

Ten Years of REFSQ: A Quantitative Analysis

Vincenzo Gervasi¹, Erik Kamsties², Björn Regnell³, and
Camille Ben Achour-Salinesi⁴

¹ University of Pisa, Dipartimento di Informatica

² University of Essen, Institut für Informatik & Wirtschaftsinformatik

³ Lund University, Department of Communication Systems

⁴ University of Paris 1, Centre de Recherche en Informatique

Abstract. In its first ten editions, the REFSQ workshop has published over 150 papers on various aspects of Requirements Engineering. In this paper, we apply statistical lexical analysis techniques to this large corpus, in order to characterize the main topics and trends that have emerged from ten years of research in this area.

The results provide both an historical perspective (and some lesson) on the evolution of favored research topics, as well as suggestions for the future, in particular about topics that have not been adequately addressed so far.

1 Introduction

Requirements Engineering (RE) has repeatedly shown to be of paramount importance in achieving high-quality software, and RE has matured as an independent research field over the last decade [1]. An important arena for the establishment and development of RE research is the annual “International Workshop on Requirements Engineering: Foundation for Software Quality” (REFSQ), where novel and on-going research is actively and interactively assessed and discussed. In 2004, we celebrate ten REFSQ events that have provided a regularly recurrent opportunity for RE researchers in both academia and industry to stay up-to-date with the RE frontier and to get feedback on current work as well as input to further research. The workshop format includes a structured feedback process, where dedicated discussants initiate prepared debates on all technical papers and position papers, and REFSQ is characterized by its rigorous referee process, its openness to new community members, as well as its dedication to exchange with industrial practitioners. Many high-quality papers from past REFSQ events have resulted in extended papers published in journals and at conferences. An extensive account of the history and scientific impact of REFSQ papers is provided in [8].

When celebrating the tenth anniversary of REFSQ, it is interesting to analyze what have been the different subjects of interests over the past decade. This paper presents a study with the aim of investigating the trends and topics among ten years of REFSQ papers in an objective and quantitative way. The study

is based on a quantitative analysis using various lexical methods and takes a statistical approach to meta research in RE.

The main research questions of the presented study are:

1. What are the frequencies of different research topics?
2. How have research topic frequencies changed over time?

The objectives behind trying to answer these questions are to be able to quantitatively and objectively investigate constant research themes and transient research trends, as well as to provide input to a qualitative discussion of how RE research has been developing and should develop in the future. We also believe that we have found a way of characterizing what is special about REFSQ with respect to its particular focus on requirements as a foundation for software quality. It is not difficult to envision how the presented meta research approach based on lexical methods can be extended to cover more of the RE literature, as well as to enable a comparison with general Software Engineering topics and trends.

The paper is structured as follows. Section 2 explains how data have been collected and provides a methodological account of the way in which the lexical analysis of topics and trends has been conducted. Section 3 presents the results from the analysis as well as an interpretation of the results in relation to RE research in general. Finally, Section 4 concludes the paper and provides a tentative extrapolation into future trends of RE research based on the differences between early and recent REFSQ papers.

2 Data collection and analysis approach

2.1 Data collection

The REFSQ corpus analyzed in this study consists of 10 volumes of proceedings, one for each of the 10 editions of the workshop in the period 1994-2004 (in 1996 the workshop was not held). Each volume included from 11 to 20 papers (both longer full papers and shorter position papers), plus editorial contributions. To avoid biasing the study based on the various editors' preferences, these editorial papers have not been included in the analysis.

Overall, 153 technical papers were considered (see Table 1). Not all of them, though, could be recovered: Some of the earlier papers (from 1994 and 1995) were not available in electronic form, and were recovered in part from such copies of the proceedings as were available (the proceedings had been photocopied from the authors' manuscripts, and not printed), or directly from the authors. Furthermore, not all recovered papers could be analyzed, due to a variety of reasons, including: (i) papers that were available electronically only in proprietary formats of long-demised document preparation systems; (ii) papers that were provided in standard formats, but that had errors of various kind (e.g., errors in

Postscript code); (iii) papers that were provided in so-called “optimized” PDF format, that makes recovering the original text extremely difficult¹.

Year	Published (papers)	Recovered (papers)	Analyzed (papers)	Size (words)
1994	13	13	4	11,327
1995	11	9	6	46,700
<i>No REFSQ in 1996</i>				
1997	16	16	10	48,273
1998	20	20	8	24,633
1999	15	15	15	64,308
2000	14	14	11	44,477
2001	19	19	12	47,106
2002	16	16	16	68,009
2003	14	14	12	48,662
2004	15	15	14	67,105
Total	153	151	108	470,600

Table 1. Composition of the REFSQ corpus.

The raw text of 108 of the papers was obtained by a variety of techniques. For older papers, optical character recognition techniques have been employed; the scanned text was corrected manually and saved in purely textual format. Papers that were available in “source” form (e.g., \LaTeX sources, Microsoft Word documents, FrameMaker files, etc.) were directly saved in text format, or manually stripped of their mark-up commands to obtain the raw text. Finally, papers that were available electronically, but only in printable form (Postscript or PDF files), were converted to text by applying a number of different tools, including Acrobat Reader from Adobe, `ps2ascii` from the Ghostscript suite [3], `pstotext` developed at DEC in the Virtual Paper project [10], and `prescript` [7]. Unfortunately, we were unable to obtain the raw text of the remaining 45 papers. While overall we are considering a sample including 70.6% of the entire corpus, that could be considered an abundant sample for a statistical study, the sample for the 1994 edition (with only 4 papers analyzed out of 13 published) is far less representative than the average, and the lack of data for that year could introduce a systematic skew in our analysis.

¹ “Optimization” of PDF files consists in removing from each font used in the document all characters, symbols and glyphs that do not actually appear in the document. Used characters are moved around, towards low-numbered codes, so that the encoding vector of a font becomes essentially random. This is equivalent to applying a font- and style-dependant, mono-alphabetic, non distance-preserving cryptographic substitution to the clear text of the document.

2.2 Analysis technique

The main goal of our analysis was to characterize, by using objective lexical measures, which concepts were addressed in REFSQ’s papers, and how the interest in various topics has evolved during the last ten years. To this end, we needed a definition of *concept* that could be traced back to purely lexical features of the papers.

We assumed that concepts could be adequately represented by n -grams, i.e., by sequence of consecutive words that appear in the text. In particular, we limited our study to $(1, 4)$ -grams, representing sequences of 1 to 4 consecutive words. The upper limit to 4 words is a compromise between the computational resources needed for the analysis and the expressiveness of the n -gram. Most relevant terms, e.g. “case study”, “hierarchical system design” or “requirements quality models” are well below this limit.

Each volume of proceedings, obtained by collating the text of all the retrieved papers for a given year, was processed to extract all $(1, 4)$ -grams (with some care to remove punctuation and other “noise”). Overall, 885,545 unique $(1, 4)$ -grams were obtained from the 10 volumes. As Table 2 shows, only around 22,000 unique words appeared in the corpus — a rather reduced lexicon, but not uncommon in highly technical, scientific writing. Table 2 also shows that, on average, each 4-gram appeared only 1.11 times in the text; in other terms, 4-grams were almost unique occurrences. In addition, the corpus contained 445,207 unique 5-grams, only marginally more than the 423,048 4-grams. These findings confirm our assumption that $(1, 4)$ -grams are sufficiently representative of the text.

	Occurrences		
	Count	Avg	Stdev
1-grams	22,671	20.72	31.76
2-grams	185,112	2.54	2.34
3-grams	352,927	1.33	0.56
4-grams	423,048	1.11	0.19
Total	983,758	1.91	1.55

Table 2. Statistical features of $(1, 4)$ -grams extracted from the whole REFSQ corpus.

We ran several different analyses on our collection. The outcome of these is reported in Section 3; here we introduce the rationale and technicalities of each kind of analysis.

Topics fingerprinting First, we wanted to identify the most “important” topics (i.e, the most frequently recurring n -grams) in the whole REFSQ corpus. To this end, we identified in the whole set of $(1, 4)$ -grams extracted from the corpus those that satisfied all the following criteria:

1. had a number of occurrences higher than the average plus six times the standard deviation for their length, according to Table 2 (we call n -grams satisfying this criterion *significant*);
2. were not formed exclusively by stopwords or other closed-class words² (i.e., they included at least a name, a verb, an adjective or an adverb);
3. were not duplicates, in the sense that, once stripped of any stopword and stemmed, the remaining n -gram did not already appear in the list.

Finally, the set was manually filtered to remove a few common, but not significant, n -grams (e.g., “section”, “figure”, “in this paper”, etc.) and some stopword that could not be filtered automatically (e.g., “in” is a preposition, and as such should be discarded, but also the abbreviation for the unit of length “inch” and of the state of Indiana, and thus was not removed automatically). Out of this set, we arbitrarily took the first 100 elements, ranked according to their frequency of occurrence, as indicators of the main topics discussed in the corpus. We term this set the *fingerprint* of the corpus.

Specificity analysis A similar analysis was conducted separately on each of the 10 volumes comprising the corpus, focusing this time on those terms in the fingerprint of each volume that did not appear in the fingerprint of the whole corpus. The n -grams thus identified represent specific themes that have emerged in a particular year or set of years, but were not general enough to make it into the entire corpus fingerprint. This specificity analysis allowed us to identify those topics whose importance has changed significantly in the course of the years.

Evolution traces Of particular interest in every historical review is the way in which phenomena change in the course of time. In our analysis, we wanted to study how different research themes have emerged, become established, or faded out. Usually, the establishment of new terminology indicates that a novel phenomenon has been pointed out by some researcher. If the novel theme catches the interest of others, as is often the case, the new term is found, with increasing popularity, in the following years. A term can also “fade out”, either because no progress is being made on the issue, so that after some time the theme is no longer the focus of novel research, or because a problem has been solved — and also in this case, new research will seldom refer to it.


Thus, the expected typical life cycle of an n -gram representing a research theme consists of a sudden appearance, followed by a period of relatively high interest (and similarly high frequency of occurrence), and ending in a stabilization phase where the frequency drops to a lower level. Of course, with only 10 data points (one for each volume of proceedings), we cannot expect this scheme to be followed smoothly for all terms. Indeed, as we will see in Section 3, exceptions are very common. An n -gram can never appear for a few years, and then become “fashionable” again, possibly following a breakthrough on the subject.

² This criterion was checked automatically by using the WordNet database [2].

The correspondence between n -grams and themes is not easily described in general. For example, the concept of “use case” can be referred to through the 2-grams “use/case” and “use/cases”. We have used regular expressions on n -grams (e.g., “use/cases?”) to capture such variants. Contrary to the kind of analysis we presented so far, deciding which regular expressions are best suited to capture the meaning of a concept is a subjective decision. We relied on our personal familiarity with the state of research in requirements engineering to choose which regular expressions to use.

The *evolution trace* associated with a regular expression ρ on n -grams is the vector obtained by considering, for each year, the sum of the frequencies of all the n -grams matching ρ found in the volume for that year. For example, the evolution trace for “use/cases?” (matching both “use/case” and “use/cases”) is

$$[0, 0, 223.32, 326.02, 212.85, 1015.41, 203.07, 54.28, 300.12, 122.48]$$

Notice that, to simplify the presentation, we express frequencies as number of occurrences over 100,000 words (as in 20.35) rather than as absolute values (as in 0.0002035). The same evolution trace for “use/cases?” can be represented visually by an histogram, as follows:  $\times 1015.41$ where the first mention of the term (in 1997) and the peak of frequency (in 2000) can be easily spotted. As we are most interested in relative variations, we will generally scale histograms non-uniformly; the number after the \times sign is the absolute value of the maximum, and provides an indication of the scale that is being used.

3 Results of the analysis

In this section we present the results of our analysis, together with our comments on their interpretation. The amount of data generated and collected in the course of this study is substantial (around 300 Megabytes of data), we thus limit ourselves to presenting the most striking features. It should be noted that, while the results we present are objective, our interpretation of them is not. In particular, the reader might disagree with our extrapolation of the data to future trends and with our identification of under-represented research areas. To address both these concerns, we have made larger portions of the data available on-line, making them accessible from the REFSQ web site <http://www.refsq.org>, so that interested readers can run their own analyses and draw their conclusions.

3.1 Fingerprinting results

Fingerprinting analysis of the whole corpus revealed, as was to be expected, that the main topic of REFSQ during the last 10 years has been... *requirements*. More interesting are other four top-ranking terms (see Table 3): “system”, “process”, “software” and “engineering”. We can optimistically interpret this list as confirming that in REFSQ the focus is on requirements for software embedded in a wider context (the “system”), and that the emphasis is on the process of

producing, or developing, these requirements with an engineering approach. In fact, “requirements engineering” is the most frequently occurring 2-gram, and two other related terms, “software engineering” and “engineering process” also make it in the fingerprint list.

Among the most occurring terms, we find many other typical artifacts of the software engineering process. These include general terms like “model”, “project”, “product”, “description”, “document” etc., but also more specific terms. We can classify these artifacts in those that are preliminary or are proper of the requirements engineering process, and those that are typically developed after the requirements have been identified.

In the first class we find, beside “requirements” and the more precise “software requirements” and “SRS” (for “Software Requirements Specification”), terms like “goals”, “scenarios”, and “use cases”. We take this as an indication that these three techniques: goals analysis, scenarios, and use cases, of all the methods and techniques proposed for early requirements engineering, have been the most successful in rousing the interest of the researchers that have contributed to REFSQ over the years.

In the second class terms like “design”³, “specification” and “implementation” appear. A deep interest among requirements engineering researchers and practitioners for these artifacts is not surprising, as requirements often dictate design decisions (especially when non-functional requirements are considered), and specifications and implementations are normally validated against requirements to make sure that the system being specified or implemented is actually what desired by the customer. What is rather surprising is, instead, the lack of connection with actual programming (i.e., writing computer code). In fact, terms like “program” or “code” are not found on their own in the fingerprint list (that is to say, they do not occur significantly more frequently than any other random word); and the only reference to coding to appear significantly often (but with a rather low frequency score of 1.7) is the 3-gram “design/and/code”, often used as a catch-all cliché to indicate whatever follows the requirements collection and analysis process. In our opinion, there are opportunities for interesting research to be done on the interaction between requirements and actual code, learning in part from the lesson of literate programming [6] and in part from small-scale development processes as in the Personal Software Process [4] or in Extreme Programming [9]. Apparently, these issues have not caught the attention of researchers attending REFSQ until now.

The organizational side of requirements engineering is well represented in our fingerprint. The term “user” is among the most common ones, and “business”, “management”, “knowledge”, “customer”, “strategy” and “organisation” are also well-represented. REFSQ characterizes itself as a workshop with strong links to business reality, where theoretical advancements are never too removed from their practical applications. On the other hand, social aspects have not

³ Notice that in our analysis we have not distinguished the different part-of-speech roles of each word, so terms like “design” conflate both the noun and the verb in a single n -gram.

<i>n</i> -gram	Freq.	<i>n</i> -gram	Freq.	<i>n</i> -gram	Freq.
requirements	1461.850	time	117.214	relationships	54.856
system	749.139	research	114.857	functional	53.999
process	597.640	customer	112.928	solution	53.785
model	576.211	study	105.642	document	53.785
case	449.569	techniques	99.642	selection	52.499
software	430.284	stakeholders	90.856	formal	51.428
goals	338.141	context	84.642	current	50.785
engineering	335.570	order	80.571	practice	50.571
information	317.356	information/systems	80.142	point	50.571
scenario	273.641	the/process	79.285	communication	50.142
user	261.856	structure	76.499	participants	49.928
requirements/engineering	253.070	issues	76.285	focus	49.928
use/case	240.856	provide	73.499	proceedings	49.714
problem	234.427	possible	71.571	identified	49.714
project	226.927	control	71.356	developed	49.714
development	226.499	results	71.142	object	49.285
approach	209.999	framework	70.499	srs	48.214
different	209.785	computer	70.285	process/model	48.214
quality	203.570	software/engineering	69.213	implementation	48.214
analysis	201.856	particular	68.999	identify	47.356
design	194.356	reuse	68.785	dependencies	47.356
decision	192.642	specific	68.571	aspects	47.356
method	178.499	description	68.571	terms	47.142
task	172.071	defined	68.356	proposed	47.142
the/system	156.428	understanding	68.142	criteria	47.142
example	156.214	application	67.071	role	46.928
data	156.214	existing	64.714	re/process	46.928
support	143.999	case/study	62.785	step	46.071
management	142.714	international	62.571	situation	45.856
specification	140.999	part/of	62.142	action	45.642
re	139.928	requirements/specification	61.928	organisation	45.428
business	139.499	language	61.071	university	45.214
of/requirements	139.071	elicitation	59.999	access	44.999
work	138.856	performance	59.785	describe/the	43.714
change	136.714	security	58.285	software/development	42.428
types	134.142	environment	56.999	software/requirements	38.142
domain	133.928	technology	56.571	the/re/process	37.499
activities	130.499	strategy	55.928	analysis/of	35.785
tool	125.999	interaction	55.928	business/process	34.499
product	125.999	questions	55.499	understanding/of	34.285
level	123.428	means	55.071	requirements/model	34.285
section	122.571	map	55.071	engineering/process	32.999
knowledge	120.214	use/of	54.856	the/supplier	30.428

Table 3. Results of fingerprinting analysis for the whole REFSQ corpus.

been particularly prominent in the workshop as a whole (but we will see later that these have gained more popularity in recent years). Terms like “agreement”, “conflict”, “negotiation” are not found in our fingerprint, indicating that they have not been a major lasting focus of research in the 10 years that we are considering. This too can be taken as a suggestion for further research.

No specific technological buzzword appears in our list, which we take as a positive note. The only two exceptions are “information systems” and, possibly, “object”. The latter is difficult to classify with certainty as a reference to object-oriented thinking, as “object” can well appear in a REFSQ paper with completely different meanings. However, “object model”, “domain object”, “business object”, “object-oriented software engineering” and “object oriented programming” appear in the list of significant n -grams, although not in the topmost 100 positions.

Based on our experience, at times researchers in requirements engineering have silently or explicitly assumed an information system, in a business context, as the type of system that is being developed. This is not consistent with what is typically assumed in specification-focused research. Indeed, in the specification community the most successful ideas have been dealing with control (possibly reactive) systems, and embedded systems in general. Whereas the archetypal example in requirements engineering research might be a library administration system, the equivalent in specification techniques research might be the control software for a steam boiler. It would be certainly of some interest working in the areas of requirements for control systems (e.g., how high-level goals and requirements can be traced into a formal specification), or specifications for information systems. The current state of the practice too often sees specifications written for control systems without a proper requirements analysis, or requirements for information systems that are directly fed to the programmer for implementation, without a proper specification of what is to be done.

Curiously enough, while “information systems” ranks highly, with a frequency score of 80.14, we could find only a single significant n -gram referring to the most typical implementation technology for such systems, namely databases, and that with a meagre score of 1.93. This is somewhat surprising, as the database community has long worked on modelling domains and customer needs long before a separate requirements engineering community was established. Indeed, many modelling techniques used in early requirements engineering had their roots in Entity-Relationships diagrams, devised by and for database designers rather than for requirements specialists. However, we can positively interpret this fact as the result of a clean separation of concerns: research in RE does not focus on specific implementation technologies anymore.

As REFSQ stands for *Requirements Engineering: Foundation for Software Quality*, and we have already noted that “requirements engineering” and “software” appear as significant terms, we are left with “quality” as a candidate for further analysis. Table 3 shows that “quality” appears among the most significant n -grams, with a respectable score of 203.57. Many longer significant n -grams appear with lower scores (actually, 163 of them), but only a handful

refer to actual facets of quality. Most often, quality is addressed in completely generic terms, e.g. “software quality”, “high quality”, “quality model”, “quality software”, “quality management”, “quality assurance”, “improve the quality”, etc. A few specific types of quality mentioned in the corpus are listed in Table 4. The reader should be warned, though, that the number of occurrences of these n -grams is generally low, so the presence of an n -gram may not be indicative of a widespread or lasting interest in the corresponding concept.

2-gram	Occ.	Freq.	(3,4)-gram	Occ.	Freq.
software/quality	83	17.785625	quality/of/requirements	13	2.785700
semantic/quality	44	9.428524	quality/of/conceptual/models	5	1.071423
high/quality	24	5.142831	quality/of/the/supplier	3	0.642854
selection/quality	16	3.428554	quality/of/the/software	3	0.642854
requirements/quality	14	2.999985	quality/of/the/requirements	3	0.642854
social/quality	12	2.571416	quality/of/software	3	0.642854
product/quality	11	2.357131	quality/of/the/scenarios	2	0.428569
pragmatic/quality	11	2.357131	quality/of/the/product	2	0.428569
linguistic/quality	9	1.928562	quality/of/the/knowledge	2	0.428569
system/quality	7	1.499993	quality/of/the/ideas	2	0.428569
knowledge/quality	7	1.499993	quality/of/the/documents	2	0.428569
proposed/quality	6	1.285708	quality/of/the/customer	2	0.428569
better/quality	6	1.285708	quality/of/technical/solutions	2	0.428569
higher/quality	5	1.071423	quality/of/software/products	2	0.428569
between/quality	5	1.071423	quality/of/re	2	0.428569
systems/quality	4	0.857139	quality/of/problem/analysis	2	0.428569
syntactic/quality	4	0.857139	quality/of/natural/language	2	0.428569
measuring/quality	4	0.857139	quality/of/information	2	0.428569
feasible/quality	4	0.857139	quality/of/informal/software	2	0.428569
different/quality	4	0.857139	quality/of/decision	2	0.428569
assess/quality	4	0.857139	quality/of/a/software	2	0.428569

Table 4. The most frequently occurring 2-grams ending in “quality” and (3,4)-grams beginning with “quality” in the whole REFSQ corpus.

Also, the relative paucity of references to quality does not necessarily mean that there has been no interest for the issue. Rather, it should be taken as an indication that no shared terminology exists for talking specifically of various qualities or of specific aspects of Quality. Indeed, while several taxonomies and quality models have been proposed, both in REFSQ papers and in official standards (e.g., [5]), none of these have really gained widespread acceptance and recognition — hence, periphrases are often used when discussing quality, and the different denominations do not gain sufficient critical mass to emerge as significant n -grams in our analysis. The lack of standard terminology is, of course, a hindrance to the diffusion of research results in this area, and in itself would be an interesting subject for further work.

3.2 Specificity results

Table 5 shows an excerpt of the fingerprint lists for each year, that is, n -grams that appear significantly more frequently than the other n -grams in the contents of the proceedings for that year. As could be expected, many terms that were significant for the entire corpus are also significant for several volumes: for example, “requirements” is the most frequent term in 9 out of 10 volumes (the exception being 1994, the first REFSQ, for which, however, we have incomplete data). Still, each volume shows a somewhat distinct personality, as the reader can observe in Table 5.

More interesting are the results of specificity analysis of each volume, giving an indication of the prominent features of each edition of the workshop. Table 6 shows the main differences between the fingerprint of the whole corpus and the fingerprints of each volume — that is, n -grams that were significant w.r.t. a single volume, but not so w.r.t. the whole corpus. As can be observed, in 1994 there was great emphasis on *performance* as well as on *traceability* and on *contribution structures*. These were indeed the themes of the papers we analyzed for that year⁴ and, given the small number of papers analyzed, we cannot draw any conclusion from that. In 1995 a special emphasis was placed on hypertexts, as a means of structuring requirements, but as we will see in Section 3.3, the interest on this theme was not destined to last.

1997 was the “Year of the Agent” in the special REFSQ calendar, while the theme of *procurement* continued to enjoy a marked popularity, following its 1995 exploit. In 1998, no n -gram was so prominent; the subject covered were more broad than in previous years. We interpret this fact as an indication that the workshop was beginning to mature, approaching a long-term stability of the subject matter.

Starting in 1999, a tradition was established that each edition of REFSQ should highlight a “Special Theme”, inviting with particular emphasis papers on that specific theme. The list of special themes selected in the six editions that have been held since then is presented in Table 7. In 1999, the special theme was COTS — commercial off-the-shelf systems. Our data, however, does not indicate a special interest for COTS (at least at the lexical level) in papers presented in that year: both the list of most common n -grams (from Table 5) and that of the most distinctive ones (from Table 6) lack any reference to COTS. A single paper⁵ mentioned CCOTS (for “Complex COTS”), but the term was non-standard, and did not make it into the fingerprint lists. The same situation occurred in 2000: the proposal of a special theme (method engineering and meta-models) did not seem to have a clear impact on the choice of themes in the papers. Instead, *use-cases* and *goals* occupy the scene. The 2000 edition also witnesses a comeback of hypertext (in its revamped form as *hypermedia*) and, one year too

⁴ As an example, we can cite *Modelling the Contribution Structure Underlying Requirements*, by Gotel and Finkelstein, *Requirements and Traceability* by Morris et al., or *Requirement Engineering for Software Performance* by Opdahl.

⁵ *Supporting Reuse and Flexibility in CCOTS Variation Development*, by Deifel.

1994	1995	1997	1998	1999
performance software system requirements quality traceability demands information engineering organisation development analysis data performance/demands data/system response/time software/systems requirement/specification information/systems requirement/engineering the/projected software/engineering this/paper the/average terms/of system/response	requirements process software information model system engineering quality management project hypertext decision development requirements/engineering specification support data srs method used procurement case situation user software/process is/architecture information/systems software/engineering requirements/management the/customer requirements/engineering/ process engineering/process performance/indicators key/practice	requirements system model software information process method engineering analysis quality design scenario development requirements/engineering different case agent goals problem support context modelling procurement level criteria use/cases information/systems problem/analysis contextual/information this/paper the/framework software/requirements software/engineering semantic/quality quality/requirements	requirements system scenarios software quality model engineering process level goal information case development relationships requirements/engineering use/case requirement/chunk the/knowledge natural/language the/crews cash/from the/user software/requirements the/bank system/interaction from/atm cash/from/atm the/book system/functions bank/customers the/analysis system/internal quality/model services/to requirements/base knowledge/curtain improve/services	requirements process software system project business engineering different model change information development user design management data case knowledge analysis task quality re group domain time new work requirements/engineering level variations approach goal scenarios problem support use/cases re/process
2000	2001	2002	2003	2004
requirements case system process cases use/cases goals software information engineering requirements/engineering scenarios rationale study analysis design tool model development specification option case/study requirements/specification the/pore electronic/commerce the/srs requirements/engineer software/development software/engineering class/model packaged/software description/of activity/graphs the/class describe/the the/rationale the/pore/method requirements/analysis process/map	requirements process information engineering user model map research scenario case system software design message requirements/engineering meeting development different ems subscriber problem re complexity analysis knowledge information/systems use/case the/ems/shall user/interface requirements/model re/process software/engineering process/map the/prototype design/explanation scenario/network understanding/of/the informal/meetings software/development the/ems/shall/allow	requirements process software system model engineering re information work decisions goal development analysis product requirements/engineering specification research domain quality security role policy management design control approach access support study project re/process software/engineering requirements/model soft/goal software/development requirements/diagrams security/and/privacy access/control requirements/specification elicitation/techniques	requirements system case process engineering model data software development domain security dependencies requirements/engineering user product analysis used quality approach variation information level use/case customer different variation/points security/requirements quality/model case/study product/family the/participants repm/level the/repository problem/frames number/of the/reference reference/release product/line engineering/process requirements/engineering/ process misuse/cases software/engineering	requirements system software business problem process goal customer engineering case different design model decision stakeholders analysis product solution requirements/engineering approach support re task information agreement development techniques case/study design/options the/suppliers requirement/statements use/case sd/model software/engineering business/goals

Table 5. Fingerprinting analysis results for each volume of proceedings (excerpt). *n*-grams are listed in decreasing frequency order.

late, characterizes itself for having significantly more than average references to COTS software.

The most significant feature of the 2001 edition is the overabundance of references to *meetings*. These come mostly from a single paper⁶ which focuses specifically on handling face-to-face meetings. Similarly, another paper⁷ contains lots of references to *messages* and the *EMS* (Enhanced Messaging System), but these are in the context of a case study, and are not relevant for our purposes.

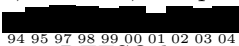
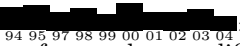
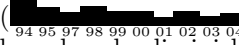
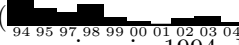
In 2002, policies and roles were on the stage, with other *n*-grams achieving only low frequency scores — again, an indication of an average mix of research subjects in that year.

In 2003, *security* was the star, and *variation* (as in “variation points”) also was mentioned frequently. Finally, the most specific term for the present year is *agreement*, a theme that for the first time receives that much attention. Interestingly, *conflict* also appears as a distinctive term for 2004, reinforcing our understanding that social issues are increasingly being studied in the RE community.

In 2001-2004, special themes have been rather general (e.g., “New ideas and on-going research”) rather than addressing specific themes (as for COTS), and we were not able to trace them to specific *n*-grams.

3.3 Evolution results

While commenting the results in the two previous sections, we have mentioned a number of *n*-grams of particular interest. Now, we want to analyze how the interest in these and other terms has changed in the course of time.

Our chief interest is, of course, in *requirements* (or *requirement*), that exhibits the following profile:  $\times 1569.58$. It is easy to see that the relevance of requirements in REFSQ has not changed substantially in the course of the last 10 years. Similarly for the pattern “engineer(ing)” (that matches both *engineer* and *engineering*) that show the substantially stable evolution profile  $\times 514.54$. The two other terms appearing in the title of the conference have a different evolution profile. The relevance of both *software* ( $\times 846.93$) and *quality* or *qualities* ( $\times 615.14$) has sharply diminished in time, from the respective maxima in 1994. In particular, we can witness the demise of quality in 1999, continuing into almost irrelevance in 2001. Even a limited comeback in 2003 does not seem to be changing the long period trend.

Our interpretation of these data is as follows. Initially, REFSQ had a distinct identity and a mission — studying how advances in requirements engineering could contribute to the improvement of software quality. But this special focus only lasted for approximately four editions. Since then, REFSQ appears to have

⁶ *Supporting Informal Meetings in Requirements Engineering*, by Braun, Bruegge and Dutoit.

⁷ *Scenario Networks: A Case Study of the Enhanced Messaging System*, by Alspaugh and Antòn.

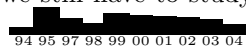
1994	Freq.	1998	Freq.	2001	Freq.
demands	445.75	refinement/relationship	103.17	message	245.83
the/projected	115.90	withdraw/cash	66.03	meeting	211.62
system/response	98.07	our/bank	66.03	ems	200.94
existing/software	89.15	reliability/testing	61.90	subscriber	188.11
the/contribution	80.24	customers/by	61.90	meetings	175.28
system/performance	80.24	crews/approach	61.90	complexity	156.05
contribution/structure	80.24	book/return	61.90	archived/message	32.06
projected/data	71.32	semantic/roles	53.65	written/examn	29.93
of/performance	71.32	the/reading	45.39	presentation/prototypes	29.93
of/hardware	71.32	data/effects	45.39	a/caller	29.93
traceability/relationships	62.41	conceptual/representation	45.39	the/scene	27.79
post-rs/traceability	62.41	and/relationship	45.39	round-trip/prototyping	27.79
performance/engineering	62.41	or/relationship	41.27	message/is	27.79
organisation/performance	62.41	normal/way	41.27	incidental/complexity	27.79
be/established	62.41	error/correction	41.27	explanation/within	27.79
to/quantify	53.49	static/verification	37.14	the/subscriber	25.65
teal/&	53.49	report/transactions	37.14	scenario/networks	25.65
pre-rs/traceability	53.49	refinement/module	37.14	in/ibistro	25.65
performance/prediction	53.49	reading/of	37.14	crisis/points	25.65
performance/measures	53.49	providing/cash	37.14		
organisation/level	53.49	linguistic/quality	37.14	2002	Freq.
in/quality	53.49	chain/of	37.14	policy	281.68
improvements/in	53.49	cash/with	37.14	roles	132.04
hardware/demands	53.49	atm/in	37.14	white-box/style	22.01
for/performance	53.49	at/level	37.14	understandability/of	20.54
demands/can	53.49	application/data	37.14	re/decisions	20.54
		analysis/model	37.14	policy/statements	20.54
				black-box/style	20.54
1995	Freq.	1999	Freq.	black-box/requirements	20.54
hypertext	281.51	variations	152.94	work/reality	19.07
procurement	174.06	requirement/set	40.99	to/soft	19.07
link/types	34.38	sap/requirements	23.65	the/understandability	19.07
isarchitecture/strategy	34.38	rm/process	23.65	the/patient	19.07
conference/on	32.23	video/parts	22.07	the/black-box	19.07
key/practices	30.08	business/object	22.07	the/alignment	19.07
structuring/informal	27.94	the/utilization	20.50	sage/publications	19.07
situational/method	27.94	the/portfolio	20.50	clustering/of	19.07
quality/system	27.94	requirements/team	20.50		
process/supporting	27.94	rapid/prototype	20.50	2003	Freq.
process/evolution	27.94	function/deployment	20.50	variation	204.25
formal/hypertext	27.94	failure/mode	20.50	satisfied/-explained	33.35
allocated/requirements	27.94	entity/types	20.50	completed/satisfied-	33.35
third/order	25.79	empirical/mean	20.50	between/variants	33.35
the/allocated	25.79	commercial/practice	20.50	the/simulator	31.26
scenario/aspects	25.79	reuse/indicators	18.92	second/workshop	31.26
project/goal	25.79	quality/function	18.92	nfrs/are	31.26
process/execution	25.79	problem/situation	18.92	user/documentation	29.18
process/area	25.79	portfolio/analysis	18.92	the/operator	29.18
informal/representations	25.79	of/reuse	18.92	points/and	29.18
informal/information	25.79	of/defects	18.92	operational/data	29.18
hypertext/nodes	25.79	logical/file	18.92	nfrs/in	29.18
case/frame	25.79	a/ccots	18.92	first/workshop	29.18
s3/model	23.64			threats/and	27.09
project/scenario	23.64	2000	Freq.	the/variant	27.09
production/process	23.64	rational	234.50	root/causes	27.09
process/monitoring	23.64	option	177.58	our/framework	27.09
hypertext/editor	23.64	withdraw/money	36.43	data/warehouses	27.09
		new/equipment	36.43	between/variation	27.09
1997	Freq.	computer/centre	36.43	are/elicited	27.09
agent	406.62	yes/no	31.87	explained/total	25.01
procurement	176.97	missing/use	31.87	derived/dependencies	25.01
systematically/explore	33.71	of/rationale	29.60	context/diagram	25.01
and/conceptions	33.71	goal/classes	29.60	cause/analysis	25.01
explore/all	31.60	distributed/prioritization	29.60	and/variation	25.01
supplier/selection	29.50	using/hypermedia	27.32		
representing/contextual	29.50	use-case/diagram	27.32	2004	Freq.
properties/and	29.50	the/use-case	27.32	agreement	135.92
karat/method	29.50	secondary/scenarios	27.32	goal/strategy	23.90
classification/scheme	27.39	pore/process	27.32	vertical/conflicts	22.40
requirements/texts	27.39	equipment/request	27.32	feature/use	22.40
agent/is	25.28	use-case/descriptions	25.04	customer/had	22.40
the/contextual	25.28	the/gui	25.04	the/gondola	19.42
procurement/processes	25.28	nonfunctional/constraints	25.04	domains/in	19.42
bww/model	25.28	check/amount	25.04	conflicts/of	19.42
are/distributed	23.18	an/option	25.04	usage/data	17.92
systems/design	23.18	user/goals	22.77	runway/atco	17.92
sub/processes	23.18	the/electronic	22.77	product/life	17.92
scenario/context	23.18	report/errors	22.77	of/groupware	17.92
not/supported	23.18	rationale/maintainer	22.77	defining/occurrence	17.92
distributed/on	23.18	move/equipment	22.77	decision/-making	17.92
agents/and	21.07	goal/set	22.77	decision/-makers	17.92
system/procurement	21.07	goal/analysis	22.77	decision/-maker	17.92
scenario-based/approaches	21.07	cots/software	22.77	a/preference	17.92
requirements/taxonomy	21.07	commerce/application	22.77		
icon/dictionary	21.07				
contractual/requirements	21.07				
communication/model	21.07				
agents/in	21.07				


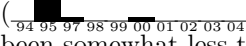
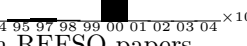
Table 6. Results of specificity analysis (excerpt).

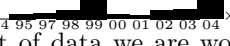
1999	Addressing RE for Commercial Off-The-Shelf (COTS) Systems
2000	Consolidating the field — Method Engineering and Meta-Modelling Approaches to Requirements Engineering
2001	Understanding and Improving RE Processes for Higher Software Quality
2002	New ideas and ground-breaking on-going research: RE innovation at the service of Software Quality
2003	Integration of Requirements Engineering into Software Engineering
2004	Quality Requirements as Business Advantage

Table 7. Special themes in recent editions of the workshop.

lost any special relationship with *quality*, and has become a venue where general requirements engineering research is discussed. This is indeed consistent with how many participants see REFSQ: as a full-fledged RE conference, paired with a very effective workshop structure.

Of the four more significant *n*-grams in the whole REFSQ corpus fingerprint, we still have to study *process*. The evolution diagram of *process* or *processes* is  $\times 936.93$. It is interesting to notice that, after a maximum in 1995, the interest in process issues gradually declined, to resume sharply in 1999 — the same year in which quality issues had a substantial drop. Since then, processes of various kind have been discussed with a continued, if slowly decreasing, interest in REFSQ papers.

Certain terms have had a sudden appearance as well as demise. For example, studying *procurement* ( $\times 176.97$) was very fashionable in 1995-1997, but researchers quickly abandoned the theme. Similarly, hypertext ( $\times 281.51$) and hypermedia ( $\times 104.73$) have been somewhat less than permanent presences in REFSQ papers.

The financially-impaired reader can find the evolution of *business* (or *businesses*) amusing, especially remembering the burst of the technology stock market bubble in 2000:  $\times 318.49$.

Given the amount of data we are working on, and due to space considerations, we cannot provide an exhaustive commentary of all interesting features that can be found while browsing the results of our evolution analysis. Table 8 shows the compared evolution of some families of related terms: the reader can draw her own conclusions about how instant fashions and long-term trends have contributed to shape the research output of REFSQ in the last 10 years.

4 Conclusions

When trying to understand the trends and motivation behind scientific research, nothing can substitute a well-reasoned assessment by a knowledgeable person familiar with the field. Still, a lexical analysis of the kind we have reported has

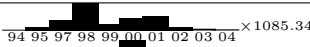
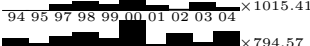
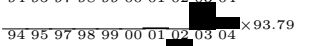
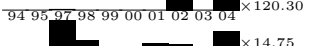
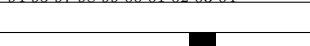
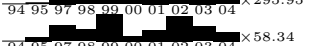
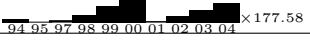
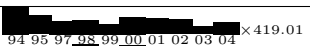
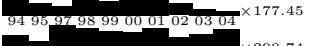
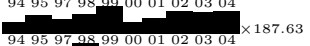
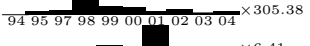
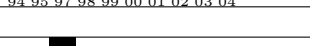
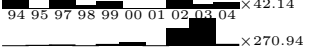

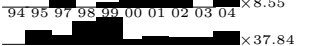
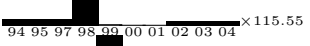
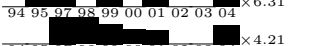
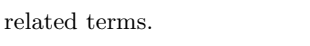





Specific RE techniques	
scenarios?	 × 1085.34
use/cases?	 × 1015.41
goals?	 × 794.57
problem/frames?	 × 93.79
i*	 × 120.30
KAOS	 × 14.75
RE activities	
elicit(ed ing ation)	 × 295.95
negotiat(e ed ing ion)	 × 58.34
priorit(y i(z s)(e ed ing ation))	 × 177.58
Other development phases	
specif(y ied ying ication)	 × 419.01
verif(y ied ying ication) validat(e ed ing ion)	 × 177.45
design(ed ing)	 × 290.74
implement(ed ing ation) cod(e ed ing) program(med ming)	 × 187.63
test(ed ing)	 × 305.38
retir(e ed ing ment)	 × 6.41
Non-functional requirements	
non(- /)functional/requirements	 × 42.14
secur(ity e)	 × 270.94
safe(ty ness)	 × 35.66
robust(ness)	 × 8.55
usability user(- friendl(y iness))	 × 37.84
reliab(ility le)	 × 115.55
interoperab(ility le)	 × 6.31
scalab(ility le)	 × 4.21

Table 8. Compared evolution for families of related terms.

its advantages. It is objective and repeatable, and is not influenced by personal preferences. Also, it is repeatable, year after year on a growing corpus, or with different corpora, for example to compare research presented at REFSQ with that presented at other conferences. This blind reliance on the lexicon can be considered a threat to the validity of our results, but the threat is mitigated by the fact that authors of scientific papers normally do their best to use consolidated terminology in a uniform manner: hence, the relationship between lexicon and semantics is in this kind of writing more close than in generic text.

We have answered, in a certain detail, to the two questions that we asked at the beginning: what are the frequencies of different research topics, and how have research topic frequencies changed over time. While we could not comment all the results, we have made the raw data on which this study is based available to interested readers on the web, so that they can browse the collection and mine the lexical content according to criteria that we may have missed.

From the results of our study, it appears that REFSQ has moved during the last 10 years from a specialized workshop, with a focus on quality models for requirements engineering, to a general RE conference with a workshop discussion model. REFSQ is not detached from the more ample RE community, and has been traversed at times by sudden peaks of interests for emerging (or re-emerging) topics. The core issues, however, have enjoyed a lasting popularity among the researchers contributing to REFSQ.

Our analysis has also brought to light a number of “sweet themes” that seem promising but that have not yet been subject to rigorous investigation. As examples, we can cite the relationships between requirements and actual coding; or what happens to the requirements of a software system that is being retired (is there any way to tell *when* a system should be retired, based on whether its original requirements are satisfied in a different manner in a changed environment?). Also, non-functional requirements continue to receive much less attention than “traditional” (functional) requirements.

The reader may wish to compare our findings with those in [8], in this same volume. The two studies were conducted in a completely independent way, and have used different research methods: objective lexical analysis in this paper, subjective literature survey in [8]. The fact that the results of the two studies have a consistent overlap reinforces, in our opinion, the value of both.

We hope that, in addition to amusing the reader, the results of our study will contribute to a deeper understanding of the long-term evolution of our discipline, as well as pointing out some stimulating new theme worth investigating in future research.

References

1. E. Dubois and K. Pohl. RE'02: A major step toward a mature requirements engineering community. *IEEE Software*, 20(1):14–15, 2003.
2. C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
3. Ghostscript, Ghostview and GSview. <http://www.cs.wisc.edu/~ghost/>.

4. W. S. Humphrey. *Introduction to the Personal Software Process*. Addison-Wesley, 1997.
5. International Organization for Standardization. *ISO 9126: Information Technology — Software Product Evaluation — Quality characteristics and guidelines for their use*. 1991.
6. D. E. Knuth. *Literate Programming*. Number 27 in Lecture Notes. CSLI, 1992.
7. C. G. Nevill-Manning, T. Reed, and I. H. Witten. Extracting text from PostScript. *Software: Practice and Experience*, 28(5):481–491, 1998.
8. A. L. Opdahl, E. Dubois, and K. Pohl. Ten years of requirements engineering: Foundations of software quality — outcomes and outlooks. In *Proceedings of REFSQ'04*, 2004.
9. G. Succi and M. Marchesi, editors. *Extreme Programming Examined*. Addison-Wesley, 2000.
10. The virtual paper project home page. <http://www.research.compaq.com/SRC/virtualpaper>.