

# REVERE: support for requirements synthesis from documents

Pete Sawyer, Paul Rayson and Roger Garside  
Computing Department  
Lancaster University  
Lancaster  
UK. LA1 4YR  
Tel: +44 1524 593780  
Fax: +44 1524 593608  
{sawyer, paul, rgg}@comp.lancs.ac.uk

## Abstract

Documents are important sources of system requirements. This is particularly true of domains that are document-centric in terms of their operational and development processes. For system evolution in organisations that have been subject to organisational change and loss of organisational memory, documents may be *the* major source of key requirements. Hence, systems engineers often face a daunting task of synthesising crucial requirements from a range of documents that include standards, interview transcripts and legacy specifications. The goal of REVERE was to investigate support for this task which has been described as *document archaeology* (Robertson and Robertson, 1999). This paper describes the resulting REVERE toolset, its utility for document archaeology and for other tasks that have emerged in the course of our experiments with the toolset.

## Keywords

Requirements engineering, systems engineering, document archaeology, natural language processing

## Introduction

Systems engineers are routinely faced with the need to extract information from documents; either as a direct source of requirements or as part of the process of gaining domain knowledge needed to interpret requirements elicited from human stakeholders. Documentation associated with systems and business processes forms a crucial source of requirements. Organisations that have been subject to organisational change frequently suffer a leakage of organisational memory as people with key knowledge leave or retire. Here, the documentation associated with systems and business processes forms a crucial source of requirements with the result that documents form the primary source of legacy requirements. Failure to recover legacy requirements risks wasting resources by having to rediscover them. Worse, key requirements may be overlooked by, for example, failing to retain support for key but unrecognised requirements.

However, synthesising requirements from documentary sources is a difficult task that requires skill, domain knowledge and is almost completely unsupported by tools. This paper describes the results of the REVERE<sup>†</sup> project which was conceived to help address problems with evolving legacy systems and business processes in organisations subject to change. REVERE resulted in a toolset that uses statistical natural language processing (NLP) techniques for assisting the synthesis of requirements from documentary sources.

REVERE was a collaboration between Lancaster University and Adelard. Adelard is a small but internationally influential safety and dependability consultancy that contributed in many valuable ways to the project, principally: expertise, technical resources, and case studies. They were particularly apposite for REVERE because:

- Systems development processes in the safety and dependability domains are typically heavily regulated and document-heavy (for example, safety cases and standards). Adelard thus represented a sector of the software industry where crucial requirements are always embedded in documents.
- Staff at Adelard are closely involved in UK and international standards definition and so were able to provide us with access to draft standards, expert knowledge of the role of standards in dependable systems, and an insight into the standards development process. This was useful because standards often provide important sources of requirements.
- During the course of the project, Adelard was engaged in a number of industrial projects that faced many of the challenges described in the introduction.

---

<sup>†</sup> REVerse Engineering of REquirements. EPSRC Systems Engineering for Business Process Change (SEBPC) programme project number GR/MO4846.

## **Related work**

Systems engineering is mainly concerned with identification and analysis of the system requirements, identification of a configuration of components (the system architecture) that will satisfy the requirements and verification that once completed and integrated, the configuration of components does meet the requirements. To do this, the system requirements must be acquired and analysed. These requirements are always constrained by many factors that include budgetary limits, the operational environment, technical feasibility, the need for standards compliance and many others. Hence, many factors must be understood and balanced by a systems engineer. Natural language invariably plays a large part in the systems engineering process and this fact has attracted sporadic interest in the employment of NLP techniques in systems engineering. Work has focused on both the products of the process and inputs to the process.

### *Natural language products of the systems engineering process*

Products of the process include specifications of the requirements and test plans. The use of natural language is needed to enable these documents to be read by a heterogeneous readership that includes both developers and customers. However, it is hard to describe complex concepts simply, clearly and concisely in natural language. In recognition of this, several research projects (Steuten, van de Reit, and Dietz 1999, Ambriola and Gervasi 1999, Rolland and Proix 1992, Cyre and Thakar 1997) have investigated the synthesis of system models (object models, data-flows, etc) from natural language descriptions. Another body of work, for example Rosenberg, Hammer and Huffman (1998) have investigated the generation of quality metrics from specification and other natural language, but highly structured documents.

### *Natural language inputs to the systems engineering process*

The inputs to the process include human-sourced information (such as interview transcripts) and structured documents. These might include standards, process descriptions, user manuals, specifications of other systems, etc. A variety of different documents and document types are often needed to complement human sourced information to identify the system requirements and the constraints under which the system must operate. These requirements and constraints are never pre-formed but have to be teased out of a mass of information that may be poorly structured, contradictory, at varying levels of detail and of uncertain relevance.

The systems engineer must use whatever information resources are available to synthesise the requirements (Butler, Esposito, and Hebron 1999). This typically entails an iterative process of inferring key abstractions (stakeholders, roles, tasks, domain objects, etc.) and verifying these against the operational and organisational environments that form the context for the projected system. Hence while work concerned with products of the systems engineering process relies upon a pre-existing formulation of the system requirements, work on inputs to the process must cope with much messier natural language text.

Dan Berry and colleagues have studied this problem over a number of years (Berry, Yavne, and Yavne 1987, Aguilera and Berry 1990, Goldin and Berry 1997). Their work is based on the use of pattern matching techniques to extract abstractions. The frequency with which the abstractions occur within the text is taken as an indication of the abstractions' relevance. The authors of the work recognise this assumption has been challenged by work on automatic abstraction in other domains, but argue that it appears to be valid in the context of requirements analysis.

Both Berry's work, and similarly motivated work by (Fliedl et al 1999), explicitly recognise that NLP cannot automate the system engineer's job. The system engineer still has to read the myriad documents. Instead, the tools seek to mitigate the problems of information overload. They do this by compensating for human weakness (such as attention lapses due to tiredness) by helping to flag abstractions that would repay detailed manual analysis. This pragmatic view of the potential for NLP is informed by a real understanding of the problems of system engineering and formed the starting point for REVERE. However, we were anxious to avoid the scalability problems suffered by the above researchers' rule-based NLP tools and the constraints this imposes on the diversity of document types, structure and language usage.

## **The REVERE toolset**

In contrast to rule-based NLP, *probabilistic* NLP techniques offered clear potential but had not been tested in a systems engineering domain. Instead of attempting to model the grammar of a natural language as a set of rules, probabilistic techniques classify words on the statistical likelihood of them being a member of a particular syntactic or semantic category in a particular context. The probabilities are derived from large corpora of free text which have already been analysed and 'tagged' with each word's syntactic or semantic category. Extremely large corpora have been compiled (the British National Corpus consists of approximately 100 million words (Aston and Burnard 1998)). For some levels of analysis, notably part-of-speech tagging, probabilistic NLP tools have been able to achieve very high levels of accuracy and robustness unconstrained by the richness of language used or the volume of documentation.

Probabilistic techniques extract interesting properties of the text that a human user can combine and use to infer meaning. Evidence from other domains suggests that such tools can effectively support analysis of large documentary sources. For example, in (Thomas and Wilson 1996) probabilistic NLP tools were used to confirm the results of a painstaking manual discourse analysis of doctor-patient interaction. In this application, they were also able to reveal information that had not been discovered manually.

The execution time of the tagging process varies approximately linearly with the document size. Once the text has been tagged, retrieval and display tools are needed to allow the user to interact with the document. These use the tags to provide views on the document that reveal interesting properties and suppress the bulk of text. They do this in a way that is largely independent of the size of the document. Hence, the user is protected from information overload by being selective about the information they want to extract.

We have adapted and experimented with a set of existing NLP tools developed at Lancaster for the processing of English language text. The most important of these is CLAWS (Garside and Smith 1997). This uses a statistical hidden Markov model technique and a rule-based component to identify the parts-of-speech (POS) of words to an accuracy of 97-98%. One obvious application of this in a system engineering context is the identification of modal verbs such as 'shall', 'must', 'will', 'should', etc. Expressions of need, desire, etc., consistent with user or system requirements can therefore be located in a document very easily and without the need to construct complex regular expressions or search templates.

A semantic analyser (Rayson and Wilson 1996) uses the POS-tagged text to assign semantic tags that represent the general semantic field of words from a lexicon of single words and an idiom list of multi-word combinations (e.g. 'as a rule'). These resources contain approximately 52000 words or idioms and classifies them according to a hierarchy of semantic classes. For example, the semantic field tag *A1.5.1* represents words or idioms meaning *Using* which is a subclass of *general and abstract terms*. Words that would be assigned this tag (in the appropriate POS context) include *user*, *end-user* and *operator*. Similarly, the tag *X2.4* is a subclass of *Psychological actions, states and processes* and would be assigned to terms meaning *Investigate*, such as *search*, *browse* and *look for*.

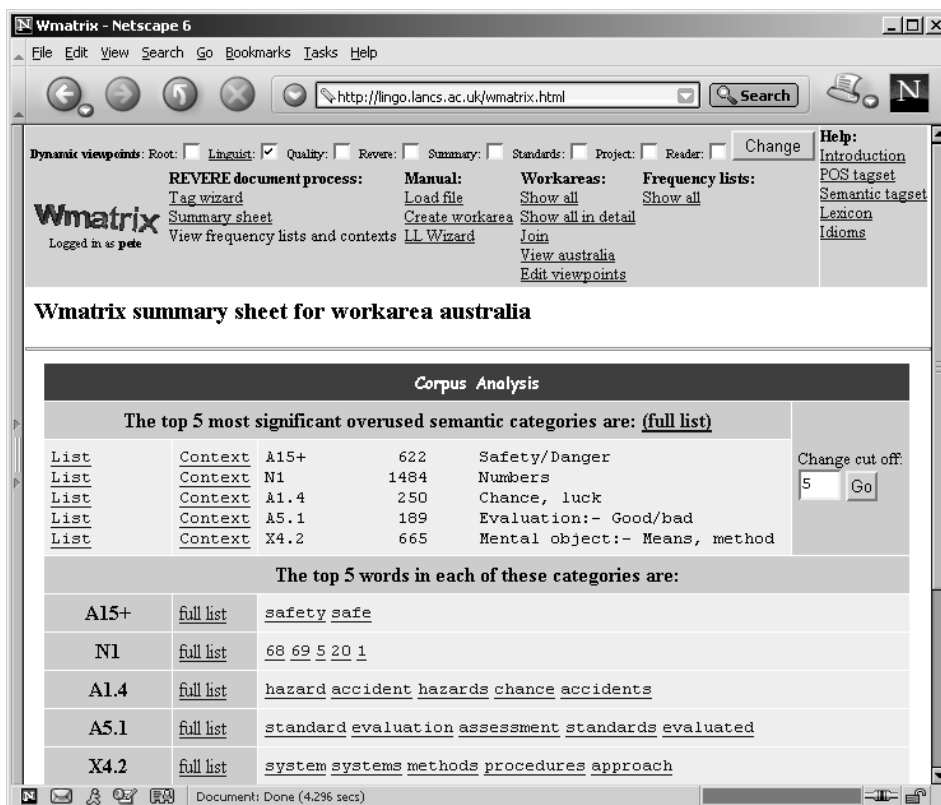


Figure 1 Wmatrix

These tools are integrated into a tool framework called WMATRIX, an evolution of an earlier tool called XMATRIX (Rayson, Emmet, Garside and Sawyer 2000) (figure 1). This embodies a process model that leads the user through a sequence of steps needed to apply the POS and semantic tagging and other types of analysis that, once done, allow the user to interact with abstractions of the text. Many of these abstractions are provided by frequency profiling. At the most basic, this produces a simple concordance of individual words and displays them in the context of their surrounding text. Frequency profiling becomes more useful when a semantically

tagged document can be compared against a normative corpus. Comparison with the normative corpus allows information to be extracted from a document by searching for statistically significant deviations from the frequency norm suggested by the corpus. This exploits the tendency for words or expressions to have different semantic profiles in different domain contexts. A general usage normative corpus is likely to reveal many of the dominant domain entities and roles as words or expressions that occur with a frequency that deviates grossly from the norm. These kinds of abstractions can help build up a picture of what the system must do and of the people and objects in the system's environment.

With complex systems, some separation of concerns needs to be imposed on the problem space in order to understand the requirements and roles of different stakeholders. This is supported by the use of viewpoints where a viewpoint is a *partial specification* (Jackson and Jackson 1996); a subset of the system requirements that represent those requirements unique to a particular stakeholder's perspective of what the system must do. Again, we were able to leverage previous research at Lancaster where we had developed viewpoint models for both requirements elicitation and process analysis (Sommerville, Sawyer and Viller 1998, and 1999). These informed the design of a function that supports the identification and definition of roles that correspond to stakeholder viewpoints. By finding the set of candidate roles and viewing where they occur in the body of the text, it is possible for the systems engineer to verify whether the roles are important and build up an understanding of how they interact with the system's environment.

To illustrate how the REVERE toolset can be used for document analysis, the following sections describe two examples. These are an analysis of the requirements for an air traffic control system and on an evaluation of a new standard for the development of safety-critical systems.

### **An air traffic control application**

The target documents were field reports of a series of ethnographic studies at an air traffic control (ATC) centre. This formed part of a study of ATC as an example of a system that supports collaborative user tasks (Bentley et al 1992). The documents consist of both the verbatim transcripts of the ethnographer's observations and interviews with controllers, and of reports compiled by the ethnographer for later analysis by a multi-disciplinary team of social scientists and systems engineers. The field reports form an interesting study because they exhibit many characteristics typical of documents from which requirements engineers have to synthesise requirements. The volume of the information is fairly high (103 pages) and the documents are not structured in a way (say around business processes or system architecture) designed to help the extraction of requirements.

Two stages in the analysis are shown: a comparison of the text to discover candidate roles; and an analysis against a normative corpus.

#### *Role analysis*

Role analysis in the example was performed by a combination of POS analysis and regular expressions. This is tailorable, but the one we have evolved in the course of our experiments looks for human agent nouns with common endings for job-titles (such as 'er' or 'or', 'et' or 'ot', 'man' or 'men', etc) and adjectives that are commonly used without their accompanying noun ('head', 'chief', 'sub', etc.). Using this, the candidate roles that occur most frequently in the ATC document (and hence imply significance) are shown in figure 2.

controller	NN1	317
chief	NN1	199
sector	NN1	88
controllers	NN2	71
pilot	NN1	61
sectors	NN2	41
computer	NN1	35
manchester	NP1	35
transfer	NN1	19
wingmen	NN2	18
man	NN1	15
roger	NP1	13
coordinator	NN1	13
number	NN1	11
dover	NP1	10
ethnographer	NP1	10
paper	NN1	10

Figure 2 Candidate roles in air traffic control

Figure 2 shows a mixture of words that are clearly roles (controller, chief, wingman, coordinator and ethnographer) along with some noise (e.g. computer, manchester, roger). Ethnographers are of course roles in the analysis rather than the application domain, but the other three are all roles or stakeholders, with their own

requirements or viewpoint on ATC. Theories about whether the candidate roles are significant or not can be tested using the tool by viewing the context of their occurrences in the document. Figures 3 and 4 show examples of the contexts in which occurrences of ‘controller’ and ‘chief’ occur.

Context line	
ch sector is worked by one radar eral sectorscan be worked by one orcan be worked by more than one a self evidently quiettime . The e States .   I watched as the tion of flight . &quot;   The n   Ian Sommerville   9:05 Nor/Southseparation .   9:35 sation with an adjacent military radar controllerto the military ual strips .   12:55 Incoming four onthis sector )   7:40 : ws attention to it by pointing . ontrolled airspace . &quot;   ne output is not to be trusted . you 're wrong . &quot;   9:15 trol Systems for HCI Design   opposite direction at 140 .   e types of aircraft .   11:20 k .   l   12:02 The work a	controller , with a chief who is responsible controller , and at busy times they can be s controller , for instance by dividing the se controller was very interested in what I was controller began to write ' l260L'i n red on controller marked a horizontal line near the Controller began to talk again . Explained t Controller marks a right hand turn on a stri controller , andpointing to information on s controller , - &quot; so he does n't shit hi controller stands behind . Situation of plan Controller is telephoned by another sector t Controller says ' chiefs decision .... ' , c Controller writes frequency into scribble bo Controller points tostrip indicating that a Controller puts a red box around the flight Controller moves one strip under another , b Controller says &quot; continue descent ; fl Controller hand-over . The situation appears controller does is described as &quot; makin

Figure 3 References to the user role *controller*

Context line	
ed by one radar controller , with a down here the radar &quot;   The e traffic is Eastbound .   10:54 's described as a quiet time .   o 31 ... &quot;   : v   11:52 ips in a rack .   11.43 Incoming ler says ' chiefs decision .... ' , three minutesearly ... &quot;   ubsequent general discussion with a on the pink strips mean .   5:17 passed by telephone .   5:40 The flight for some reason .   5:55 > 6:30 Assistant ton the ' phone to   &quot; Eastem with a joiner , &quot; passes strip to chiefJ   eturns it and nods .   &quot; OK nods .   &quot; OK Chief tTo the quot; anything above 35 &quot;   anyway , will he ? &quot;   7:35 t;   iWrites 350J on stript  	chief who is responsible for co- ordinatio chief also took an interest in the convers Chief describes his writing of information Chief sets &quot; targets &quot; which he Chief puts a handwritten , pink strip into Chief and controllers . Chiefs discuss cur Chief nods , andplaces strip fully in posi Chief adds &quot; If you assume all the in Chief , who opined that electronic flight Chief says to assistant :   &quot; I 'm Chief explains to me that strips will some Chief mentions Mode S transponders and sug Chief :   &quot; Eastem with a joiner , Chief &quot; passes strip to chiefJ   C Chief returns it and nods .   &quot; OK Chief tTo the chief on the phone from anot Chief on the phone from another sector he Chief : &quot; except 37 &quot;   Pole Chief points out me that use of strips in Chief : &quot; OK ... what are you going t

Figure 4 References to the user role *chief*

By browsing the roles, the systems engineer can impose a viewpoint on the mass of information that allows them to build up a picture of the corresponding stakeholders’ activities within the system’s application domain. The first lines in each of figure 3 and 4, for example, (which show the same sentence) include information about both roles’ responsibilities. Of course, there is also some noise and sometimes synonyms are used for a single role. However, the technique allows candidate roles to be quickly identified and verified, and information that can be used to inform the derivation of roles' requirements is isolated.

### Corpus analysis

The motivation for corpus analysis is that entities that are significant to the application domain will be revealed by the relative frequency of their appearance in the text when compared against a normative corpus. The normative corpus used by the REVERE toolset is a 2.3 million-word subset of the BNC derived from the transcripts of spoken English. Using this corpus, the most over-represented semantic categories in the ATC field reports were (the LH column is the log-likelihood figure - a measure of deviation from the word’s frequency deviation from the normative corpus, see Rayson and Garside, 2000):

Log-likelihood	Tag identifier	Semantic category (examples)
3366.66	S7.1	<i>power, organising</i> ('controller', 'chief')
2578.80	M5	<i>flying</i> ('plane', 'flight', 'airport')
988.09	O2	<i>general objects</i> ('strip', 'holder', 'rack')
643.54	O3	<i>electrical equipment</i> ('radar', 'blip')
535.97	Y1	<i>science and technology</i> ('PH')
449.34	W3	<i>geographical terms</i> ('Pole Hill', 'Dish Sea')
432.33	Q1.2	<i>paper documents and writing</i> ('writing', 'written', 'notes')
372.80	N3.7	<i>measurement</i> ('length', 'height', 'distance', 'levels', '1000ft')
318.82	L1	<i>life and living things</i> ('live')
310.32	A10	<i>indicating actions</i> ('pointing', 'indicating', 'display')
306.90	X4.2	<i>mental objects</i> ('systems', 'approach', 'mode', 'tactical', 'procedure')
290.06	A4.1	<i>kinds, groups</i> ('sector', 'sectors')

With the exception of Y1 (an anomaly caused by an interviewee's initials being mistaken for the PH unit of acidity), all of these semantic categories include important objects, roles, functions, etc. in the ATC domain. The frequency with which some of these occur, such as M5 (flying), are unsurprising. Others are more revealing about the domain of ATC. Figure 5 shows some of the occurrences the semantic category O2 (general objects) being browsed by a user of WMATRIX. The important information revealed here is the importance of 'strips' (formally, 'flight strips'). These are small pieces of paper with printed flight details that are the most fundamental artefact used by the air traffic controller to manage their air space. Examination of other words in this category also reveals that flight strips are held in 'racks' to organise them according to (for example), aircraft time-of-arrival.

#### Context line

<p>land and the Idle of Man ; and to write ' 1260L' in red on a he Isle of Man ... &amp;quot; This cated by the beacon printed in on printed in box ' B ' of the arrival time over that beacon ( box viously only approximate- some al line near the call sign on a med much busier . There were 16 were 16 strips in one of his sy , that talking and using an hat talking and using an input : &amp;quot; the nice thing about r marks a right hand turn on a strip with a ' 1 ' &lt;BR&gt; 10:00 organised into yellow and blue sed into yellow and blue strip holders his writing of information on w Con &lt;BR&gt; , oa &lt;BR&gt; . &lt;BR&gt; 17 oa &lt;BR&gt; . &lt;BR&gt; 17 strips in a en it 's quiet ... ' Describes es strips as a &amp;quot; planning chief puts a handwritten , pink and written , pink strip into a military aircraft . &lt;BR&gt; , 119</p>	<p>Pole strip strip box strip box strips strip strips racks input device strips strip Strips holders strips strips rack strips strip rack strips</p>	<p>Hill , which deals with the air , whilst at the same time instru was towards ' the bottom of one ' B ' of the strip ( second left ( second left ) Strips seemed br ' A ' ) This was obviously only were out of position , and I got to indicate an unusual speed . &lt; in one of his racks . &lt;BR&gt; A ; . &lt;BR&gt; A ; ' &lt;BR&gt; c&amp;lt;Tide &amp;gt device might also be , but that might also be , but that the pr is their flexibility . &amp;quot; a with a ' 1 ' &lt;BR&gt; 10:00 Strips are organised into yellow and b holders , the former to indicat , the former to indicate Westbou as &amp;quot; laborious &amp;quot; He 5 in a rack , and it 's described , and it 's described as a quie as a &amp;quot; planning mechanism . &amp;quot; &lt;BR&gt; 11:05 There is a into a rack , in what he consid , in what he considers to be as in a rack . &lt;BR&gt; 11.43 Incoming</p>
---	--	---

Figure 5. Browsing the semantic category M5

Similarly, browsing the context for Q1.2 (paper documents and writing) reveals that controllers annotate flight strips to record deviations from flight plans, and L1 (life, living things) reveals that some strips are 'live', that is, they refer to aircraft currently traversing the controller's sector. Notice also that the semantic categories' deviation from the normative corpus can also be expected to reveal roles. In this example, the frequency of S7.1 (power, organising) confirms the importance of 'controllers' and 'chiefs' as roles in the ATC domain, that were also identified by role analysis.

Using the REVERE tools does not automate the task of identifying abstractions. Much less does it produce fully formed requirements that can be pasted into a specification document. Instead, it helps the systems engineer isolate potentially significant domain abstractions that require closer analysis. It cannot guarantee completeness. For example, some important abstractions may be overlooked because their frequency of occurrence in the document being analysed is close to that of the normative corpus. In addition, word semantics

may be domain-specific leading to them being tagged wrongly (in the domain context) and hence, making it easy to overlook their significance. Nevertheless, the REVERE toolset does help the systems engineer rapidly build up an overview of the important concepts contained in a document by helping them filter the information to focus on (for example) user roles, while suppressing the effects of synonyms and inconsistent terms, and the sheer volume of documentation.

### **A standards document analysis**

This example is based upon the analysis of a new national standard for the procurement of safety-critical military systems (approximately 21000 words). Systems engineering is often constrained by regulations, standards and best operating practice, all of which may be documented in a number of sources. Safety-critical systems engineering processes tend to be particularly tightly regulated and document-centric. Systems engineers working in safety critical domains therefore need to acquire a good awareness of the standards landscape and how individual standards apply to different projects. This obviously requires a lot of reading and interpretation of standards documents. This might be to:

- keep abreast of emerging standards in their domain in order to monitor international best practice;
- to anticipate the effect of new standards on future international, national and sector (such as defence) standards;
- to build competence for possible future work in the market for which the standard was written;
- to identify a set of key attributes to assess against the standard to establish compliance.

Systems engineering standards are like *meta* requirements documents in that they specify generic requirements on the development processes and their products within a domain. In contrast to the class of documents used in the ATC experiment, standards tend to be strongly structured and highly stylised in the lexical and syntactic conventions used, and in the semantics attached to certain terms. In particular, modal verbs are frequently used to signify where system properties or development practices are mandatory or advisory. Our analysis of the standard had two goals: to determine the weight given to different development practices mandated or advised by the standard; and to identify the roles of people who would use or be affected by the standard. Here, we focus on the POS analysis because it reveals both the potential utility of the approach and the need to carefully interpret the results obtained.

WMATRIX allows the engineer to isolate words that are assigned any given POS tag. In standards documents, words given the modal verb tag ('VM') are of particular interest. Figure 6 illustrates the frequency profile of all the standard's modal verbs.

shall	VM	199	<a href="#">Context</a>
must	VM	127	<a href="#">Context</a>
may	VM	76	<a href="#">Context</a>
can	VM	60	<a href="#">Context</a>
will	VM	48	<a href="#">Context</a>
should	VM	39	<a href="#">Context</a>
could	VM	13	<a href="#">Context</a>
might	VM	1	<a href="#">Context</a>
would	VM	1	<a href="#">Context</a>

Figure 6 Modal verbs' occurrence in the standard

The most common modal verb in the standard is 'shall'. In standards (and other types of specification documents) a convention is often adopted in which 'shall' is used as a keyword in clauses that denote mandatory requirements. The convention often extends to using other modal verbs to denote weaker obligations. Any of the modal verbs may also appear in normal English usage in informal, descriptive sections of the document.

Once identified, the modal verbs were browsed in their context within the standard to build up a picture of the conventions used in the standard and to see if their usage complied with our expectations. We started by browsing the occurrences of 'shall', to distil a view of the mandatory requirements of the standard. Figure 7 illustrates this by showing a view of some of the 199 occurrences of 'shall' in the context of their surrounding text.

ety Testing 17.4.2.1 Safety Testing	shall	be carried out for all Components ..
. Corrective action and re-testing	shall	be carried out .
t case .	shall	17.4.2.4 A safe
17.4.2.5 The Developer	shall	define the safety tests to be conduc
t Plan .	shall	provide a reasoned justification of
17.4.2.6 The Developer	shall	also include traceability of tests t
, provide assurance of safety . It	shall	also provide the results of all safe
ements .	shall	be conducted under Configuration Man
17.4.2.7 The Developer	shall	also be validated to ensure that the
run , all aspects of safety testing	shall	be the maximum of the CSR levels ass
mulators used during safety testing	shall	incorporate defensive programming te
es . In this case the level applied	shall	be supported by static analysis tool
.1 The development of software code	shall	be included in the Implementation Ve
used .	shall	provide evidence that any potential
17.5.3.2.2 Flow analysis	shall	demonstrate that the software is exe
the results of applying these tools	shall	attempt to demonstrate the following
on Verification Report . The report	shall	include :
y Testing 17.5.3.3.1 Software tests	shall	17.5.3.3.4 Standard Unit T
5679 63 17.5.3.3.2 Software testing	shall	be set to minimum and maximum values
eria for test coverage of unit test	shall	be executed at least once with true a
only )	shall	be executed , including zero and non
&middledot; all input variables	shall	be executed ; and &middledot; all varia
e value ; and &middledot; all booleans	shall	be set to each possible value .
ot; all statements and all branches	shall	
feasible combinations of predicates	shall	
t; all variables of enumerated type	shall	

Figure 7 Occurrences of 'shall' in the standard

As expected, most of the occurrences of 'shall' in the standard occur in formal clauses representing mandatory requirements. However, documents cannot always be relied upon to use modal verbs consistently. This is illustrated in figure 8 which shows a subset of the 39 occurrences of 'should'. Normally, where 'should' appears in a formal clause of a standard, it is used to denote an advisory requirement. Our suspicion was aroused when, by browsing the lists of modal verbs' contexts, we noticed that nowhere did any modal verbs appear in a statement of the convention used to differentiate between mandatory and advisory requirements. We then discovered the following occurrence of 'should' in the standard: "17.7.6 Operators **should** be qualified and trained ...". This turned out to represent a mandatory requirement and hence violated the lexical convention that we had assumed for the standard.

ties .	should	include 1 . A list of CSRs ( result
16.2.2 Such a plan	should	also be consulted on the choice of
tor .	should	be used to test the functional beha
17.5.3.4.4 The Evaluator	should	be generated using a test pattern g
ing 17.6.1.3.1 Software simulation	should	provide adequate margins on critica
ing a set of test vectors , which	should	be alerted , both visually and aura
Assurance Report , and the design	should	be qualified and trained to a level
s been initiated , the operator(s)	should	be invoked by the Developer , in co
( AJST ) 5679 66 17.7.6 Operators	should	be forwarded to : Assistant Program
afety issues for such technologies	should	
the form and any additional papers	should	

Wrote 39 occurrences.

Figure 8 Occurrences of 'should' in the standard

Our exploration of the document's use of modal verbs revealed mistaken assumptions about the conventions used by the document. We eventually isolated the clause in the standard that defined the conventions used: mandatory requirements were indicated by bold text and advisory requirements were written in plain text. The REVERE toolset was unable to detect this because the tools currently do not include any formatting analysis.

Identifying the paragraph that defined the convention was complicated because we had to find words with semantic tags that corresponded to *mandatory* and *advisory*. This requires experimentation with several tags:

- S6 'Obligation and necessity'
- S8 'Helping/hindering'

The terminology in the standard for a mandatory requirements was simply the word 'requirement' This was tagged S6. The terminology used for an advisory requirement was the word 'guidance'. This was the tag S8. Clearly, in a standards document context, these two terms, as well as others such as 'mandatory requirement' and 'advisory requirement' should all be given the same tag. This revealed a problem in the use of a tagset derived from the analysis of general English for the analysis of technical or formal documents.

## Conclusions

REVERE's principal research contribution has been to pioneer the application of powerful probabilistic NLP techniques to the systems engineering domain. Although attempts have been made to deploy NLP to systems engineering, REVERE is, as far as we are aware, unique in having exploited probabilistic NLP and in having successfully demonstrated its application to real-world problems and data. The REVERE tools do not, for

example, only handle natural language subsets. Nor are they restricted to narrow application domains. Rather, they are robust in the face of the richness of language used and large volumes of documents. Moreover, the toolset can be applied to any systems engineering domain where requirements have to be synthesised from documents. This is certainly true for system evolution projects where it is necessary to rediscover requirements, and where reverse engineering is often the only viable way to do this. If there is an archive of documentation available, then REVERE's tools help the systems engineer rapidly abstract key requirements information (if not ready-formed requirements). As expected, the toolset is not foolproof and, as revealed by the standards example, the results need to be interpreted carefully. Similarly, the tagsets defined for general-purpose corpora can throw up anomalous results. We are convinced, however, that despite these shortcomings, an experienced systems engineer would be able to exploit statistical NLP tools to good effect.

An unplanned result of our experiments with the toolset was its general utility beyond purely legacy system evolution. It can be used anywhere large volumes of documents have to be assimilated in system development and other domains such as standards development. Some specific applications piloted during the project are:

- Document quality. A 'quality' viewpoint was implemented which applies a number of quality attributes for standards or specification documents. For example, 'weak' phrases that lead to ambiguity can be identified. We have used the quality viewpoint to replicate the functionality of NASA's ARM tool (Wilson, Rosenberg, Hyatt 1996) with the significant difference that the WMATRIX quality viewpoint uses POS and semantic tagging rather than regular expressions. This approach is more robust because it makes it easier to adapt to different contexts and to derive new quality attributes.
- The toolset can be seen as an enabling technology for the use of ethnographic studies of systems and processes. Whilst it has been shown that ethnography has much to offer systems engineering, the cost of synthesising requirements from the field reports that result from an ethnographic study is an inhibitor (amongst others) to more widespread take-up of the technique.

The problem that REVERE set out to address is one that is economically significant; systems evolution is an enormously costly activity. It is possible to view systems evolution as a (large) subset of the more general problem of systems engineering and REVERE's relevance spans this wider context. Given this, the impact of REVERE on global development costs is not likely to be high. However, there are a number of document-centric application domains where we envisage substantial potential. Domains tend to evolve document-centric processes as a defence against high risks; safety-critical systems, for example. Hence, any support that can be provided for systems engineers is likely to have high potential impact in individual cases because of the cost multiplier that attaches to overlooked requirements, failed understanding of actors' roles or poorly formulated standards clauses. A long-term result of REVERE may be to so reduce the cost of document processing that documentary archives become, for the first time, exploitable assets.

## **Acknowledgements**

We are grateful for the support of our industrial partner, Adelard, who have provided us with motivation, data and advice.

## **References**

- Aguilera C, Berry D. The Use of a Repeated Phrase Finder in Requirements Extraction, *Journal of Systems and Software*, 1990, 13 (9).
- Ambriola V, Gervasi V. Experiences with Domain-Based Parsing of Natural Language Requirements, *Proc. 4<sup>th</sup> International Conference NLDB '99*, Klagenfurt, Austria, 1999.
- Aston G, Burnard L. *The BNC Handbook: Exploring the British National Corpus with SARA*, Edinburgh University Press, 1998.
- Bentley R, Rodden T, Sawyer P, Sommerville I, Hughes J, Randall D, Shapiro D. Ethnographically-informed systems design for air traffic control, *Proc. CSCW '92*, Toronto, November 1992.
- Berry D, Yavne N, Yavne M. Application of Program Design Language Tools to Abbott's method of Program Design by Informal Natural Language Descriptions, *Journal of Software and Systems*, 7, 1987.
- Butler K, Esposito C, Hebron R. Connecting the Design of Software to the Design of Work, *Communications of the ACM*. 42 (1), 1999.
- Cyre W, Thakar A. Generating Validation Feedback for Automatic Interpretation of Informal Requirements, in *Formal Methods in System Design*, Kluwer, 1997.
- Fliedl G, Kop C, Mayr H, Mayerthaler W, Winkler C. Linguistically Based Requirements Engineering - the NIBA Project, *Proc. 4<sup>th</sup> International Conference NLDB '99*, Klagenfurt, Austria, 1999.
- Garside R, Smith N. A Hybrid Grammatical Tagger: CLAWS4, in Garside R, Leech G, McEnery A. (eds.) *Corpus Annotation: Linguistic Information from Computer Text Corpora*, Longman, 1997.
- Goldin L, Berry D. AbstFinder, A Prototype Natural Language Text Abstraction Finder for Use in Requirements Elicitation, *Automated Software Engineering*, 4, 1997.
- Jackson D, Jackson M. Problem decomposition for reuse, *BCS/IEE Software Eng. J.*, 11 (1), 1996.

- Rayson P, Wilson A. The ACAMRIT semantic tagging system: progress report, *Proc. Language Engineering for Document Analysis and Recognition (LEDAR)*, Brighton, England. 1996.
- Rayson P, Emmet L, Garside R, Sawyer P. The REVERE project: experiments with the application of probabilistic NLP to systems engineering, *Proc. 5<sup>th</sup> International Conference on Applications of Natural Language to Information Systems (NLDB'2000)*, Versailles, France. June 2000.
- Rayson P, Garside R. Comparing corpora using frequency profiling. Proceedings of the workshop on Comparing Corpora, held in conjunction with the 38th annual meeting of the Association for Computational Linguistics (ACL 2000). 2000: 1 - 6.
- Robertson S, Robertson J. *Mastering the Requirements Process*, Addison Wesley, 1999.
- Rolland C, Proix C. A Natural Language Approach for Requirements Engineering, *Lecture Notes in Computer Science*, Vol. 593, 1992.
- Rosenburg L, Hammer T, Huffman L. Requirements, Testing & Metrics, *Proc. 15<sup>th</sup> Annual Pacific Northwest Software Quality Conference*, Utah, USA, 1998.
- Sommerville I, Sawyer P, Viller S. Viewpoints for Requirements Elicitation: a Practical Approach, *Proc. Third IEEE International Conference on Requirements Engineering (ICRE 98)*, April 1998.
- Sommerville I, Sawyer P, Viller S. Managing Process Inconsistency using Viewpoints, *IEEE Transactions on Software Engineering*, 25 (6), 1999.
- Steuten A, van de Reit R, Dietz J. Linguistically Based Conceptual Modeling of Business Communication, *Proc. 4<sup>th</sup> International Conference NLDB '99*, Klagenfurt, Austria, 1999.
- Thomas J, Wilson A. Methodologies for Studying a Corpus of Doctor-Patient Interaction, in Thomas J, Short M. (eds.) *Using Corpora for Language Research*, Longman, 1996.
- Wilson W, Rosenberg L, Hyatt L. Automated quality analysis of natural language requirement specifications, in *Proceedings of Fourteenth Annual Pacific Northwest Software Quality Conference*. 1996.