Requirements Statements Content Goodness and Complexity Measurement

CHAO Y. DIN
NuWave Solutions, LLC, Virginia, USA.
and
DAVID C. RINE,
George Mason University, Virginia, USA.

In software development projects where written requirements are produced, a requirements document summarizes the results of the requirements analysis and becomes the basis for subsequent software development. In many cases, the quality of the requirements documents dictates the success of the software development. The need for determining the quality of requirements documents is particularly acute when the target applications are large, complicated, and mission critical. The goal of this research is to develop quality indicators to indicate the quality of requirements statements in a requirements document. To achieve the goal, the goodness properties of the requirements statements are adopted to represent the quality of requirements statements. A suite of complexity metrics of requirements statements is proposed as the quality indicators and is developed based upon research of noun phrase (NP) chunks. A two phased empirical case study is performed to evaluate the usage of the proposed metrics. By focusing upon the complexity metrics based on NP chunks, the research aided in development of complexity indicators of low quality requirements statements of requirements documents.

Keywords: Complexity Measurement, Complexity Metrics, Cohesion, Coupling, NP Chunk, Requirements, Software Quality, Information Retrieval, Natural Language Processing

1. INTRODUCTION

This paper asserts that a requirements document, or formally a software requirements specification (SRS), is the single artifact produced through the requirements engineering process. A possible organization for an SRS includes, functional requirements, non-functional requirements, system requirements, user requirements, , etc (Sommerville, 2006). The quality of the SRS document inevitably becomes the main focus of requirements management. And the SRS, which is comprised of requirements, or requirements statements, is a basis for developing or building the rest of the software, including verification and validation phases. Despite abundant suggestions and guidelines on how to write high quality requirements statements, high quality SRS's are difficult to find.

The goal of this research is derived from that of prior research (Din, 2007, 2008; Din and Rine, 2008, 2012) and is to develop a set of metrics as quality indicators of requirements statements in an SRS. Many research studies on software quality (Cant et. al. 1995; DeMarco, 1982; Evangelist 1983; Garson, 2006; Kemerer, 1998; Lee, 2003), and various quality factors have been proposed to represent the quality of software. The quality factors adopted in this research are developed by Schneider (2002, 2000a, 2000b) and are named as goodness properties. The proposed metrics in this research are designed to work as quality indicators of requirements statements and are able to identify requirements statements with low goodness property values.

This research is derived from that of prior research (Din, 2007, 2008; Din and Rine, 2008, 2012) and uses statistical and partial parsing approaches to obtain a subset of noun phrases, named Noun Phrase (NP) chunks. Abney indicated that chunks are the basic language parsing unit and they correspond to "the basic concepts" for human brains to comprehend a text document Abney and Abney (1991). NP chunks are hence adopted as the basic processing units in this research.

This research is derived from that of prior research (Din, 2007, 2008; Din and Rine, 2008, 2012) and has developed three complexity metrics: count of NP chunks (NPC-Count), cohesion of requirements sections (NPC-Cohesion), and coupling of requirements sections (NPC-Coupling).

A two phased empirical case study was performed (Din, 2007, 2008; Din and Rine, 2008, 2012) to evaluate the proposed complexity metrics. Phase I of the case study compared the NPC-Cohesion and NPC-Coupling metrics with the cohesion and coupling metrics proposed by Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011). Ricker's research asserted the correlation between the complexity metrics and understandability, or comprehension, of the requirements statements. By confirming the consistency between Ricker's metrics and the proposed metrics, this research using (Din, 2007, 2008; Din and Rine, 2008, 2012) supports the assertion that the proposed metrics are correlated to understandability, one of the goodness properties, of the requirements statements. Furthermore, the case study reveals that the NP chunk based complexity metrics possess the following two additional capabilities: (1) they differentiate nouns from other syntactic categories (or word classes) - an important capability to differentiate object methods and properties from object classes, and (2) they adopt the spatial distance of NP chunks as the measuring units - an important capability in a cognition complexity model (Cant et. al. 1995).

The phase II case study then explains how the three proposed metrics can be used to identify low quality requirements statements in various scenarios.

Based upon the evidence provided by the two phased case study, it is concluded that the proposed complexity metrics indicate the content goodness properties of requirements statements. The contribution of the research from (Din, 2007, 2008; Din and Rine, 2008, 2012) is the construction of a suite of NP chunk based complexity metrics and the evaluation of the proposed suite of metrics.

The article is organized as follows. Section 2 presents the research problem statement and the importance of the research problem. The background of the research is summarized in Section 3. Section 4 illustrates the detailed process of obtaining the elements of measurement, Noun Phrase (NP) chunks, and then presents the proposed suite of metrics. Section 5 depicts the case study that is performed to evaluate the proposed metrics. Section 6 summarizes the contributions of the research.

2. RESEARCH PROBLEM AND ITS IMPORTANCE

2.1 Research Problem

The research was designed to answer the following question: How can low quality requirements statements be identified in an SRS? Although the research focuses on SRS's, the conclusions of the research can be applied to other requirements documents such as system requirements documents.

This research is derived from prior research (Din, 2007, 2008; Din and Rine, 2008, 2012) and answers the research question in a constrained environment where the current best practices of (1) identifying requirements and (2) identifying requirements defects are adopted. The constraints below are assumed to be best practices or are included in best practices.

(1) A systematic requirements method such as Viewpoint Oriented Requirements Definition (VORD) has been followed to produce the SRS (Sommerwille, 2006).
(2) Requirements are written in the IEEE standard requirements document format (IEEE/ANSI 830-1993).
(3) The SRS is grammatically correct and spelling errors have been checked.
(4) Traditional requirements guidelines to avoid ambiguous terms (large, many, user friendliness) and weak phrases (as applicable, as required, as a minimum) have been followed.
(5) A domain thesaurus and/or company term definitions have been supplied.
(6) A requirements inspection (Fagan, 1986) method has been adopted to eliminate requirements defects.

The above constraints identify the current best practices of (1) identifying requirements and (2) identifying requirements defects. However, certain requirements defects are hard to identify

and hence using these current best practices may not lead to effective outcomes. In particular, although constraint 6 indicates the requirements inspection (Fagan, 1986) can be used to identify requirements defects, requirements inspections can in the present practice be effective only if the sections of an SRS is limited to 8-15 pages so that a requirements quality inspector can perform an inspection of a given section within two hours time frame (Lee, 2003; Sommerville, 2006).

Another difficulty in identifying requirements defects is due to the spatial complexity of distant requirements statements, which refers to related requirements that are scattered far apart in a large SRS. Defects such as inconsistent or missing requirements statements can easily be missed due to the spatial distance. The current requirements inspection practice does not consider these kinds of requirements defects.

This research will expand the current practices in identifying low quality requirements, and hence requirements defects. In particular, this research supports identifying requirements defects due to the spatial complexity of distant requirements statements. This research will hence be supplement to the current practices. However, this research will not and has never intended to replace the current practices in identifying requirements defects.

2.2 The Importance of the Research

The proposed suite of complexity metrics is supported by a software tool researched and developed as part of this research to identify high complexity and hence low quality requirements. Once low quality requirements are identified, analysis of the low quality requirements can be conducted so that they can be grouped into categories of potential risks. Appropriate management actions, such as adding resources, simplifying the low quality requirements, or even abandoning the low quality requirements, can then be considered. (Wilson et. al. 1996) Identified several categories of system risks due to low quality requirements.

Low quality requirements are not only the source of software product risks but also the source of software development resource risks, which includes cost overrun and schedule delay. Using the proposed suite of complexity metrics as quality indicators, an impact assessment and threats classification of the identified low quality requirements can be performed. Such an early warning is vital to rescue a possibly failing project.

The development of high quality software products depends on management's awareness of such low quality requirements, their ability to expediently assess the impacts of those low quality requirements, and the capability to develop a plan to rectify the problem. The proposed suite of complexity metrics expedites the process of identifying low quality requirements statements. The subsequent risk analysis of those requirements can be performed earlier. The rectification plan can hence be developed and carried out in a timely manner. This process, from quickly identifying low quality requirements to developing and carrying out the corresponding rectification plan, provides the foundation for the development of high quality software.

Requirements inspection (Fagan, 1986) is designed to identify requirements defects, which is also the goal of the proposed complexity metrics. However, the requirements inspection process usually requires significant time and effort. Furthermore, defects due to related requirements that span more than 8 to 15 pages apart cannot be identified by current requirement inspection (Lee, 2003).

On the other hand, the proposed set of complexity metrics can be implemented using software tools and hence can quickly identify potential requirements defects. More importantly by using a software tool such as the one developed for this research, using the proposed set of complexity metrics supports solving the problem of identifying defects due to scattered but related requirements in large requirements documents.

3. BACKGROUND

3.1 Quality and Content Goodness Properties

Schneider proposed eleven goodness properties as a better coverage of quality factors (Purao and Vaishnavi 2003): Understandable, Unambiguous, Organized, Testable, Correct, Traceable, Com-

plete, Consistent, Design independence, Feasible, and Relative necessity. Representing quality with a set of properties that are each relatively easier to measure is an important step towards measuring quality.

It must be emphasized that this research does not attempt to assess all such properties. In this research and referencing (Din, 2007, 2008; Din and Rine, 2008, 2012), the main concerns which are important are the four goodness properties: Understandable, Unambiguous, Organized, and Testable, because they are relevant to the focus of our current research. The rationale for just using these four is that these four are related to the quality of requirements documents which are to be formally inspected. These four goodness properties realted to inspections quality are named as Content Goodness Properties and are the only goodness properties on which the remainder of the research will focus.

3.2 Complexity, Complexity Metrics and Measurement

Complexity is a major software characteristic that controls or influences quality. It has been widely accepted as an indirect indicator of quality (Fenton and Neil, 2000, Henderson-Sellers, 1996; Klemola, 2000; Levitin, 1986) and hence the content goodness properties.

(Purao and Vaishnavi, 2003) Provided a survey of complexity metrics and identified five out of 375 metrics that are related to requirements. Unfortunately, none of those five metrics are used to measure the natural language descriptions of the requirements. Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011), not listed on the survey, developed a set of requirements metrics: cohesion, context, and coupling. One of the contributions Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011) made is the demonstration of a positive correlation between cohesion, context, and coupling metrics and understandability of requirements statements. In (Pleeger, 2003), the context metric is assessed by the relationships between the sentences of a section and their section title. This research from (Din, 2007, 2008; Din and Rine, 2008, 2012) does not consider the context metric.

3.3 Readability Index

When measuring the quality of documents written in natural languages, the readability indexes or metrics must be considered. In general the written communication skills are measured in terms of readability and hence the use of readability indexes. Readability is a measure of the ease that a piece of writing can be read. Readability indexes are designed to access the suitability of a piece of writing for readers at particular grade levels or ages.

Factors considered in the readability indexes are number of words, number of syllables in words, number of words in sentences, , etc. Scores of the readability indexes are compared with scales based on judged linguistic difficulty or reading grade level.

One of the arguments against readability indexes is that difficult text often contains difficult words because it discusses abstract ideas while easy text uses common words because it discusses concrete experiences. In other words, difficult words are necessary to introduce new concepts and ideas, especially in education and research.

Another argument against readability indexes is that Readability formulas are based on the simple surface characteristics of the text. Measuring text elements that are primarily based on surface characteristics does not adequately capture comprehension (McNamara et. al. 1996).

A recent research project named Coh-Metrix has developed new readability indexes that are based on cohesion relations, interaction between a reader's skill level, world knowledge, and language and discourse characteristics. The Coh-Metrix project uses lexicons, part-of-speech classifiers, syntactic parsers, templates, corpora, latent semantic analysis, and other components that are widely used in computational linguistics (Graesser et. al. 2004).

Unfortunately, readability indexes, including Coh-Metrix, are not comparable with this research (Din, 2007, 2008; Din and Rine, 2008, 2012) for the following reasons:

(1) The readability metrics are designed for the whole documents, instead of sections of documents.

(2) The readability scores are not reliable indicators when the document under evaluation has less than 200 words (McNamara, 2001). However, many of the requirements statements have less than 50 words.

(3) Although Coh-Metrix attempts to measure the cohesion of texts, the definition of cohesion used by Coh-Metrix is different from the definition of cohesion used in Computer Science, and there are no coupling metrics in Coh-Metrix.

(4) Coh-Metrix does not have a single metric to represent the size, cohesion, or coupling complexity. Coh-Metrix includes more than 50 metrics to measure very specific aspects of texts. No composite metric that combines those specific aspects of a document has been proposed.

(5) Coh-Metrix attempts to measure the cohesion of texts. Future work of Coh-Metrix may address comprehension, or understandability. However, Coh-Metrix will never address the issue of testability and many other goodness properties.

4. NP CHUNK BASED COMPLEXITY METRICS

A major issue of a measurement program is "what to measure," and it is one of the most critical issues to the measurement research.

Because humans tend to read and speak texts one chunk at a time, Abney proposed using what is called chunks as the basic language parsing unit. There are several categories of chunks similar to the traditional categories of phrases. For example, there are Noun Phrase (NP) chunks, Verb Phrase (VP) chunks, Prepositional Phrase (PP) chunks, , etc (Abney and Abney, 1991). This research (Din, 2008) focuses on NP chunks and ignores other types of chunks.

4.1 Three Core Metrics

It has been recognized that it is not likely that a single metric can capture software complexity (Fenton and Neil, 1999; Kitchenham et. al. 1995). To deal with the inherent difficulty in software complexity, a myriad of indirect metrics of software complexity have been proposed (Purao and Vaishnavi, 2003). On the other hand, it is believed that a small subset of existing metrics can enable parsimonious evaluation, prediction and control of software complexity (Kemerer, 1998). This research introduces three types of complexity metrics, NPC-Count, NPC-Cohesion, and NPC-Coupling, for measuring the complexity of requirements statements in a requirements document.

Size counts are the oldest method of measuring complexity. For software design and coding, the most popular size count is Line of Code (LOC). The wide acceptance of LOC as a complexity metric is due to its simplicity, ease of application, inertia of tradition, absence of alternative size metrics, and its intuitive appeal (Dunsmore, 1984; Lee, 2003). In addition, multiple empirical studies indicate that LOC is better or at least as good as any other metric (Basili, 1980; Evangelist, 1983; Weyuker, 1988). All these evidence and findings indicate that counting should be one of the core software metrics. Zuse (Zuse, 1998) also identified simple counts in his measurement theory as one of the metrics that possesses all the desired properties of an ideal metric. Based upon the above reasons, two distinct metrics (NPC-Sentence and NPC-Req) are developed to count the NP chunks of a text, and these two metrics are collectively named as NPC-Count.

Many of the published metrics, either for procedural languages or for object-oriented languages, include some variation of the cohesion and coupling metrics (Briand et. al. 1998; Briand et. al., 1999). Furthermore, cohesion and coupling metrics are ubiquitous across a wide variety of measurement situations, including 4GLs, software design, coding and rework. Darcy and Kemerer believe that cohesion and coupling are effective metrics and they can represent the essential complexity measures for the general software design tasks (Darcy and Glass, 1990). Hence, NPC-Cohesion and NPC-Coupling are chosen to represent the complexity of requirements. To

assist the identification of low quality requirements, a composite metric (NPC-Composite) that combine cohesion and coupling measures is also proposed and studied in the research.

4.2 Requirements Documents Used

Two requirements documents are used in this research:

(1) A public domain requirements document for Federal Aviation Agency (FAA).
The requirements document is available in Ricker's dissertation Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011).

(2) Versions of the Interactive Matching and Geocoding System II (IMAGS II) requirements documents for U. S. Bureau of Census.
The IMAGS II, or IMAGS, project has gone through several iterations of requirements analysis. Four versions of the requirements documents are available for the research.

4.3 Sentence/Requirements Statement Level Complexity

The sentence level complexity metric, or NPC-Sentence, can be calculated as follows. For each NP chunk, the occurrence count in a sentence is divided by the total occurrence counts in all sentences. Then all the frequency distributions of the NP chunks in the sentence are added together to form the final complexity value. The calculation can be expressed as follows.

$$NPC - Sentence(sentence_j) = \sum_{1 \leqslant i \leqslant N} \frac{Entry(i,j)}{\sum_{1 \leqslant j \leqslant C} Entry(i,j)}$$

where $Entry(i,j)$ is the number of occurrence of **NP** chunk NP_i in $sentence_j$, $1 \leqslant i \leqslant N, 1 \leqslant j \leqslant C, N$ is the total number of **NP** chunks, and C is the total number of sentences. Intuitively, NPC-Sentence is a metric that measures the normalized count of **NP** chunks in a sentence of a document.

The requirements statement level complexity metric, or $NPC - Req(req_j)$, is the aggregation of $NPC - Sentence$ of the component sentences and can be expressed as follows.

$$NPC - Req(req_j) = \sum_{sentence_i \in req_j} NPC - Sentence(sentence_i)$$

where $1 \leqslant i \leqslant L_j, 1 \leqslant j \leqslant M, L_j$ is the total number of sentences of requirement j, and M is the total number of requirements.

Example - From partial parsing to sentence level complexity The following three requirements (four sentences) are extracted from the IMAGS Version 4 requirements document. The requirements in the IMAGS requirements document are marked with a label (e.g., "IM2-WKASSIGN-4") and a short description (e.g., "Assign Users to WAAs"). Note that WAA stands for Work Assignment Area and is a collection of counties. WAA is defined to facilitate the assignment of workload to individual users.

IM2-WKASSIGN-4: Assign Users to WAAs IMAGS II shall track and maintain users' assignment to WAAs. A user can be assigned to one or more WAAs, and a WAA can have more than one user assigned.

IM2-WKASSIGN-7: Assign Incoming Addresses on WAAs IMAGS II shall assign incoming addresses to users based upon their WAAs.

IM2-WKASSIGN-8: Assign Multiple WAAs to Multiple Users IMAGS II shall provide a way to assign a list of WAAs to multiple users at once.

The results of the chunk parsing are as follows.

$(S:$
$(0:<imags/NN><ii/NN>)$
$<shall/MD>$
$<track/VB>$
$<and/CC>$
$<maintain/VB>$
$(1:<users/NNS><'/POS><assignment/NN>)$
$<to/TO>$
$<waas/VB>$
$<./.>$
)

$(S:$
$(7:<imags/NN><ii/NN>)$
$<shall/MD>$
$<assign/VB>$
$<incoming/VBG>$
$(8:<addresses/NNS>)$
$<to/TO>$
$(9:<users/NNS>)$
$<based/VBN>$
$<upon/IN>$
$(10:<their/PRP\$><waas/NNS>)$
$<./.>$
)

$(S:$
$(2:<a/DT><user/NN>)$
$<can/MD>$
$<be/VB>$
$<assigned/VBN>$
$<to/TO>$
$(3:<one/CD>)$
$<or/CC>$
$(4:<more/JJR><waas/NNS>)$
$<,/,>$
$<and/CC>$
$(5:<a/DT><waa/NN>)$
$<can/MD>$
$<have/VB>$
$<more/JJR>$
$<than/IN>$
$(6:<one/CD><user/NN>)$
$<assigned/VBN>$
$<./.>$
)

$(S:$
$(7:<imags/NN><ii/NN>)$
$<shall/MD>$
$<assign/VB>$
$<incoming/VBG>$
$(8:<addresses/NNS>)$
$<to/TO>$
$(9:<users/NNS>)$
$<based/VBN>$
$<upon/IN>$
$(10:<their/PRP\$><waas/NNS>)$
$<./.>$
)

$(S:$
$(11:<imags/NN><ii/NN>)$
$<shall/MD>$
$<provide/VB>$
$(<12:<a/DT><way/NN>)$
$<to/TO>$
$<assign/VB>$
$(<13:<a/DT><list/NN>)$
$<of/IN>$
$(<14:<waas/NNS>)$
$<to/TO>$
$(<15:<multiple/NN><users/NNS>)$
$<at/IN>$
$<once/RB>$
$<./.>$
)

The chunk parsing results in 16 NP chunks (see Table 1), where sentence is abbreviated as "sent." and requirement is abbreviated as "req.". Note that the word "WAAs" in the first sentence is tagged as "VB", a verb. This is an error due to the nature of statistical NLP process, which considers words after "to" as verbs. The stop list indicated in the table is used to filter **NP** chunks that have little meanings.

The chunk parsing results in $16NP$ chunks (see Table 1), where sentence is abbreviated as "sent." and requirement is abbreviated as "req.". Note that the word "WAAs" in the first sentence is tagged as "VB", a verb. This is an error due to the nature of statistical NLP process, which considers words after "to" as verbs. The stop list indicated in the table is used to filter NP chunks that have little meanings.

By applying the stemming and text normalization techniques, the result is depicted in Table 2

For the sake of example, it is assumed that the four sentences constitute the complete requirements document. The resulting NPC-Sentence and NPC-Req are shown in Table 3.

Stop NP chunks: (a user), (one), (users), (a way), (a list)		req. 1		req. 2	req. 3
		sent. 1	sent. 2	sent. 3	sent. 4
<imags/NN> <ii/NN>	0,7,11	1	0	1	1
<users/NNS> <'/POS> <assignment/NN>	1	1	0	0	0
<more/JJR> <waas/NNS>	4	0	1	0	0
<a/DT> <waa/NN>	5	0	1	0	0
<one/CD> <user/NN>	6	0	1	0	0
<addresses/NNS>	8	0	0	1	0
<their/PRP$> <waas/NNS>	10	0	0	1	0
<waas/NNS>	14	0	0	0	1
<multiple/NN><users/NNS>	15	0	0	0	1

Table 1 Example - Parsing Results

Stop words: (a user), (one), (users), (a way), (a list)		req. 1		req. 2	req. 3
		sent. 1	sent. 2	sent. 3	sent. 4
<imags/NN> <ii/NN>	0,7,11	1	0	1	1
<users/NNS> <'/POS> <assign/NN>	1	1	0	0	0
<waa/NN>	4,5,10,14	0	2	1	1
<user/NN>	6	0	1	0	0
<address/NN>	8	0	0	1	0
<multiple/NN><user/NN>	15	0	0	0	1

Table 2 Example - Stemming and Text Normalization

Stop words: (a user), (one), (users), (a way), (a list)	req. 1		req. 2	req. 3
	sent. 1	sent. 2	sent. 3	sent. 4
<imags/NN> <ii/NN>	1	0	1	1
<users/NNS> <'/POS> <assign/NN>	1	0	0	0
<waa/NN>	0	2	1	1
<user/NN>	0	1	0	0
<address/NN>	0	0	1	0
<multiple/NN><user/NN>	0	0	0	1
NPC-Sentence	1/3+1 =1.3	2/4+1 =1.5	1/3+1/4+1 =1.58	1/3+1/4+1= 1.58
NPC-Req	1.3 + 1.5 = 2.8		1.58	1.58

Table 3 Example - NPC-Sentence and NPC-Req

4.4 Intra-Section Level Complexity

The proposed $NPC - Cohesion$ metric is a normalized cluster size that can be calculated using the sum of all cluster sizes in a requirement section divided by the size of the requirements section. Here a cluster is defined as the collection of adjacent sentences in a requirements section that shares the same **NP** chunks. For example, if sentence 1 contains **NP** chunk A, sentence 2 contains **NP** chunk A and B, and sentence 3 contains **NP** chunk B, then the three sentences form a single cluster. The formula for $NPC - Cohesion$ is as follows.

$$NPC - Cohesion(S_j) = \begin{cases} \frac{\sum_{1 \leqslant i \leqslant M_j} ClusterSize(i,j)}{L_j - 1} & , L_j \geqslant 1 \\ 1 & , L_j = 1 \end{cases}$$

where M_j is the total number of clusters in section S_j, and L_j is the total number of sentences in section S_j.

If a requirements section consists of a single sentence ($L_j = 1$), the $NPC - Cohesion$ is 1. If all the adjacent sentences have common **NP** chunks, then the $NPC - Cohesion$ is also 1.

For example, Figure 1 shows three sections of the requirements. The first section contains three sentences, the second section contains two sentences, and the third section has one sentence. Section 1 has a cluster that covers the whole section, and the size of the cluster is two. Section 2 has a cluster that covers the whole section, and the size of the cluster is one. Section 3 does not have any cluster. Based upon the above formula, the values of the $NPC - Cohesion$ metric are as follows.

$$\text{NPC-Cohesion}(S_1) = 2/(3-1) = 1,$$
$$\text{NPC-Cohesion}(S_2) = 1/(2-1) = 1,$$
$$\text{NPC-Cohesion}(S_3) = 1$$

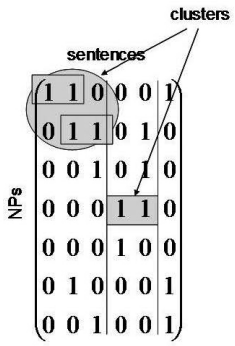


Figure 1 : Clusters of NP Chunks

4.5 Inter-Section Level Complexity

The proposed **NPC-Coupling** metric value is the sum of the spatial distances among **NP** chunks and clusters that share the same **NP** chunks. When calculating the distance involving **NP** chunks that form a cluster, the location of the cluster, or the centroid, represents the locations of its component **NP** chunks. In other words, once a cluster is formed, the cluster represents all the components NP chunks inside the cluster. One possible algorithm to calculate the **NPC-Coupling** metric is as follows.

As an extreme case, a requirements section may not have NP chunks in common with the rest of the document. In this case, the NPC-Coupling is zero. Another extreme case is a requirement section that does not contain clusters, but the requirements section shares common NP chunks with all sentences of the rest of the document. In this case, the NPC-Coupling is in the order of n_2, where n is the total number of sentences of a requirements document.

Figure 2 shows the calculation of **NPC-Coupling** using the same requirements sections in the previous example. The centroid of the cluster of the first section resides in the second sentence.

Algorithm 1 : To calculate the NPC-Coupling metric

```
The NPC-Coupling metric of section S_j:
x = 0,
remarks: handle clusters coupling
for all cluster i in section S_j do
  for all NP_k in cluster i do
    for all sentence l outside section S_j do
      if sentence l contains NP_k then
        calculate the distance between sentence l and the centroid of cluster i,
        add the distance to x,
      end if
    end for
  end for
end for
remarks: handle non-clusters coupling
for all NP_k of section S_j that does not belong to any cluster of S_j do
  for all sentence l in section S_j do
    if sentence l contains NP_k then
      for all sentence m outside section S_j do
        if sentence m contains NP_k then
          calculate the distance between sentence l and sentence m,
          add the distance to x,
        end if
      end for
    end if
  end for
end for
return x
```

The centroid of the cluster of the second section resides between sentence 4 and 5. Based upon the above formula, the values of the **NPC-Coupling** metrics are as follows.

$$\text{NPC-Cohesion}(S_1) = 4 + 3 + 2 + 4 + 3 = 16,$$
$$\text{NPC-Cohesion}(S_2) = 3 + 2 = 5,$$
$$\text{NPC-Cohesion}(S_3) = 4 + 4 + 3 = 11$$

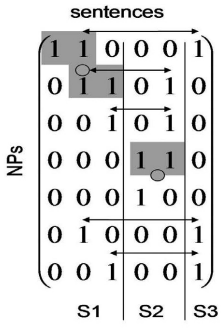


Figure 2 : Calculation of Coupling Metrics

4.6 A Composite Metric

It has been recognized that a single metric for a software product does not work. Multiple metrics provide multiple views of the subject, and each view serves its own purpose. Without carefully following engineering disciplines, operation, comparison, combination, or manipulation of different metrics can result in meaningless measures (Henderson-Sellers, 1996).

The need to have a single metric does arise sometimes. For example, a project may be constrained by its resources and can only afford to improve one or two high risk components of an application. In this case, combining carefully selected metrics using well-known engineering disciplines is necessary to provide a single indicator. Guidelines on combining different metrics are available in numerous research articles (Card and Glass, 1990; Fenton and Pleeger, 1997; Henderson-Sellers, 1996; Zuse, 1998).

To identify the low quality requirements sections, this research combines the cohesion and coupling metrics into a single indicator. The formula used in the research is as follows.

$$\text{NPC-Composite}(S_i) = \text{NPC-Cohesion}_i - a * (\text{NPC-Coupling}_i - b),$$
$$\textbf{for all } i, 1 \leqslant i \leqslant S, S \text{ is the total number of requirements sections,}$$
$$\text{where } b = min(\text{NPC-Coupling}_i) \text{ and } a = 1/(max(\text{NPC-Coupling}_i) - b).$$

The coefficients, a and b, are used to adjust the coupling metric so that (1) the measure falls in the range between-1 and 1 and (2) both the cohesion and coupling metrics use the same unit.

5. EMPIRICAL CASE STUDY

Case studies have reemerged as a viable empirical research evaluation (validation) methodology (Yin, 2003). Because only a few instances, or cases, are studied, the case researcher typically uncovers more variables than other empirical methodologies. Its capability of uncovering causal paths and identifying interaction effects is considered strength of case study research. In recent years there has been increased attention to the implementation of case studies in a systematic approach (Garson, 2006).

Although the proposed complexity metrics can be used to evaluate requirements documents for software applications, hardware products, and firmware components, only software requirements are evaluated in the case study in this research.

In this chapter a brief introduction of the case study methodology is presented. The research question (see Section 2) is analyzed and split into two sub-questions, and the case study is split into two phases accordingly. The first phase of the case study uses an exploratory case study to evaluate the proposed metrics to see whether the metrics can indeed measure the quality of the requirements statements. The second phase of the case study uses an explanatory case study to explain how and why the proposed metrics can identify the low quality requirements statements.

5.1 Case Study Methodology

The case study methodology (Yin, 2003) is an empirical research strategy commonly used in psychology, sociology, political science, social work, business, community planning, and economics. The methodology offers a way to investigate empirical research questions by following a set of pre-specified procedures. Our two case studies are used foranswering “how” and “why” research questions.

We have used twocase studies I and II:

—Case Study I is exploratory,

—Case Sdtudy II is explanatory.

Case study I is exploratory and is used to define the questions and hypotheses of the subsequent explanatory case study II. . Explanatory case study II explains the cause-effect relationships indicated in the research question (Yin, 2003).

Each of the two case studies adopted here consists of five components, which form a logic plan for the research design of case studies.

(1) A study's questions
(2) Study propositions, or hypotheses
(3) Unit(s) of analysis
(4) The logic linking of the data to the propositions
(5) The criteria for interpreting the findings

5.2 Two Phased Case Study

This case study is divided into two phases, Exploratory Case Study I and Explanatory Case Study II. Case Study I explores the properties of the NP chunk based complexity metrics, and explanatory Case Study II explains how and why the NP chunk based complexity metrics work.

The goal of this researchis to answer the constraint question about identifying low quality requirements statements in a requirements document with the specified constraints. This question can be divided into two sub-questions.

Q1. Can NP chunk based complexity metrics be more effective than the term based complexity metrics in terms of measuring requirements content goodness properties?

Q2. How and why NP chunk based complexity metrics measure the content goodness properties of requirements statements?

The two case studies, exploratory in phase I and explanatory in phase II, are designed to answer the two sub-questions, respectively. The first sub-question and hence the phase I case study is an evaluation of NP chunk based complexity metrics. If NP chunk based complexity metrics cannot produce consistent results as the term based complexity metrics proposed by Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011), there is no reason to perform further research on the research question. Ideally, the NP chunk based complexity metrics should be more effective than the term based complexity metrics; otherwise, the research provides little contribution to the research question.

The second sub-question and the corresponding phase II case study assume the phase I case study has positive outcomes. The phase II study then explains how and why the NP chunk based complexity metrics work. Evidence and findings to support the proposed metrics are presented one by one in this case study.

5.3 Exploratory Case Study - Phase I

The five components of the phase I case study are described as follows.

Study Question: The study question is "Can NP chunk based complexity metrics be more effective than the term based complexity metrics in terms of measuring requirements content goodness properties?"

Study Propositions, or Hypotheses: The purpose of the phase I case study is to determine whether the NP chunk based complexity metrics can measure content goodness properties of requirements documents. The term based complexity metrics published by Ricker (1995) reveal positive correlation to understandability, one of the content goodness properties of requirements documents.

The derived specific study hypotheses/propositions are as follows.

H1. Consistency: The NP chunk based complexity metrics are consistent with the term based complexity metrics.

Ricker proposed three term based complexity metrics: context, cohesion, and coupling, for requirements statements. However, the published metric values, or measures in Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011) focus mainly on the cohesion and coupling metrics. The only metrics that can

be compared against are cohesion and coupling metrics. Hence, the above hypothesis is divided into the two propositions **P1** and **P2**: **P1** for cohesion and **P2** for coupling.

P1. Cohesion: NPC-Cohesion, the NP chunk based cohesion metric, is consistent with the term based cohesion metric.

P2. Coupling: NPC-Coupling, the NP chunk based coupling metric is consistent with the term based coupling metric.

Unit(s) of Analysis: The unit of analysis for the phase I case study is the same requirements document of a Federal Aviation Agency (FAA) project used in Ricker (1995), (Boegh, 2008, Chung and Cesar, 2009, Costello and Liu, 19995, Davis et. al. 1993, Farbey, 1990, Wnu et. al. 2011) so that the results of the research can be compared.

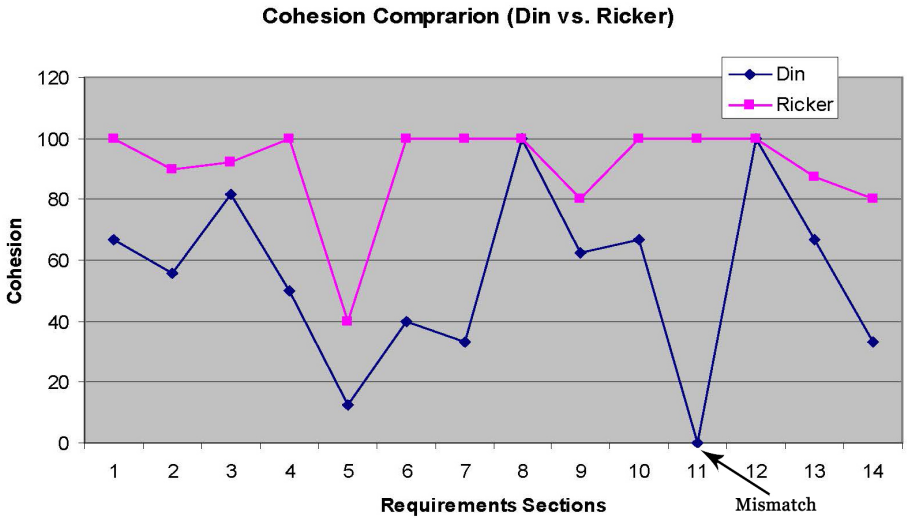


Figure 3 : Cohesion Values Between Two Methods

Cohesion Metrics: Based upon the proposed NPC-Cohesion metric defined previously, the NPC-Cohesion measures are depicted in Figure 3, together with the cohesion measures published in (Pleeger, 1993). It is clear that the two metrics are consistent with each other except in one section - section 11 of the FAA requirements document[1]

The mismatch between the two cohesion metrics can be explained as follows. Section 11 of the FAA requirements document consists of two sentences. Here are the sentences and the corresponding chunk parsing results.

Sentence 1: "The system shall provide flight plan outputs to a variety of operational positions, collocated processors, and remote facilities."

$(S:$
$(0 :< the/DT >< system/NN >)$
$< shall/MD >$
$< provide/VB >$
$(1 :< flight/NN >< plan/NN >< outputs/NN >)$
$< to/TO >$
$(2 :< a/DT >< variety/NN >)$

[1]The FAA requirements document has 26 sections. Ricker (1995) does not use all the 26 sections when calculating cohesion and coupling measures, and there are multiple sections that have coupling measures but not cohesion measures. Hence, the total number of sections varies on different diagrams.

$< of/IN >$
$(3 :< operational/JJ >< positions/NNS >)$
$<,/,>$
$< collocated/VBN >$
$(4 :< processors/NNS >)$
$<,/,>$
$< and/CC >$
$(5 :< remote/JJ >< facilities/NN >)$
$< ./. >$
)

Sentence 2: "The ACCC shall output data periodically, on request, or in accordance with specified criteria (NAS-MD-311 and NAS-MD-314)."

$(S :$
$(6 :< the/DT >< accc/NN >)$
$< shall/MD >$
$< output/VB >$
$(7 :< data/NNS >)$
$< periodically/RB >$
$<,/,>$
$< on/IN >$
$(8 :< request/NN >)$
$<,/,>$
$< or/CC >$
$< in/IN >$
$(9 :< accordance/NN >)$
$< with/IN >$
$< specified/VBN >$
$(10 :< criteria/NN >)$
$< (/(>$
$(11 :< nas - md - 311/NN >)$
$< and/CC >$
$(12 :< nas - md - 314/NN >)$
$<)/) >$
$< ./. >$
)

There are 13 **NP** chunks. It is clear that there are no common **NP** chunks between the two sentences. This is why the **NPC-Cohesion** metric gives a low cohesion measure for the above requirements section. On the other hand, Ricker uses terms to measure the cohesion of the section, and the word "output" appears in the first sentence as a noun, while the word "output" appears in the second sentence as a verb. Ricker's algorithm does not consider syntactic categories and hence links the two sentences.

It is believed that a word in different forms, i.e., verbs and nouns, in different sentences should not always be considered as cohesive, since the two words in the two forms can refer to two totally different objects. By closely examining the two sentences, it can be found that the word "output" in the two sentences indeed refers to two different things or two different concepts. Hence, the proposed cohesion metrics is more effective.

Although **NPC-Cohesion** is able to identify low cohesion requirements in the above example using syntactic categories of words, syntactic categories can sometimes mislead the analysis. For

example, the following two sentences are low cohesion sentences according to **NPC-Cohesion**.

"The computer program shall discretize the continuous function $f(t)$. Then, it shall approximate the integral using the discretization."

In this example, the two sentences are cohesive because both discuss "discretization." Unfortunately, **NPC-Cohesion** is not designed to reconcile different syntactic categories of the same word. The weakness of **NPC-Cohesion** should not a major concern because a requirements section typically contains more than two sentences. Furthermore, **NP** chunks are chosen as the element of measurement because they are the most content-bearing word classes. It will be rare that a section of document containing multiple sentences, and yet all the sentences are using different syntactic categories of the same **NP** chunk.

One remedy to the weakness of **NPC-Cohesion** is to parse verb phrase (**VP**) chunks. Part of the text normalization process is to transform words into its root form. Then "discretize" and "discretization" can be normalized as the same chunk. However, the incorporation of a second type of chunks, i.e., verb phrase (**VP**) chunks, to the proposed metrics substantially increase the complexity of the parser and hence the processing effort and time. The addition of the parsing process for **VP** chunks to cope with the weakness that rarely occurs does not seem to be cost effective. Hence, it was decided to focus on **NP** chunks for the research.

As indicated above, the evaluation criterion for the cohesion is whether the two sets of metrics are strongly consistent with each other. Since there is only one mismatch and the mismatch can be explained, the degree of consistency is strong. The evaluation criterion for linking the data to the proposition is whether the NP chunks based complexity metrics can provide additional information. Since the **NP-Cohesion** metric can differentiate word classes, the **NP-Cohesion** metric does provide additional information.

Coupling Metrics: The coupling measures based on the **NPC-Coupling** metric and the coupling metric in (Pleeger, 1993) are depicted in Figure 4. The two metrics display consistent results except in one section - section 4 of the requirements document.

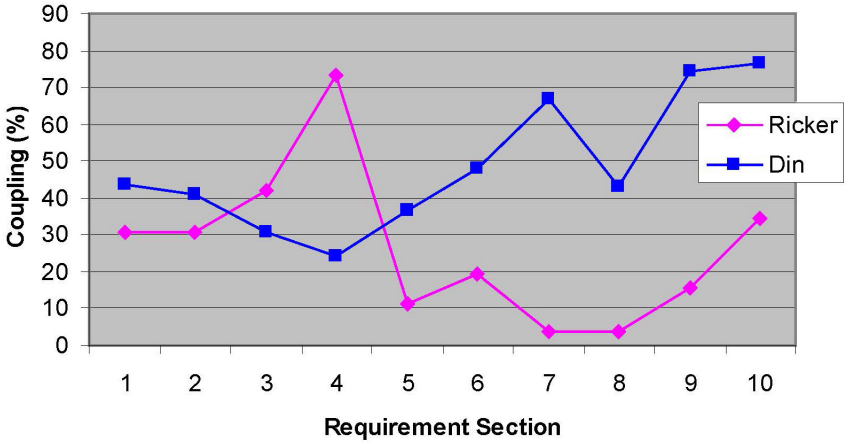


Figure 4 : Coupling Values between Two Methods

The discrepancy between the two coupling metrics can be explained as follows. Section 3 is titled as "routing processing", and Section 4 is titled as "additional routing processing." Since the fourth section is a supplement to the third section, its coupling in Ricker's method is very high.

On the other hand, the coupling value for section 4 is low for the **NPC-Coupling** metric because the spatial distance between the two sections is low. In other words, the effect of spatial

distance is counted in the **NP-Coupling** metric, while Ricker's method does not consider the spatial distance.

The evaluation criterion for cohesion is whether the two sets of metrics are strongly consistent with each other. Since there is only one mismatch and the mismatch can be explained, the degree of consistency is strong. The evaluation criterion for linking the data to the "additional information" proposition is whether the **NP** chunks based complexity metrics can provide additional information. Since the **NPC-Coupling** metric can measure spatial distance, the **NPC-Coupling** metric does provide additional information.

Sensitivity/Accuracy: The **NPC-Cohesion** metrics are relative measures. They are normalized and fall in the range of 0 to 1. Comparing such relative measures derived from different requirements documents is not logical. In other words, it is not appropriate to compare the sensitivity and accuracy of the **NPC-Cohesion** metrics with Ricker's metrics.

Although the **NPC-Coupling** metrics are based upon spatial distance between **NP** chunks and they are not normalized, comparing it with Ricker's metric which uses different units of measurement does not seem to be logical either.

Summary: Based upon the above analysis, it can be concluded that the derived data from the case study met the evaluation criteria for the consistency hypothesis (**H1**).

5.4 Explanatory Case Study - Phase II

Study Question: The phase II study question is "How and why NP chunk based complexity metrics measure the content goodness properties of requirements statements?"

Study Propositions, or Hypotheses: In this case study design the study question stated above can be decomposed into three specific study propositions.

P3. NP Chunk Counts: The **NP-Count**, as a simple form of complexity metric, can measure the content goodness properties of the requirements statements.

P4. Cohesion: NP chunk based cohesion complexity metrics such as the NPC-Cohesion metric can measure the content goodness properties of the requirements statements.

P5. Coupling: NP chunk based coupling complexity metrics such as the NPC-Coupling metric can measure the content goodness properties of the requirements statements.

The evaluation criteria for the linking of derived data from the case study to the above three propositions is whether the linking can explain the cause-effect relationship. For simplicity reason, the cause-effect relationship can be categorized into three ordinal values: strong, medium, and weak/no cause-effect relationship.

Unit(s) of Analysis: In the phase II research design, the unit of analysis is also a requirements document. Two sources of requirements documents are used for the case study: (1) four versions of the Interactive Matching and Geocoding System II (IMAGS II) requirements documents for U. S. Bureau of Census and (2) the FAA requirements document used in the phase I study.

The Logic Linking of the Data to the Propositions and Criteria for Interpreting the Findings: In this section the three major categories of metrics, sentence/requirements complexity metrics, cohesion metrics, and coupling metrics, are discussed separately.

NPC-Sentence (Sentence Level Complexity Metrics): The **NPC-Sentence** metric proposed in a previous section is basically a way to count the **NP** chunks, and it can be used to identify complex requirements.

Section 3.4 of the IMAGS II requirements document is used to illustrate how the **NPC-Sentence** metric works. The complexity measures are first obtained from Section 3.4 of both the version 2 and version 3 of the requirements documents. The complexity measures are then compared between the two versions of the requirements. The **NPC-Sentence** measures of Section 3.4 of the version 2 requirements document are depicted in Figure 5, where sentence 10, 11, and 12 show high degree of complexity. Subsequent iteration of requirements review indeed identified those three sentences as "difficult to understand". A new set of sentences were then developed in the version 3 of the requirements document. The comparison of the two versions of the requirements section shows that the complexity measures of the three sentences are improved

in the new version of the requirements document.

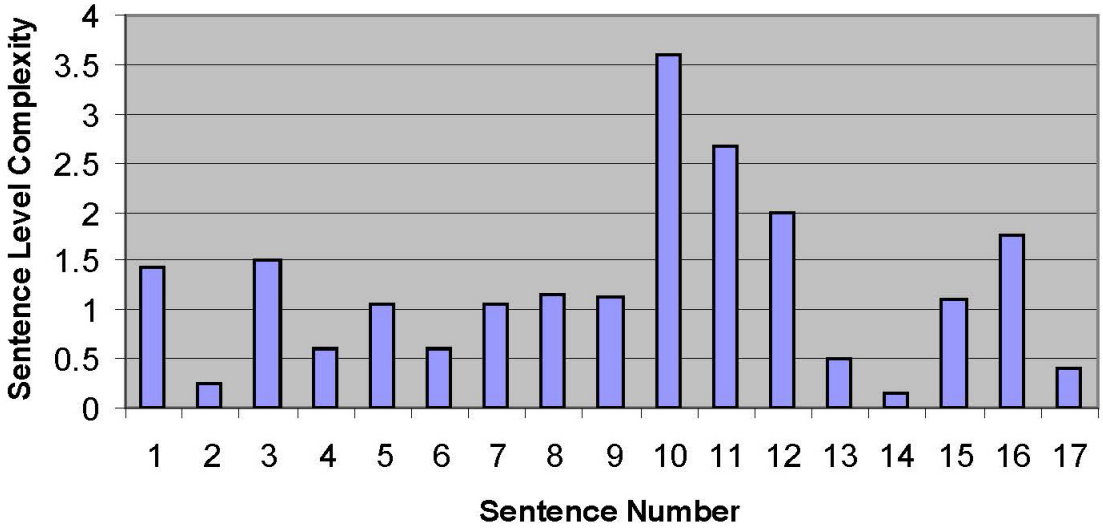


Figure 5 : Sentence Complexity of IMAGS Version 2 Requirements

The version 3 requirements document includes not only the changes to sentence 10 to 12 but also other changes irrelevant to sentence 10 to 12. To show the effects of the changes to the three sentences, irrelevant changes are removed from the version 3 of Section 3.4 of the requirements document, and the resulting requirements section is named as the version 2-3 requirements. The comparison of the two versions of the requirements section depicted in Figure 6 shows that the complexity measures of the three sentences are improved in the new version of the requirements document.

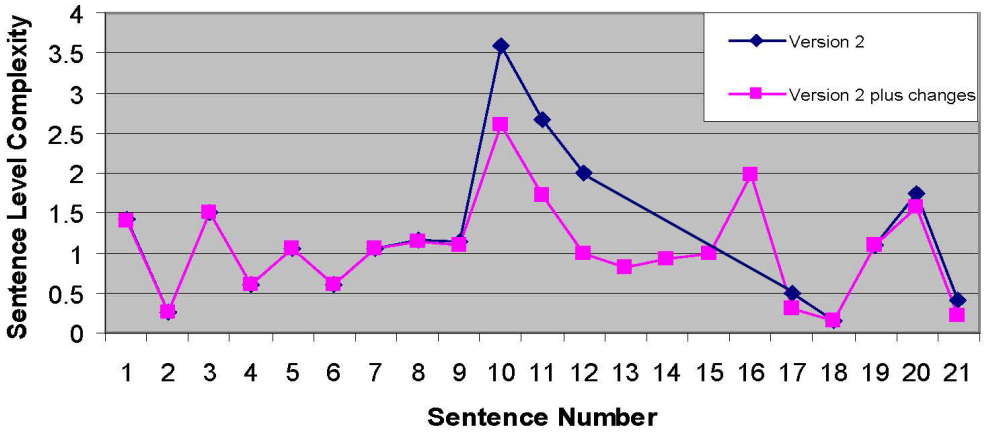


Figure 6 : Sentence Complexity of IMAGS Version 2 and Version 2-3 Requirements

NPC-Req (Requirements Level Complexity Metrics): The **NPC-Sentence** can be aggregated to the **NPC-Req** because a requirement consists of one or more sentences. Another section of the IMAGS II requirements document is used to illustrate the capability of the **NP** chunk complexity metrics at the requirements level.

The **NP-Req** metric values are compared between two versions of Section 3.2 of the IMAGS II requirements document: version 2 and 3. During the third iteration of the requirements gathering phase, four modifications are made, and the version 2 **NPC-Sentence** measures do not show clearly which sentences should be improved or re-written. The **NPC-Sentence** for the version 2 document is shown in Figure 7.

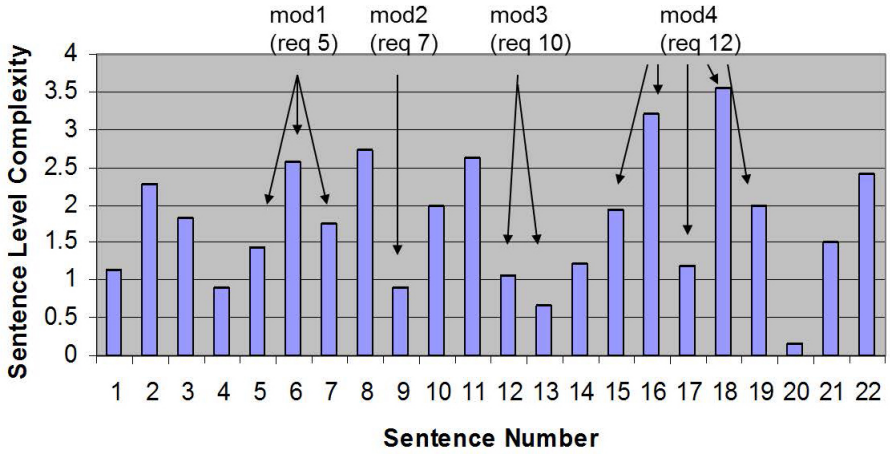


Figure 7 : Sentence Complexity of IMAGS Version 2 Requirements

If **NPC-Req** is used to aggregate the **NPC-Sentence** metric values, the resulting complexity measures are shown in Figure 8. It is observed now that requirements 5 and 12 are the most complex requirements in the section. This coincides two of the modifications shown on the version 3 requirements document.

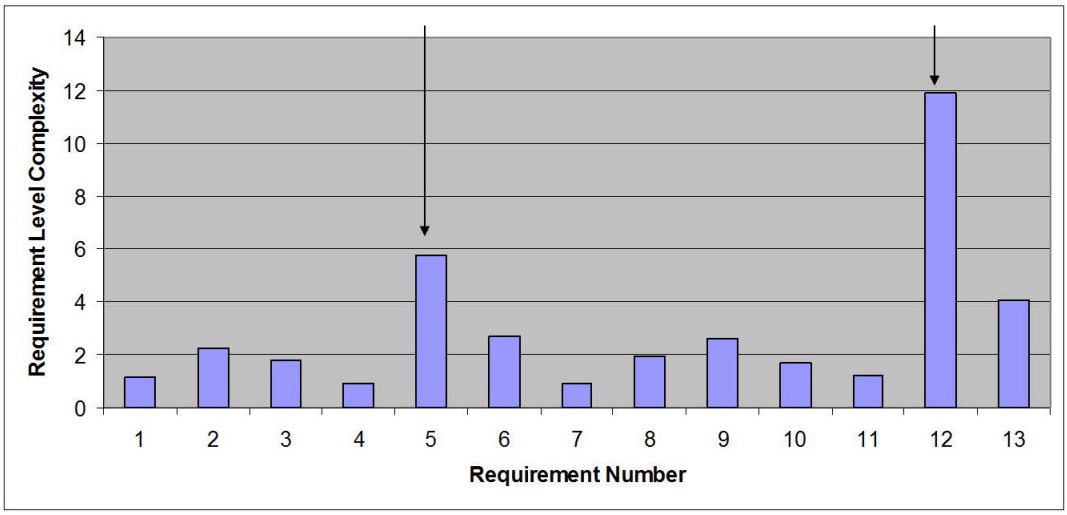


Figure 8 : Requirements Complexity for IMAGS Version 2 Requirements

Counts, Cohesion, and Coupling: Counting the elements of measurement is the simple form of complexity metrics. The question is how the different ways of counting NP chunks relate to

each other. Positive correlations among the simple NP chunk count, sentence count, and cluster counts exists (see Figure 9). This is consistent with intuition.

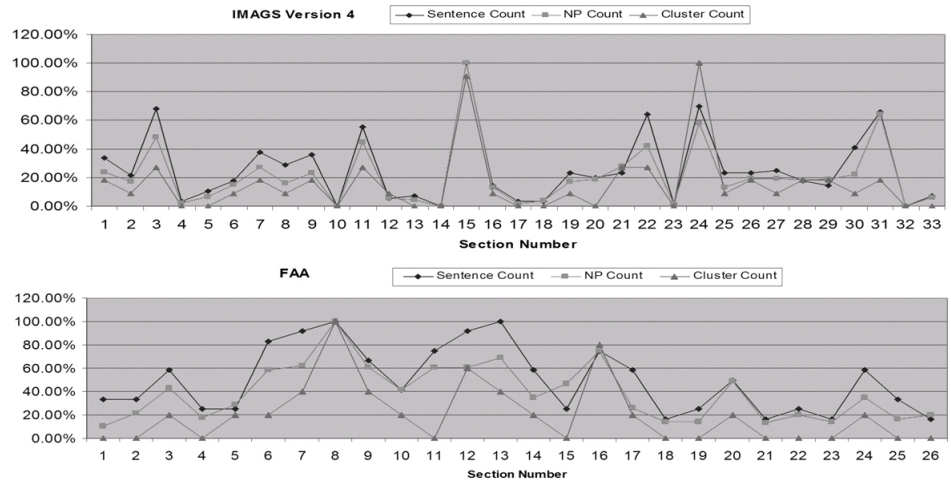


Figure 9 : Relationships Among Sentence, NP, and Cluster Counts

This research (Din, 2007, 2008; Din and Rine, 2008, 2012) also shows that the above three count metrics and the **NPC-Cohesion** metric have no correlation between them (see Figure 10). On the other hand, the above three count metrics are correlated to the **NPC-Coupling** metric, although the degrees of correlations varies among those counts (see Figure 11).

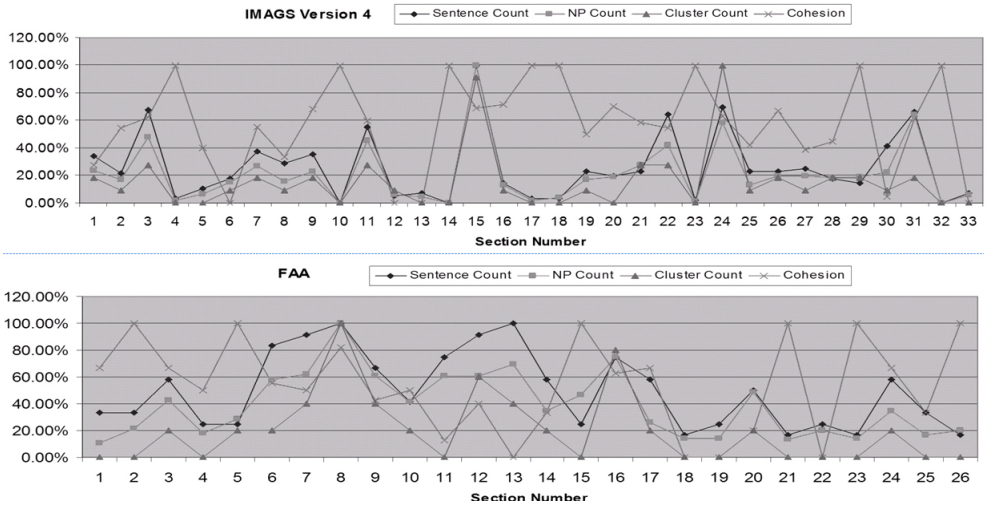


Figure 10 : Relationships between Counts and Cohesion

The correlation between **NP** chunk counts and the cohesion/coupling metrics can be explained as follows. Assuming a requirements section contains an **NP** chunk X, the addition of a new X in another section of the requirements document increases the coupling metric by the spatial distance between the new NP chunk and the requirements section. Deletion of an NP chunk has the opposite effect. Hence, those NP chunk counts are correlated to the NP chunk based coupling metrics.

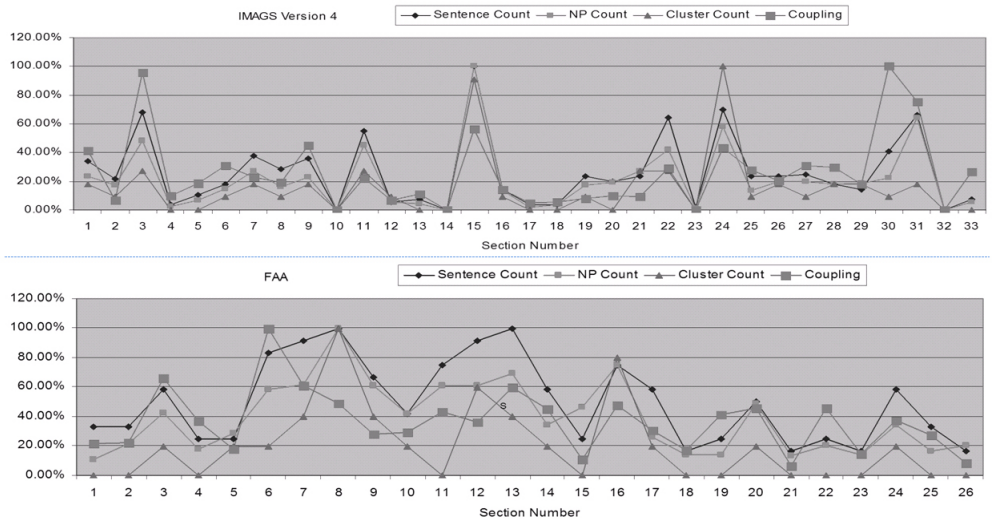


Figure 11 : Relationships between Counts and Coupling

The addition or deletion of an **NP** chunk has very little effects on the cohesion metric because the **NPC-Cohesion** metric is based upon the size of the cluster of the **NP** chunk. Unless the addition or deletion of an NP chunk happens at the boundary of a cluster, they may not affect the values of the **NPC-Cohesion** metric.

Based upon the above analysis, it is observed that **NP** chunk counts can measure the complexity of requirements statements and hence show the strong cause-effect relationship to the content goodness properties of requirements statements. In other words, the proposition **P3** is supported.

Cohesion and Coupling Metrics: The cohesion measures for the version 1 to version 4 of IMAGS II requirements documents can be illustrated by the differences between the two adjacent versions of IMAGS II requirements documents. The results are three sets of values depicted in Figure 12. It can be seen that the iteration from version 1 to version 2 and from version 2 to version 3 have substantial changes. On the other hand, the iteration from version 3 to version 4 is bounded in a relatively narrow range.

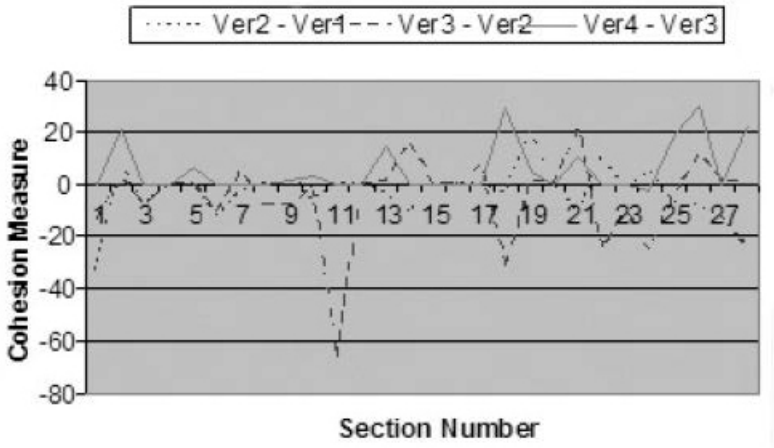


Figure 12 : Cohesion Measure Trend

Similar to the cohesion metrics, the coupling metrics also show that the iteration from version

3 to version 4 of the requirements stays in a relatively narrow range (see Figure 13).

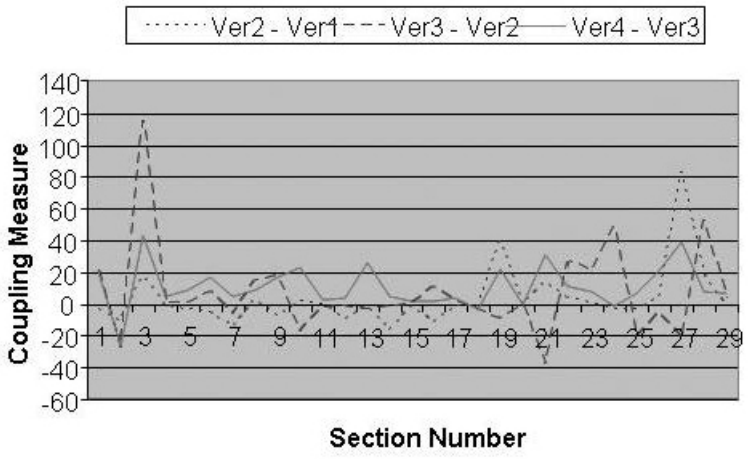


Figure 13 : Coupling Measure Trend

NPC-Composite: In addition to the traditional cohesion and coupling metrics, it is valuable to have a single metric to measure the complexity of requirements sections. Figure 14 and 15 show the NPC-Composite measures of the IMAGS and FAA projects, respectively. For the IMAGS II project, NPC-Composite shows that Section 30 is the worst requirements section. After examining the requirements document, it is found that Section 30 is for reports. Reports requirements typically reference all other sections and are independent of each other. The next low quality requirements sections are Section 3, 6, and 33. Section 3 is the requirements for the overall operations, which includes multiple requirements for suspend and shutdown some operations but leaves others operational. Section 3 indeed contains complicated requirements. Section 6 discusses the address import, and Section 33 provides performance related requirements. These two sections do not seem to be complicated.

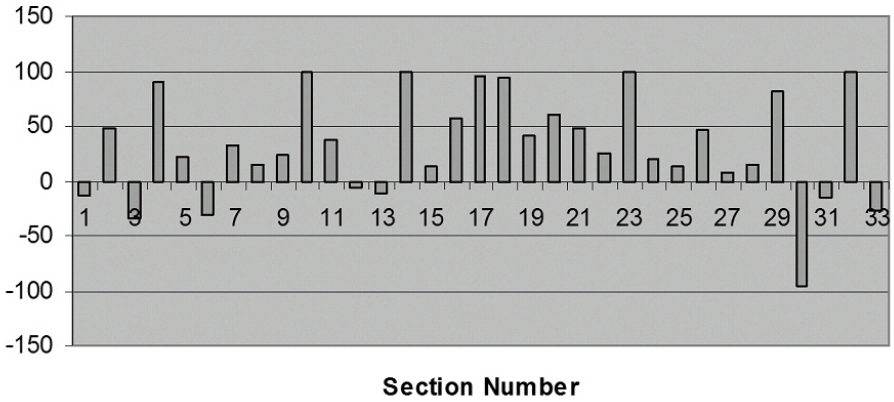


Figure 14 : Composite Measures for IMAGS

For the FAA project, the most complicated requirements section indicated by NPC-Composite is Section 13, which has the highest number of sentences (see Figure 9) and the cohesion value for Section 13 is zero (see Figure 10). This section is indeed complicated. The next set of low

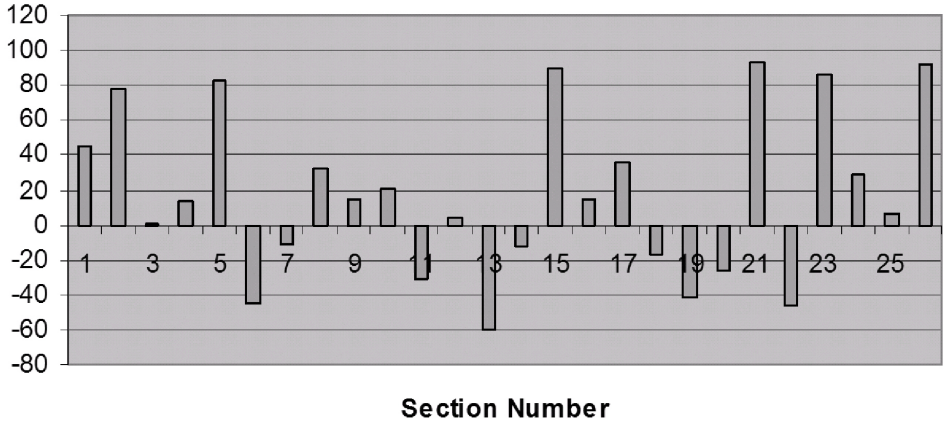


Figure 15 : Composite Measures for FAA

quality requirements sections are Section 6, 19, and 22. Although the NP chunk count of Section 6 is not the highest, the coupling value is the highest. The problem with Section 19 and 22 is evident by their zero cohesion value.

The Phase I exploratory case study gives evidence that the proposed NPC-Cohesion and NPC-Coupling metrics are consistent with Ricker's metrics which are correlated to understandability, a content goodness property, of requirements statements. This section again reports evidence that the NPC-Composite metric has a strong cause-effect relationship to the content goodness properties of requirements statements. In other words, NPC-Cohesion supports the proposition P54 and NPC-Coupling supports the proposition P5.

Summary: Based upon the evidence discussed above, the hypothesis that the proposed complexity metrics can identify low quality requirements statements in a requirements document can be asserted.

6. SUMMARY OF CONTRIBUTIONS

This research derived from (Din, 2007, 2008; Din and Rine, 2008, 2012) made two contributions: (1) the invention of a suite of complexity metrics to measure the content goodness properties of requirements documents and (2) the empirical case study to evaluate the invented suite of complexity metrics.

The invented complexity metrics are researched and developed to identify low quality requirements statements in SRS's. These metrics are based on the **NP** chunks in SRS's. In the empirical two phased case study, it is concluded that the proposed metrics can measure the content goodness properties of requirements statements.

This research provides evidence for the feasibility of using **NP** chunks as the elements of measurement for complexity metrics. In addition the invented suite of complexity metrics provides requirements engineers and managers with a tool to measure the quality of the requirements statements. These metrics can be use to identify low quality requirements statements. They can also be used to identify requirements statements and requirements sections that may require more rigorous testing. Potential flaws and risks can be reduced and dealt with earlier in the software development cycle.

At a minimum, these metrics should lay the groundwork for automated measures of the quality of the requirements statements in SRS's. Because those metrics are constructed by a software tool, their measures are easy to collect, a vital characteristics for a quality measurement program (Pleeger, 1993).

REFERENCES

Abney, S., and Abney, S.(1991). Parsing By Chunks In *R. Berwick, S. Abney, and C. Tenny, editors, Principle-Based Parsing. Kluwer Academic Publishers, 1991.*

Basili, V. R. (1980). Qualitative Software Complexity Models: A Summary, *Tutorial on Models and Methods for Software Management and Engineering. 1980.*

Briand, L. C., Daly, J. W., Wust, and J. K. (1999). A Unified Framework for Coupling Measurement in Object-Oriented Systems. *IEEE Transactions on Software Engineering, 25:91 121, 1999*

Cant, S., Jeffery, D. R, Henderson-Sellers, and B (1995). A conceptual model of cognitive complexity of elements of the programming process. *Information and Software Technology, 37(7):351 362, 1995.*

Card, D. N., and Glass, R. L. (1990). Measuring Software Design Quality *Prentice-Hall, 1990.*

Darcy, D. P., and Kemerer, C. F. (2002). Software Complexity: Toward a Unified Theory of Coupling and Cohesion. *not working here, 8, February 8 2002*

DeMarco, T. (1982). Controlling Software Projects. *Yourdon Press, Englewood Cliffs, NJ, 1982.*

Din, C. Y. (2008). Requirements Content Goodness and Complexity Measurement Based On NP Chunks. *PhD thesis, George Mason University, Fairfax, VA, 2007, Reprinted by VDM Verlag Dr. Muller, 2008*

Din, C. Y., and Rine, D. C.(2008). Requirements Content Goodness and Complexity Measurement Based On NP Chunks. *Proceedings, Complexity and Intelligence of the Artificial Systems: Bio-inspired Computational Methods and Computational Methods Applied in Medicine, WMSCI 2008 Conference.*

Din, C. Y., and Rine, D. C.(2012). Requirements Metrics for Requirements Statements Stored in a Database. *Proceedings of the 2012 International Conference on Software Engineering Research and Practice, SERP'12, July 16-19, 2012, pp. 1-7.*

Dunsmore, H. E.(1984). Software Metrics: An Overview of an Evolving Methodology. *Information Processing and Management, 20(1-2):183 192, 1984.*

Evangelist, W.(1983). Software Complexity Metric Sensitivity to Program Structuring Rules. *Journal of Systems and Software, 3(3):231 243, 1983.*

Fagan, M.(1986). Advances in Software Inspections. *IEEE Transactions in Software Engineering, 12, 7: 744-751 July 1986.*

Fenton, N. E., Neil, and Neil, M. (1999). A Critique of Software Defect Prediction Models. *IEEE Transactions on Software Engineering, 25:675 689, 1999.*

Fenton, N. E., and Neil, M.. Software metrics: roadmap. In *Proceedings of the International Conference on Software Engineering (ICSE), pages 357 370, 2000.*

Fenton, N. E., and Pleeger, L. (1997). Software Metrics: A Rigorous and Practical Approach. *International Thomson Computer Press, Boston, MA, 1997. 2nd ed.*

Garson, G. D. (2006). Public Information Technology and E-Governance: Managing the Virtual State. *Jones & Bartlett Pub., January 2006.*

Graesser, A. C., McNamara, D. S., Louwerse, M. M., and Cai, Z. (2004). Coh-Metrix: Analysis of Text on Cohesion and Language. *Behavioral Research Methods, Instruments, and Computers, 36(2):193 202, 2004.*

Henderson-Sellers, B. (1996). Object-Oriented Metrics textendash Measures of Complexity. *Prentice Hall PTR, New Jersey, 1996.*

Kemerer, C. F. (1998). Progress, Obstacles, and Opportunities in Software Engineering Economics. *Communications of ACM, 41:63 66, 1998.*

Kitchenham, B. A., Pleeger, S. L, Fenton, and N. E (1995). Towards a Framework for Software Measurement Validation. *IEEE Transactions on Software Engineering, 21:929 943, 1995.*

Klemola, T. (2000). A Cognitive Model for Complexity Metrics. *volume 13, June 13 2000.*

Lee, S. (2003). Proxy Viewpoints Model-Based Requirements Discovery. *PhD thesis, George Mason University, Fairfax, VA, 2003.*

Levitin, A. V. (1986). How to Measure Size, and How Not to. *volume 10, pages 314 318, Chicago, Oct 8-10 1986. IEEE Computer Society Press.*

Li, H. F., and Cheung, W. K. (1987). An empirical study of software metrics. *IEEE Transactions on Software Engineering, 13(6):697 708, 1987.*

McNamara, D. S. (2001). Reading both high and low coherence texts: Effects of text sequence and prior knowledge. *Canadian Journal of Experimental Psychology, 55:51 62, 2001.*

McNamara, D. S., Kintsch, E, Songer, N. B, Kintsch, and W (1996). Are good texts always better? Text coherence, background knowledge, and levels of understanding in learning from text. *Cognition and Instruction, 14:1 43, 1996.*

Pleeger, S. L.(1993). Lessons learned in building a corporate metrics program. *IEEE Software, 10(3):67 74, 1993.*

Purao, S., and Vaishnavi, V. (2003). Product Metrics for Object-Oriented Systems. *ACM Computing Surveys, 35(2):191 221, 2003.*

Ricker, M. (1995). Requirements Specification Understandability Evaluation with Cohesion, Context, and Coupling. *PhD thesis, George Mason University, Fairfax, VA, 1995.*

Schneider, R. E. (2002). Process for building a more effective set of requirement goodness properties. *PhD thesis, George Mason University, Fairfax, VA, 2002.*

Schneider, R., and Buede, D. (2000). Properties of a High Quality Informal Requirements Document, In *Proceedings of the Tenth Annual International Conference on Systems Engineering, INCOSE, July 16-20, 2000b, 377-384.*

Schneider, R., and Buede, D. (2000). Criteria for Selecting Properties of a High Quality Informal Requirements Document, In *Proceedings of the International Conference on Systems Engineering, Mid-Atlantic Regional Conference, INCOSE-MARC, April 5-8, 2000a, 7.2-1 to 7.2-5.*

Sommerville, I. (2006). Software Engineering: update. *Addison Wesley, 8 edition, 2006.*

Weyuker, E. (1988). Evaluating Software Complexity Measures. *IEEE Transactions Software Engineering, 14(9):1357 1365, 1988.*

Wilson, W. M., Rosenberg, L. H, Hyatt, and L. E (1996). Automated Quality Analysis Of Natural Language Requirement Specifications. In *Fourteenth Annual Pacific Northwest Software Quality Conference, October 1996.*

Yin, R. K. (2003). Case study research. *Sage Publications, Thousand Oaks, CA, 3 edition, 2003.*

Zuse, H. (1998). A Framework of Software Measurement. *Walter de Gruyter, Berlin/New York, 1998.*

Begh, J. (2008). A New Standard for Quality Requirements. *IEEE Software, March/April 2008 pp. 57-63.*

Chung, L., and Cesar, J. (2009). On Non-Functional Requirements in Software Engineering. *Conceptual Modeling: Foundations and Applications, Lecture Notes in Computer Science, Volume 5600, 2009, pp 363-379.*

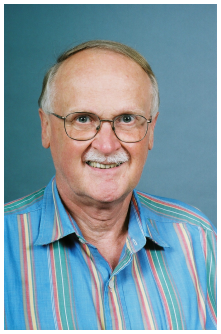
Costello, R., and Liu, D. (1995). Metrics for requirements engineering. *Journal of Systems and Software, Volume 29, Issue 1, April 1995, Pages 3963.*

Davis, A., Overmyer, S., Caruso, J., Dandashi, F., and Dinh, A. (1993). Identifying and measuring quality in a software requirements specification. In *Proceedings Software Metrics Symposium, 1993, First International, 21-22 May 1993, pp.141 152.*

Farbey, B (1990). Software quality metrics: considerations about requirements and requirement specifications. *Information and Software Technology, Volume 32, Issue 1, JanuaryFebruary 1990, Pages 6064.*

Wnu, K., Regnell, B., and Berenbach, B. (2011). Scaling Up Requirements Engineering Exploring the Challenges of Increasing Size and Complexity in Market-Driven Software Development. *Requirements Engineering: Foundations for Software Quality, Lecture Notes in Computer Science Volume 6606, 2011, pp 54-59 .*

Chao Yun Din is a senior consultant with NuWave Solutions. He has been a practitioner of software engineering for more than 20 years. His research interests include Requirements Engineering, Software Development Process, and Cyber Security. He received a PhD in Information Technology with a study in software engineering at George Mason University in the Volgenau School of Information Technology and Engineering. He is a member of IEEE and ACM.



Dr. David C. Rine is Professor Emeritus of Computer Science, Volgenau School of Information Technology and Engineering at the George Mason Unviersity, USA. He received his Ph.D. in 1970 from The University of Iowa in the Mathematical Sciences Division with a Dissertation in Computer Science. Dr. Rine has received numerous awards from computer science and engineering societies and associations, including the IEEE Centennial Award, the IEEE Pioneer Award, the IEEE Computer Society Meritorious Service Award, the IEEE Computer Society Special Award, and the IEEE Computer Society 50th anniversary Golden Core Award. He has also been a multiple-time recipient of the IEEE Computer Society Honor Roll (and Distinguished Technical Service) Award and the IEEE Computer Society Certificate of Appreciation Award. His core areas of research are software systems engineering, computational science, information systems, computer science and science and engineering education. He has produced over 300 research papers and graduated over 40 PhD students during his career. The Army, Air Force, Navy, NATO, National Science Foundation, NASA, IEEE, ACM, IBM, AFIPS and many industrial organizations have funded Dr. Rine's scientific research and development work. IEEE, AFIPS, The College Board, Air Force, Educational Testing Service, George Mason University have funded his educational research and development. He has directed research and development projects that have integrated combinations of science, engineering and educational models. He directed the first model curriculum in software engineering as well as the first Advanced Placement programs in computer science.