blink processing function (blinking on and off) specific to Netscape and is invalid in Internet Explorer. The total number of detections of discrepancy ID32 is fairly high, and most of the detections are concentrated in IBM Homepage Builder. While other Generators are also detected, among Microsoft-related products we find a fairly high rate of detection in the FrontPage application for homepage creation and a relatively low rate of detection in Word. It remains unclear, however, whether the non-compatibility descriptions that become invalid in Internet Explorer are produced in FrontPage, etc.

### (4) Trend of discrepancy ID77 (The standard isn't yet supported.)

Discrepancy ID77 is related to blink processing (blinking on and off). The web interoperability discrepancy is based on the web standard, but isn't yet supported by Internet Explorer. There are many detections for this web interoperability discrepancy, as well as many production-authoring tools involved. When we consider that it isn't yet supported by Internet Explorer, we deem it to be a major web interoperability discrepancy.

### (5) Trend of discrepancy ID80 (Disregard of the standard)

Discrepancy ID80 is a description in disregard of the web standard. When specifying a value (e.g., negative (-1)) for z-index in CSS, a reference value should be specified. In Internet Explorer, however, the page is rendered based on a proprietary interpretation even in the absence of the reference value. Here, we extracted web interoperability discrepancies from authoring tools that produced descriptions in disregard of the reference value. As for this web interoperability discrepancy description, we can see that IBM Homepage Builder has an overwhelmingly large number of detections. In the count of Generators, the rate of use of IBM Homepage Builder is the highest. When the breakdown of the rate of use for each version is reviewed, we detect average numbers of web interoperability discrepancies in Homepage Building Ver. 6 to Ver. 10. Ver. 11 has only been released for two months, hence there are still very few users and very few detections. It therefore remains unclear whether Ver. 11 will show the same trend as the earlier versions of this application.

Thus, when we compare the numbers of web interoperability discrepancies detected in sites with Generator descriptions, web interoperability discrepancies can be classified roughly into three types; "discrepancies specific to authoring tools," "discrepancies in disregard of the standard," and "discrepancies arising from a failure to support standards."

"Web interoperability discrepancies specific to authoring tools" can be seen frequently in Microsoft-related products. Many sites apparently use HTML prepared especially with Microsoft Word, particularly home pages, without adding modifications.

On a percentage basis, the “web interoperability discrepancies in disregard of the standard” are most common in IBM Homepage Builder, a very popular authoring tool. web interoperability discrepancies in disregard of the standard are detected, overall and on average, in Home Builder Ver. 3 through Ver. 11. Interestingly, however, there is no single version with a particularly conspicuous rate of detection.

As for “web interoperability discrepancies arising from a failure to support standards,” many authoring tools are detected as web interoperability discrepancies in items from standards not yet supported by Internet Explorer.

#### 3.4.4.6 Rate of non-compatibility detection by jp domain

When investigated by jp domain, the rate of non-compatibility detection was almost the same for almost all domains. Accordingly, we omit these results here. We note, however, that the rates for the sites of educational institutions (ac.jp), government-affiliated sites (go.jp), and local government-related sites (lg.jp) are lower than those for other domains.

Table 49 Rate of detection by JP domain

Type of domain	Rate of non-compatibility detection per site
ac.jp	39.7%
ad.jp	59.0%
co.jp	50.2%
ed.jp	46.0%
go.jp	30.4%
gr.jp	48.9%
lg.jp	34.0%
ne.jp	67.0%
or.jp	51.4%
Other jp domains	64.0%

#### 3.4.4.7 Rate of non-compatibility detection by provider

Sites to be researched this time include those which can be determined to belong to the same provider, judging from their URLs. Many of these sites are thought to be the home pages or blogs of individuals. These sites are likely to use templates, etc. provided by providers, hence we decided to count the web interoperability discrepancies by provider. By doing this, we can ascertain how many sites with high dependence each provider includes. First we counted the domains with 500 or more sites, then we researched the sites in each domain. The number of sites doesn't necessarily represent the share of the provider, but rather the rate included in the Yahoo! Japan directory. Each provider has a different number of sites. For clarification, we therefore present a graph comparing numbers of sites and numbers of detections to exhibit the simple trends in the rates of content web interoperability discrepancies. As the graph shows, the rate of content web interoperability discrepancies per site is 0.5% for Infoseek and 19.1% for Yahoo! Japan, and the sites belonging to these two providers generally include relatively few web interoperability

discrepancies. With regard to the other providers, the 50% rate of content web interoperability discrepancies suggests that steps are taken to take web compatibility into account using site-provided templates, etc.

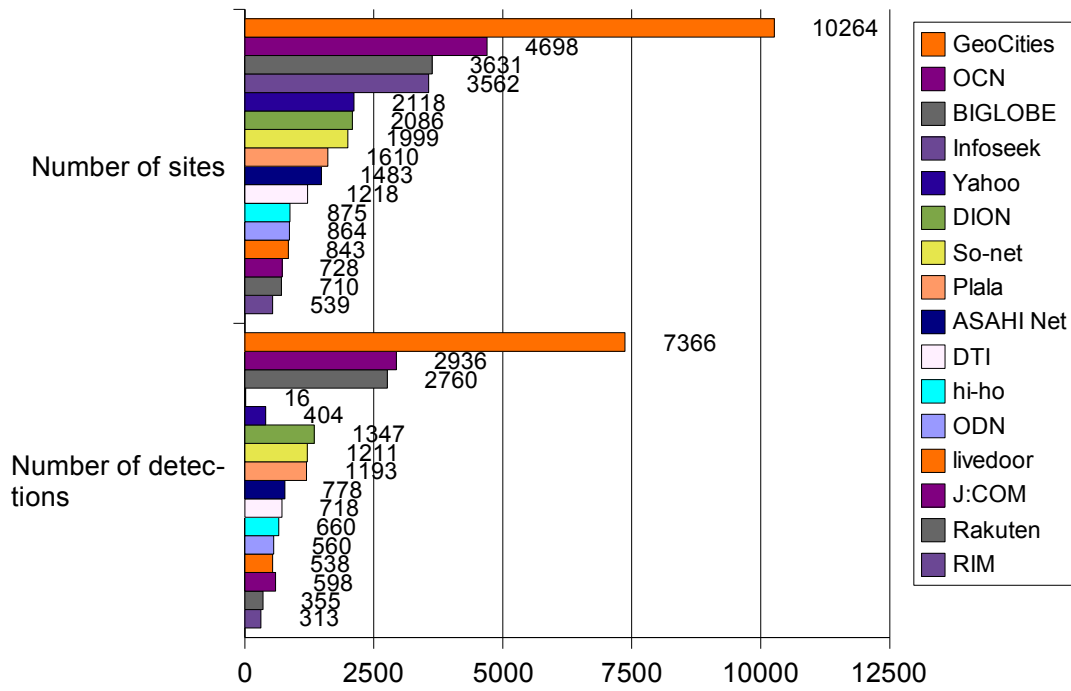


Fig. 15 Comparison of the number of sites and the number of detections

Table 50 Rate of detection by provider

Provider	Number of non-compatibility detections by site
GeoCites	71.8%
OCN	62.5%
BIGLOBE	76.0%
Infoseek	0.5%
Yahoo! Japan	19.1%
DION	64.6%
So-net	60.6%
Plala	74.1%
ASAHI Net	52.5%
DTI	59.0%
hi-ho	75.4%
ODN	64.8%
livedoor	63.8%
J:COM	82.1%
Rakuten	50.0%
RIM	58.1%

#### 3.4.4.8 Problems in web interoperability discrepancy analysis

Many Generator descriptions of META information are used to analyze the number of web interoperability discrepancy detections in this research. While the Generator descriptions account for the largest percentage in the META information, they only make up a little more than 33% of the URLs registered in Yahoo. In other words, we still cannot clearly determine how the contents were produced for the remaining 67%. There were also other pieces of META information of many other types, but descriptions for these pieces were discretionary and there were still too few for use in a meaningful analysis. Some pieces of META information also included information on CREATE or DATA-CREATE, that is, the date of creation and information on a date of renewal. If there are many descriptions of this type of information in a time series, it seems possible to analyze the rate of occurrence and rate of increase in the time series of web interoperability discrepancies. Actually, however, there were too few descriptions for use as quantitative data for the analysis.

### 3.5 Research on situations of intranets

With regard to research on intranets, we have checked many of the sites listed previously.

In the case of intranets, there is an expectation that more complicated problems will arise. Because intranets are not opened to the public, many problems are expected to remain obscure or to remain essentially closed information. For this reason, it is difficult in some respects to expect that automatic check tools such as TouchUpWeb will function well. In the case of research on intranets, therefore, we analyzed the situations through manual research and research by interviews with the information system department, a party

concerned.

### 3.5.1 Work procedure

The work procedure varied slightly from intranet to intranet. Basically, however, the work was carried out through the following procedure.

1. Interview on systematization policy
2. Selection of the system
3. Interview on the overall details of the system
4. Login ID selection and login
5. Selection of menu items
6. Access to each function
7. Dependency check

#### (1) Interview on systematization policy

First of all, we interviewed the persons in charge of systematization policies of the information systems departments of the organizations to be researched. The items covered in this interview research (shown below) were related mainly to the web system.

- Framework and policy for system development and management
- Selection criteria for web systems and client/server type systems
- Browser subject to operation checks (IE only, IE and Netscape Navigator, etc.)
- Awareness of and opinions on browsers with different operations and displays
- Status of IE7 and plans to respond to it
- Complaints from users who use browsers other than those to covered by the operation checks
- Methods for testing web systems
- Points to keep in mind during development when a system is manufactured in-house
- Points to keep in mind when a system is ordered

#### (2) Selection of system

Next, out of the web systems owned by organizations to be researched, we selected those to be covered by the intranet research. The selections were actually made by persons in charge in the information system department, in consideration of whether their basic systems or department management systems were to be researched, and so on.

#### (3) Interview on the overall details of the system

Before we actually researched each system, we interviewed a person in charge in the information system

department to grasp the overall details of the system to be researched and thereby smoothed out the research work itself. We tried to clarify the following in these interviews.

1. For what purpose is the system designed?
2. Who will be users of the system?
3. Number of users, frequency of use, and number of accesses.

#### (4) Login ID selection and login

Starting with this item, we researched the systems selected in (2) in good order.

First, we launched Firefox 1.5 and IE 6.0 on the research PCs and accesses the system's login screen. Next, we logged in either by using a test account (if one was available) or as a general user (if a test account wasn't available). If the login screen wasn't displayed or we failed to login for other reasons, we terminated research on the system concerned and investigated the cause if possible.

#### (5) Selection of menu items

Once we logged in we extracted items to be researched from the menu, paying attention to the following points.

- Information on the rating and salary of individuals and confidential information on the organization should be excluded.
- It would be redundant to research all items of the same kind, hence only one representative item should be selected and researched when there are two or more items of the same kind.

#### (6) Access to each function

We selected each menu item with both Firefox and IE to compare the resulting operations and displays between the two browsers. The comparison items are as follows. When we found differences between Firefox and IE, we recorded the differences as problems, obtained the approval of a person in charge from the information system department, and recorded a screen shot and the page source

- Inspection of the layout of the entire screen
  - Whether any of the images, tables, text, etc. are overlapping or omitted.
- Inspection of the operation of buttons
  - Whether a confirmation screen is provided for buttons, and whether the operation itself proceeds smoothly. We didn't press the send button if we were logged in as a general user and no confirmation screen appeared.

### 3.5.1.2 Method of analysis

In our analytical work we clarified the causes of the problems that arose and examined the degree of impact from problems, studies solutions (including the consideration of TCO), and assessed whether check tools can be applied. To gather information, we conducted research on intranets by visiting other companies and organizations with system administrators, etc. in attendance. In some cases, therefore, we were unable to fully investigate the causes of discrepancy arising from limitations of time and security. In some cases we managed to obtain the source of a page but failed to reproduce the page in the local environment.

The following shows the three levels of impact defined.

- Large: The use of the relevant function is conspicuously hindered.
- Medium: A part of the relevant function is unusable.
- Small: The relevant function is unhindered.

### 3.5.2 Organizations on which research was to be performed

The intranet research was to be performed on the five organizations.

During the development of the intranet systems, some clients chose to remain fixed to non-OSS browsers in order to avoid paying extra development costs. We also clarified the sources of these problems through interviews with information system departments and web creators, and we used this information to work out recommendations for improvement.

We interviewed three web creators, in addition to the five organizations which could offer the information of their intranets.

### 3.5.3 Research results (Example in a company A)

The intranet system research was to be performed on 25 systems that could be accessed from the portal site of a company A. Two systems dealing with management information were excluded. These systems to be researched are shown in Table 51. The system names are not shown as the actual designations, we list them as generic descriptions.

Table 51 Systems investigated by the intranet system research

ID	System name	Remarks
1	Portal system	
2	Business management system	Research was inapplicable.
3	ISO information site	
4	Management conference report site	
5	Management conference system	Research was inapplicable.
6	Dispatch request system	
7	Staff site	
8	Order receipt performance search system	
9	Estimation and order placement system	
10	Late-evening and holiday applications	
11	IT equipment management system	
12	Quality control system	
13	Project management system	
14	Work flow system	
15	Transportation expense adjustment system	
16	Personnel and salary system	
17	Subcontract law management system	
18	Conference room reservation system	
19	Business card preparation system	
20	e-learning system	
21	Management system for the state of entering / leaving the room	
22	Extension / e-mail search system	
23	Post-office box system	
24	Information materials center site	
25	Book stock DB search system	
26	Report search system	

#### 3.5.3.1 Research results

Table 52 shows the results of the intranet of company A. Ten problems concerning seven systems were found. Out of these problems, three were caused by HTML output by MS Office.

Table 52 Outline of the results of the company A's intranet research

ID	System name	Function / menu	Problem classification	Cause	Degree
----	-------------	-----------------	------------------------	-------	--------

		name			of impact
1	Portal system	Content display screen	Garbled display.	HTML/CSS	Small
2	ISO information site	Collection of know-how	Garbled display.	HTML/CSS	Medium
3	Quality control system	Manual	Some imaged don't display.	HTML	Large
4	Quality control system	Manual	Cannot be displayed.	Others	Large
5	Project management system	Login	Operation fails.	Others	Large
6	Work flow system	Login	Operation differs.	Javascript	Large
7	Transportation expense adjustment system	Login	Operation fails.	VBScript	Large
8	e-learning system	Main screen	Garbled display.	Character code	Large
9	e-learning system	Answer screen	Operation fails.	Javascript	Large
10	e-learning system	Learning screen	Garbled display.	HTML/CSS	Medium

(1) Portal system “Content display screen”

Table 53 Portal system “Content display screen”

System name	Portal system
Function / menu name	Content display screen
Problem classification	Garbled display
Cause	HTML
Degree of impact	Small
Is it possible to apply the check tools?	Possible

In some cases, an object with a line feed at the right end of a cell when viewed in IE will change into an object with a line feed occurring at the beginning of the next line when viewed in Firefox. This anomaly results from differences between IE and Firefox in the methods to handle margins and padding. Firefox handles margins and padding in accordance with the standard.

To correct the garbled display, the margins and/or padding should be adjusted to equalize the cell width in IE and Firefox. Many websites have similar problems that can be easily corrected using the extended capability of TouchUpWeb.

(2) ISO information site “Collection of know-how”

Table 54 ISO information site “Collection of know-how”

System name	ISO information site
Function / menu name	Collection of know-how
Problem classification	Garbled display
Cause	HTML/CSS
Degree of impact	Medium
Can check tools be applied?	Highly possible

When a page showing a collection of know-how concerning ISO is accessed, Firefox displays a message informing users that displays other than those in IE cannot be guaranteed, as in Fig. 16. If we continue to browse the ensuing pages, the browser displays the content and text, but not the images in the presentations (Fig. 17). The page display sizes are also different from those in IE (Fig. 18).

This problem is thought to arise from HTML or CSS. But because this page was output by MS PowerPoint, the complicated structure prevented us from identifying the source of the problem. The problem is likely to be easily correctable once the cause is known, if cause is rooted in HTML or CSS as we suspect.

このプレゼンテーションの内容は、お使いのブラウザで正しく表示されない可能性があります。このプレゼンテーションは、より新しいバージョンの Microsoft Internet Explorer 用に最適化されています。  
 続行する場合は、次をクリックしてください：[ここ](#)。

Fig. 16 Display in Firefox (1)

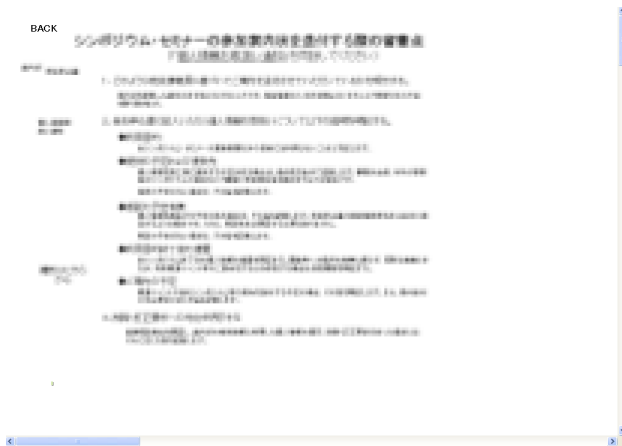


Fig. 17 Display in Firefox (2)

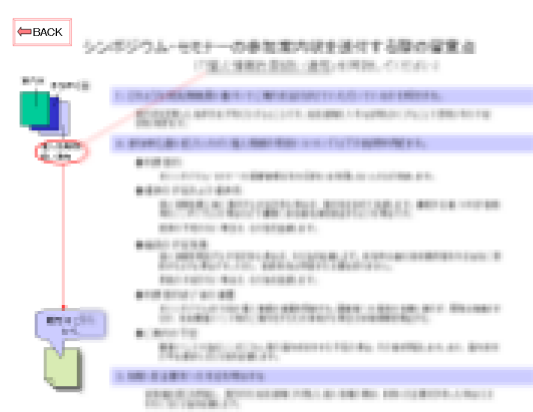


Fig. 18 Display in IE

※ Screenshots are blurred to conceal information not open to the public.

(3) Quality control system “Manual”

Table 55 Quality control system “Manual”

System name	Quality control system
Function / menu name	Manual
Problem classification	No images displayed
Cause	HTML
Degree of impact	Large
Can check tools be applied?	Highly possible

IE displays a page showing the manual using images (Fig. 19). Firefox doesn't display any of the images (Fig. 20). No image file exists in the URL designated as an img tag, hence the operation of Firefox is correct. If we consider a phenomenon in which no image file can be found, we can identify HTML as the source of the problem. If the cause can be identified, the problem is therefore very likely to be correctable using check tools. For information, this page was output by MS Word.



Fig. 19 Display in IE



Fig. 20 Display in Firefox

※ Screenshots are blurred because they contain information not open to the public.

(4) Quality control system “Manual”

Table 56 Quality control system “Manual”

System name	Quality control system
Function / menu name	Manual
Problem classification	Operation fails
Cause	Others
Degree of impact	Large
Can check tools be applied?	Not possible

Some of the pages linked from the manual in the quality control system can be displayed in IE, but not in Firefox (a download dialog is displayed). This is because the pages linked from the manual are formatted in an IE-specific mht format.

To enable the display of these pages in Firefox, the files linked from the manual should be prepared in the HTML format rather than the mht format. This cannot be accomplished with check tools, as it involves work on the server side.

(5) Project management system “Login”

Table 57 Project management system “Login”

System name	Project management system
Function / menu name	Login
Problem classification	Operation fails
Cause	Others
Degree of impact	Large
Can check tools be applied?	Not possible

The project management system is activated by clicking the link in the portal system. In Firefox, however, the project management system won't activate even when the link is clicked. This is because this system uses a plug-in which is not provided for Firefox.

Given that no plug-in is available, it is impossible to activate this system from Firefox. The problem is therefore unsolvable even if check tools, etc. are used.

(6) Work flow system “Login”

Table 58 Work flow system “Login”

System name	Work flow system
Function / menu name	Login
Problem classification	Operation differs
Cause	Javascript
Degree of impact	Large
Can check tools be applied?	Possible

When this system is accessed in Firefox, the browser displays the dialog shown in Fig. 21 and login fails. This occurs because Javascript checks the browser and displays an error dialog if it encounters any browser other than IE.

There are two ways to avoid this dialog. The first is to use the extended capability of applications such as TouchUpWeb to rewrite Javascript with instructions to prevent its display. The second method is to use the extended capability for changing the User Agent in order to give the browser an appearance identical to that of IE.



Fig. 21 Error dialog displayed in Firefox

※ Screenshots are blurred to conceal information not open to the public.

You are now using a browser other than Internet Explorer (IE). A problem may arise in system operation. Log on again using IE. OK
--

## (7) Transportation expense adjustment system “Login”

Table 59 Transportation expense adjustment system “Login”

System name	Transportation expense adjustment system
Function / menu name	Login
Problem classification	Operation fails
Cause	VBScript
Degree of impact	Large
Can check tools be applied?	Not possible

When this system is accessed in Firefox, the browser displays the error message shown in Fig. 22 and no login screen appears. This is because VBScript is used on the page concerned.

To enable the use of this system in Firefox, the VBScript must be rewritten into Javascript. This is unrealistic even from the viewpoint of cost. Therefore, the problem cannot be corrected with check tools, etc.

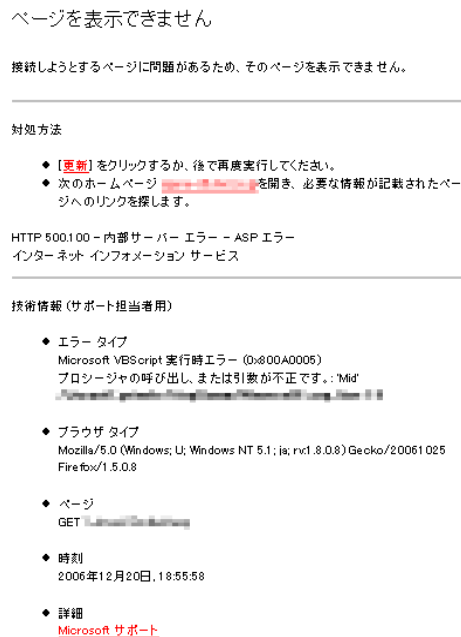


Fig. 22 Error display in Firefox

※ Screenshots are blurred to conceal information not open to the public.

(8) e-learning system “Main screen”

Table 60 e-learning system “Main screen”

System name	e-learning system
Function / menu name	Main screen
Problem classification	Garbled display
Cause	Character code
Degree of impact	Large
Can check tools be applied?	Unlikely

The main e-learning screen is garbled in both IE (Fig. 23) and Firefox (Fig. 24). The problem probably stems from the character code setting. If the setting on the web server side needs to be changed, the problem cannot be corrected with check tools.



Fig. 23 Display in IE



Fig. 24 Display in Firefox

※ Screenshots are blurred to conceal information not open to the public.

(9) e-learning system “Answer screen”

Table 61 e-learning system “Answer screen”

System name	e-learning system
Function / menu name	Answer screen
Problem classification	Operation fails
Cause	Javascript
Degree of impact	Large
Can check tools be applied?	Highly possible

Testees taking the e-learning test must press the “send” button to send answers. In the case of IE, this button is disabled (un-clickable) unless all of the questions are answered. Once all the questions are answered, the button is enabled in IE but remains disabled in Firefox.

The cause lies in Javascript, which checks whether all questions are answered. To determine radio buttons in the answer form, a form name should be designated after a document object, as in “document.form1.radiobuttons...”. In this script, however, “document” is omitted and “form1.radiobuttons” is applied instead. We encounter this type of problem because IE adds “document” automatically, whereas Firefox does not.

This problem can be corrected by adding “document” before a form name. Thus, the problem can be corrected with check tools.

(10) e-learning system “Learning screen”

Table 62 e-learning system “Learning screen”

System name	e-learning system
Function / menu name	Learning screen
Problem classification	Garbled display
Cause	HTML/CSS
Degree of impact	Medium
Whether it’s possible to apply the check tools	Highly possible

When a page showing educational materials in e-learning is accessed, Firefox displays a message informing users that displays other than those in IE are not guaranteed (see Fig. 16). If the user continues to browse the next pages, the content is displayed but the presentations are garbled (Fig. 26). In addition, the page display sizes are different from those in IE (Fig. 25).

This problem is thought stem from HTML or CSS. But because this page was output by MS PowerPoint, the complicated structure prevented us from identifying the source of the problem. The problem is likely to be easily correctable once the cause is known, if the cause is rooted in HTML or CSS as we suspect.



Fig. 25 Display in IE



Fig. 26 Display in Firefox

※ Screenshots are blurred to conceal information not open to the public.

### 3.5.3.2 Interview with the information planning group

To research the opinions and awareness of user companies with regard to the compatibility of web browsers, we held interviews with the information planning group of the corporate planning department, the department in charge of intranet system management in the company A.

An outline of the interview follows.

(1) Framework and policy for system development and management

There are several company-wide systems and departmental systems. The company-wide systems are under the control of each section in the clerical work department which has responsibility of the work using such systems. A staff in such a section determines the operation specifications and requirement definitions. The information planning group is responsible for transferring the operation specifications and requirement definitions into the system adopted. There is no internal manufacture system, and all work operations are subcontracted.

(2) Selection criteria for web system and client/server type system

The web system selection criteria are unclear. As for C/S, maintenance is challenging and requires extensive time and work. Up until a short time ago, we have therefore been tacitly using web systems. Recently, however, problems with the usability and speed of these web systems have prompted us to start using .NET. We think that the number of rich clients will increase in the future.

Rich clients provide options, such as Flash and AJAX. We have judged, however, that Flash would be unsuitable for business-oriented systems. We decided to start using .NET because Microsoft has begun to gear up for business system platforms and good ones have started to appear on the market. We were also encouraged by the low introduction cost. A low introduction cost means that the levels of entry are low. Also, there are many engineers.

(3) Browser examined by the operation check (IE only, IE and Netscape Navigator, etc.)

The operation check covers IE6 only. The support of two or more browsers puts a strain on resources. We have too few personnel and would prefer not to incur the extra cost.

(4) Awareness and opinions as to each browser having different operations and displays

We are not happy to see browsers with different looks and operations. If possible, we would like a uniform appearance and operations in all browsers. It would cause us trouble, however, to sacrifice compatibility as we strive to realize that. Considerable cost and time will be required for correction. In some cases it would be more convenient to use IE-specific functions. Some users think that our own functions should be used. In light of these conditions, we do not make systems which are completely in conformity with the standard. Most users consider the system to be adequate as long as it works properly in their environment (IE).

(5) Status of IE7 and plans to respond to it.

Our coverage of IE7 has been limited to a quick check. In the web system, we ingeniously prevent users from pressing the “return” button, etc. by positioning the menu bar in the hidden mode, or by some other means. In IE7, however, the menu bar sometimes fails to disappear. We encountered many problems in the beta version, but none were major.

(6) Complaints from users who use browsers other than those covered by the operation check

We have also received complaints and requests from users who use Firefox or Lunascape. Lunascape is compatible with IE, but in rare cases it behaves differently.

(7) Method for testing web systems

Web systems are tested on the user side based on the test specifications prepared by the developers. Web systems are considered adequate if they operate in IE. We do not think that the internal codes are checked. When major modifications are made, they are also checked by the information planning group or section taking control. Web systems are considered adequate if there are no problems with use.

If there are unmistakable bugs in the code, we make it a rule to fix them even though the web system is operating in IE. If the framework needs to be modified, however, the modification work can be very difficult.

(8) Points to keep in mind when a system is ordered

The information planning group serves as a bridge between sections taking control and subcontractors. The group has a difficult time ensuring that SE reliably understands the business design. The group rarely checks the method for implementation, however, as a common platform is used.



## Chapter 4 Summary

In this research we carried out cyclopedic examinations using the TouchUpWeb system and check tools based on the TouchUp script controlled by the TouchUpWeb system. The research focused on a few more than 200,000 Japanese websites registered in Yahoo! Japan. We identified the sites to be web crawled based on the directory tree API provided by the Yahoo! Developer Network and then examined the sites with the check tools. The sites researched were limited to Japanese pages only.

As a result of checks by crawling, we found web interoperability discrepancies in as many as 32.8% of the sites. It was quite surprising to discover some sort of discrepancy in close to one-third of all of the web pages checked. However, note that there remains a possibility of over-estimation due to limitations and defects of determining modules.

Except discrepancies from the limitation of the determining modules, the web interoperability discrepancies found in this research frequently were the discrepancies concerning colors and spaces. Therefore such discrepancies are considered negligible in retrieving information of the page. However, we could count 0.017 discrepancies, which have fatal effect on the page rendering and/or interacting with the page, per one site. Since we are repeating click-and-visit various web pages in a daily life, it is a severe condition if one or two sites of the 100 sites have such interoperability discrepancies.

We also revealed that some authoring tools produce the pages which contain web interoperability discrepancies, by calculating cross tables with Generator information and non-compatible items. 33.8% of the sites uses a authoring tool and 33.4% of such sites have some interoperability discrepancies. In this case, determining rate per one site is twice as much as the average of all sites. This result makes us conclude that authoring tools have a risk to produce non-compatible pages.

The research on the actual conditions of websites in intranets focusing on the intranets of five companies was also conducted. In reporting research results on products that handle sensitive information in these intranets, note that we have either increased the complexity or omitted specific system configurations, screen examples, and so forth. The research on the intranets included analysis of the actual conditions through manual research and interviews with information system departments (a party concerned).

The actual situation on development of intranet systems and awareness of cost were discussed in the interview. Comments from the respondent of the interviews made it clear that managers of information system departments have to be conservative for preparation of the migration from Internet Explorer 6.0 to Internet Explorer 7.0. In addition, they told us that the web contents dependencies came from not only the defects of authoring tools but also packages for building the intranet systems. SI vendors also have the responsibility for this problem because they provide no support other than the specified browser as the intranet system developed by them.

These results have been compiled separately as a set of written recommendations to promote the creation of web content that will not depend on specific platforms, and to show guidelines for web content creators and content authoring tool vendors. We hope that this report and the achievements of this research will be used effectively and will help to popularize the use of OSS desktops.

**Research on the improvement of Web contents compatibility conducive  
to widespread use of OSS desktops  
Research report**

February 2007

Mitsubishi Research Institute, Inc.  
Research Center for Information Technology  
Tel: +81-3-3277-0750  
Jun IIO and Hiyoruki SHIMIZU

Copyright (C) 2007 Information-technology Promotion Agency, Japan, All rights reserved