Complexity Metrics for Cascading Style Sheets

Adewole Adewumi[1], Sanjay Misra[2], and Nicholas Ikhu-Omoregbe[1]

[1] Department of Computer and Information Sciences, Covenant University, Nigeria
[2] Department of Computer Engineering, Atilim Univeristy, Ankara, Turkey
{wole.adewumi,nomoregbe}@covenantuniversity.edu.ng
smisra@futminna.edu.ng

Abstract. Web applications are becoming important for small and large companies since they are integrated with their business strategies. Cascading Style Sheets (CSS) however are an integral part of contemporary Web applications that are perceived as complex by users and this result in hampering its widespread adoption. The factors responsible for CSS complexity include size, variety in its rule block structures, rule block reuse, cohesion and attribute definition in rule blocks. In this paper, we have proposed relevant metric for each of the complexity factors. The proposed metrics are validated through a practical framework. The outcome shows that the proposed metrics satisfy most of the parameters required by the practical framework hence establishing them as well structured.

Keywords: Cascading Style Sheets, Complexity Metrics, Software Metrics, Validation Criteria.

1 Introduction

Web applications are becoming important for small and large companies since they are integrated with their business strategies [10]. In this point of view, it is necessary that the applications should be reliable, usable and adaptable. However, achieving this goal is not an easy task. Web applications for large scale are always being complex and therefore the maintainability of such types of system is high. There exist several quality attributes: maintainability, usability, efficiency, functionality, reliability, portability and reusability. Maintainability is one of the most important quality attribute, which must be taken care of in the development process of the web applications otherwise maintaining quality of the web application for large systems will become a challenge. One of the ways to maintain the quality is by reducing the complexity of the applications.

The complexity of software is measured in order to be able to predict the maintainability and reliability of such software. Several complexity metrics have been proposed as at today to measure typical software programs. These measures are often based on cognitive informatics [12], [17], [18], [19], [20] a way of measuring software complexity based on cognitive weights [13]. In recent times, complexity metrics have also been proposed for the web domain particularly XML schema documents

B. Murgante et al. (Eds.): ICCSA 2012, Part IV, LNCS 7336, pp. 248–257, 2012.

[3], Web Services [2][16] and DTDs [1]. These are an integral part of contemporary web applications. Another integral part of web applications is Cascading Style Sheets (CSS). This is a style sheet language used to format the presentation of web pages written in HTML and XHTML. In addition it can also be applied to any kind of XML document bringing about aesthetically pleasing and user-friendly interfaces. The core advantage that CSS offers is separation of content from presentation. Despite this advantage, CSS is perceived as complex by users and this result in hampering its wide-spread adoption. Though it is perceived as complex, no metric has been proposed to measure its complexity as the field of style sheets is under-researched [7]. To solve this problem in this present paper, we start in section 2 by identifying the factors that bring about complexity in a CSS document and propose relevant metrics that can be used to measure each attribute. In section 3 we demonstrate each metric using suitable example(s) and in section 4 we validate the proposed metrics through a framework. Section 5 concludes the paper.

2 Proposed Metrics

The complexity of CSS refers to how easy it is to understand and maintain. All factors that make CSS difficult to understand or maintain are responsible for its complexity. Factors that are responsible for the complexity of CSS include: size, variety in its rule block structures, rule block reuse, cohesion and attribute definition in rule blocks.

The greater the size of a CSS the more complex the CSS will be. Since size is an important measure we are proposing rule length metric which is similar to lines of code in procedural programming and number of rule block metric which is similar to the number of modules in structured programming.

Also, the more dissimilar the rule blocks in a CSS are to one another, the more complex it will be to understand. Since variety in rule block structure is an important measure, we are proposing entropy metric.

The less number of modules that are reused in CSS increases the complexity of the CSS. Since reuse is an important measure, we are proposing number of extended rule blocks metric.

Furthermore, cohesion plays a vital role in the complexity of CSS as the lower the level of cohesion among rule blocks, the more complex the CSS. Since cohesion is an important measure, we are proposing number of cohesive rule blocks as metric.

In addition, the more the attributes defined for a rule block the more complex it will be. Since attribute definition in rule blocks is an important measure, we are proposing number of attributes defined per rule block as metric. In the paragraphs that follow, we describe the proposed metrics in detail.

2.1 Rule Length (RL)

A style sheet consists of a list of rules. Rule Length (RL) metric measures the number of lines of rules (code) in a CSS. This metric does not take into account white spaces

or comment lines in the CSS. This is essentially because white spaces and comments are not executed in CSS. RL is calculated using the following formula:

$$RL = \sum \text{rule statements in a CSS file}$$

A rule statement is any of the following:

- Selector(s) + opening brace of a rule block ({) for example, **body {**
- the attribute(s) of a selector ending with a semicolon (;) for example **color: #FFFFFF;**
- Closing brace of a rule block depicted as (**}**)

We now apply the metric to an example given in CSS code listing 1(Figure 1).

```
/* CSS Code Listing 1 */
body {
  margin: 0;
  padding: 0;
  background: #1B120B;
  font-family: Arial, Helvetica, sans-serif;
  font-size: 14px;
  color: #402C16;
}
h1, h2, h3 {
  margin: 0;
  padding: 0;
  font-weight: normal;
  color: #FFFFFF;
}
```

Fig. 1. CSS Code Listing 1

From CSS code listing 1, we can count 14 rule statements. Therefore,

$$RL = 14 \text{ lines}$$

2.2 Number of Rule Blocks (NORB)

A rule block refers to a selector and its attributes (properties) depicted by the syntax shown as follows.

```
/* Syntax of a rule block */
selector [, selector2, ...] [:pseudo-RLass] {
  property: value;
  [property2: value2;
...]
}
```

A typical CSS file will contain at least one rule block.

2.3 Entropy Metric (E)

The word 'entropy' was adapted from information theory [5] and is defined as a measure of uncertainty or variety. The entropy concept has been applied for the assessment of the rule complexity of procedural software [4] [6] [8] [11]. In recent times, [1] [3] have applied the concept to assess the structural complexity of XML schema documents written in W3C Document Type Definition (DTD) language and to measure the complexity of the schema documents written in W3C XML Schema Language [21] respectively. This was done by closely following the approach used by [4]. In this paper, we tow the same path to compute the entropy value of CSS documents.

According to [1] the definition is given by E = <S, F, P>, where E is an experiment with S as the set of elementary events, F is a Borel field [9] over S, and P is a probabilistic function assessing real values to events in F, then for a finite number of events C1, C2, ..., Cn the entropy of the given experiment E is

$$H = -\sum P(Ct)\log 2P(Ct) \text{ where } t = 1 \ldots n$$

Based on this definition the entropy of a given CSS document having n distinct class of elements can be calculated using the relative frequencies as unbiased estimates of their probabilities P(Ci), i=1, 2, ..., n. The distinct class of elements means that elements having the same structural complexities are grouped in the same class called equivalence class (C). This concept is demonstrated in section 3 using the CSS rule in CSS code listing 2 in Figure 2 in Appendix.

2.4 Number of Extended Rule Blocks (NERB)

This metric counts the number of rule blocks that are extended in a CSS file. It is calculated as follows:

$$NERB = \sum \text{extended rule block(i)} \text{ where } i = 1 \ldots n$$

2.5 Number of Attributes Defined per Rule Block (NADRB)

This metric determines the average number of attributes defined in the rule blocks of a CSS file. It can be calculated as follows:

NADRB = (Total no. of attributes in all rule blocks / Total no. of rule blocks)

2.6 Number of Cohesive Rule Blocks (NCRB)

Cohesion can be described as the "single-mindedness" of a component [10]. In the case of CSS, this can refer to rule blocks possessing a single attribute. NCRB metric counts all rule blocks that possess only one attribute. It can be calculated as follows:

$$NCRB = \sum \text{rule block (i) possessing only one attribute} \text{ where } i = 1 \ldots n$$

3 Demonstration of the Proposed Metrics

For illustration, we apply all the proposed metrics from section 2 to the example given in CSS code listing 2 (Figure 2 in Appendix).

RL metric: The value of the rule length metric is 40 lines.

$$RL = 40 \text{ lines}$$

NORB metric: The value of the NORB metric is 9.

$$NORB = 9$$

Entropy metric: The entropy value of the CSS rule in CSS code listing 2(Figure 2 in Appendix) is calculated by first determining the equivalence classes – this means grouping similar rule blocks. This is given as follows:

C1 = {body} = 1 element
C2 = {{h1, h2, h3}} = 1 element
C3 = {h1, h2, h3} = 3 elements
C4 = {a} = 1 element
C5 = {a:hover} = 1 element
C6 = {#page, #content} = 2 elements

The relative frequency of occurrence of the equivalence classes of the CSS document is the number of elements (i.e. attributes inside the equivalence class), divided by the total number of rule blocks in the CSS document. There are nine (9) rule blocks in the CSS document shown in CSS code listing 2 and so the relative frequency of occurrence of the equivalence class C5 = {#page, #content}, is P(C6) = 2/9. When all elements fall into only one equivalence class, then the minimum entropy value is determined. In that case, P (C1) = 9/9 = 1, this would then imply that entropy value is:

$$H = -\sum P(C_t)\log_2 P(C_t)$$

$$= P(C_1)\log_2 P(C_1)$$

$$= 0$$

On the other hand, the possible maximum entropy occurs when each rule block in the CSS rule is distinct. In such a case, the number of equivalence classes equal to the number of rule blocks, i.e. P(Ci) = 1/n, i = 1, 2, ..., n and n is the number of rule blocks or equivalence classes in the CSS. The entropy value of the CSS rule, in this case is:

$$H = -\sum P(C_t)\log_2 P(C_t)$$

$$= -\sum (1/n) \log_2 (1/n)$$

The entropy value for the CSS document shown in CSS code listing 2 is therefore calculated as:

$$E(CSS) = H$$

$$= -\sum P(Ct)\log 2P(Ct) \text{ where } t = 1..\ n$$

$$= (1/9)*\log_2 (1/9) + (1/9)*\log_2 (1/9) + (3/9)*\log_2 (3/9) + (1/9)*\log_2 (1/9) + (1/9)*\log_2 (1/9) + (2/9)*\log_2(2/9)$$

$$= 0.2441 + 0.2441 + 0.3662 + 0.2441 + 0.2441 + 0.3342$$

$$= 1.6768$$

NERB: There are 2 extended rule blocks in CSS code listing 2 namely: h1, h2, h3 {...} and a {...}.
h1, h2, h3 {...} is extended to give h1{...}, h2{...} and h3{...} while
a{...} is extended to give a:hover{...}. Therefore,

$$NERB = 2$$

NADRB: There are 22 attributes in all inside CSS code listing 2. There are also 9 rule blocks in all. Applying the formula defined in section 2 we have:

$$NADRB = 22/9 = 2.44$$

In essence, the result shows that on the average two attributes are defined per rule block. The higher this value is the more complex will be the CSS.

NCRB: The total number of rule blocks that possess one attribute in CSS code listing 2 is 4 namely: h1{...}, h2{...}, h3{...}, and a:hover{...}
Hence,

$$NCRB = 4$$

4 Practical Validation of the Proposed Metrics

To validate the six complexity metrics proposed we use the framework given by Kaner [14]. It is one of several validation criteria. The framework is more practical than the formal approach. It is based on answering the following points:

Purpose of the measures
The purpose of the measures is to evaluate the complexity of cascading style sheets.

Scope of usage of the measure: The proposed RL, NORB, entropy, NERB, NADRB and NCRB metrics are good predictors of understandability of CSS. They are

therefore a valuable contribution for maintainability of CSS. The scope of use is by web development teams that work on styling web interfaces.

Identified attribute to measure
The identified attributes to measure from our suite of metrics are understandability, reliability and maintainability. All these attributes are directly related to the quality of CSS.

Natural scale of the attribute
The natural scales of the attributes cannot be defined, since they are subjective and require the development of a common view about them.

Natural variability of the attribute
Natural variability of the attributes can also not be defined because of their subjective nature. It is possible that one can develop a sound approach to handle such attribute, but it may not be complete because other factors also exist that can affect the attribute's variability. In this respect, it is difficult to attain knowledge about variability of the attribute.

Definition of metric
The metrics have been formally defined in Section 2.

Measuring instrument to perform the measurement: We have counted all the parameters of the metrics manually and computed the proposed metrics. Further, we aim at developing a tool/software for measuring the proposed suite of metrics.

Natural scale for the metrics
For the natural scale of the proposed metrics, we have to go through measurement theory. When we analyze our metrics according to Briand and Morasca [15] we find that, they are in the ratio scale.

The natural variability of readings from the instrument
Since the reading from our counting instrument is not subjective and does not require any interpretation, we can say that no variability (i.e. measurement error) on readings from the instrument can be expected. Note that, in case of automated counting, we assume that there is no bug in the devised algorithm.

Relationship between the attribute to the metric value
There is a direct relation between the complexity of CSS and our proposed metrics. In other words all the proposed metrics are predictors of complexity in CSS.

Natural and foreseeable side effects of using the instrument
Once we automate the complexity calculation, it will not require considerable additional workload of manpower of the company. The only cost will be the automation.

Table 1. Results of applying proposed metrics to CSS code listing 2

Metrics	Code Listing 2
RL	40
NORB	9
Entropy	1.6768
NERB	2
NADRB	2.44
NCRB	4

5 Concluding Remark and Further Work

In this paper, we identified factors that bring about complexity in CSS and also proposed complexity metrics based on each of these factors for analyzing the complexity of CSS documents. With these proposed metrics, Web developers and designers can measure the complexity of CSS documents in terms of size, variety in rule block structure, rule block reuse, cohesion and the average number of attributes defined per rule block. The proposed metrics were validated practically through a framework to prove their usefulness and practical applicability. It was found that the proposed metric satisfies most of the parameters required by the practical evaluation framework.

As future work, we intend to validate each metric through Weyuker's properties. Rigorous empirical validation will also be done. In addition, the development of an automated tool for computing the metrics is also a task of future work.

References

1. Basci, D., Misra, S.: Entropy Metric for XML DTD Documents. ACM SIGSOFT Software Engineering Notes 33(4) (2008)
2. Basci, D., Misra, S.: Data Complexity Metrics for XML Web Services. Advances in Electrical and Computer Engineering 9(2) (2009)
3. Basci, D., Misra, S.: Entropy as a Measure of Quality of XML Schema Document. The International Arab Journal of Information Technology 8(1), 16–24 (2011)
4. Davis, J., LeBlanc, R.: A study of the applicability of complexity measures. IEEE Transaction on Software Engineering 14, 366–372 (1988)
5. Hamming, R.: Coding and information theory. Prentice Hall, Englewood Cliffs (1980)
6. Harrison, W.: An entropy-based measure of software complexity. IEEE Transactions on Software Engineering 18, 1025–1029 (1992)
7. Marden, P.M., Munson, E.V.: Today's Style Sheet Standards: The Great Vision Blinded. Computer (1999)
8. Mohanty, S.N.: Entropy metrics for software design evaluation. The Journal of Systems and Software 2, 39–46 (1981)
9. Papoulis, A.: Probability, random variables and stochastic processes. McGraw-Hill, New York (1965)
10. Pressman, R.S.: Software Engineering: A Practitioner's Approach. McGraw-Hill, New York (2005)

11. Torres, W., Samadzadeh, M.H.: Software reuse and information theory based metrics. IEEE Transactions on Software Engineering, 437–446 (1990)
12. Wang, Y.: On Cognitive Informatics. In: Second IEEE International Conference on Cognitive Informatics (ICCI 2002), pp. 34–42 (2002)
13. Wang, Y, Shao, J.: A New Measure of Software Complexity based on Cognitive Weights. Can. J. Electrical and Computer Engineering, 69–74 (2003)
14. Kaner, C.: Software Engineering Metrics: what do they measure and how do we know? In: Proc. Tenth Int. Software Metrics Symp., Metrics, pp. 1–10 (2004)
15. Briand, L.C., Morasca, S., Basily, V.R.: Property based software engineering measurement. IEEE Transactions on Software Engineering 22, 68–86 (1996)
16. Basci, D., Misra, S.: Metrics Suite for Maintainability of XML Web-Services. IET Software 5(3), 320–341 (2011)
17. Misra, S., Cafer, F.: Estimating Complexity Of Programs In Python Language. Technical Gazette 18(1), 23–32 (2011)
18. Misra, S., Akman, I., Koyuncu, M.: An Inheritance Complexity Metric for Object Oriented Code: A Cognitive Approach. SADHANA 36(3), 317–338 (2011)
19. Misra, S., Akman, I.: Unified Complexity Measure: a measure of Complexity. The Proc. National Academy of Sciences India (Sect. A) 80(2), 167–176 (2010)
20. Misra, S., Akman, I.: Weighted Class Complexity: A Measure of Complexity for Object Oriented Systems. Journal of Information Science and Engineering 24, 1689–1708 (2008)
21. Basci, D., Misra, S.: Measuring and Evaluating a Design Complexity Metric for XML Schema Documents. Journal of Information Science and Engineering 25(5), 1405–1425 (2009)

Appendix: 1

```
/* CSS Code Listing 2 */
body {
  margin: 0;
  padding: 0;
  background: #FFFFFF;
  font-family: Arial, Helvetica, sans-serif;
  font-size: 14px;
  color: #402C16;
}
h1, h2, h3 {
  margin: 0;
  padding: 0;
  font-weight: normal;
  color: #2E9F13;
}
h1 {
  font-size: 2em;
}
h2 {
```

```
  font-size: 2.4em;
}
h3 {
  font-size: 1.6em;
}
a {
  text-decoration: none;
  color: #2E9F13;
}
a:hover {
  text-decoration: underline;
}
/* Page */
#page {
  width: 960px;
  padding: 0;
  border-top: 1px solid #D0D0D0;
}
/* Content */
#content {
  float: right;
  width: 600px;
  padding: 0px 0px 0px 0px;
}
```

Fig. 2. CSS Code Listing 2