ReSIST: Resilience for Survivability in IST

A European Network of Excellence

Contract Number: 026764

Deliverable D24: Thesaurus and ontology of resilience terms

Report Preparation Date: February 2008
Classification: Public

Contract Start Date: 1st January 2006
Contract Duration: 36 months
Project Co-ordinator: LAAS-CNRS
Partners: Budapest University of Technology and Economics
City University, London
Technische Universität Darmstadt
Deep Blue Srl
Institut Eurécom
France Telecom Recherche et Développement
IBM Research GmbH
Université de Rennes 1 – IRISA
Université de Toulouse III – IRIT
Vytautas Magnus University, Kaunas
Fundação da Faculdade de Ciencias da Universidade de Lisboa
University of Newcastle upon Tyne
Università di Pisa
QinetiQ Limited
Università degli studi di Roma "La Sapienza"
Universität Ulm
University of Southampton

Information Society Technologies

SIXTH FRAMEWORK PROGRAMME

Table of Contents

Introduction 3
1. The project: building a thesaurus and an ontology of dependability and security 4
2. The Corpus 4
3. The Tools 4
- 3.1 Linguistic analysis 4
- 3.2 Term extraction grammar 4
- 3.3 Special requirements 5

4. Term extraction – the process 5
- 4.1 Choosing the extraction set 5
- 4.2 Defining stop words 5
- 4.3 Term extraction 6
- 4.5 Judging term candidate informativity 6

5. Building a basic taxonomy 7
6. Document Retrieval 7
7. Indexation 8
8. Computing vector correlation 8
9. Automatic classification 9
10. Clustering 10
11. Visualization 12
12. Conclusions 12
13. Further work 13
References 14

Introduction

This is an interim report of Task IT-T3 *"Resilience Thesaurus and Ontology"* of Workpackage WP1 "*Integration Technologies"* of the ReSIST Network of Excellence.

Task IT-T3 is described in the Workplan Update D8 as follows:

The purpose of this task is to refine the existing resilience ontology that is based on the "ALRL" taxonomy and to compile a thesaurus of resilience terms extracted from selected technical publications. A primary resource will be a major corpus of dependability conference papers, which will be utilized to create a deliverable version of the thesaurus and a refined ontology to be made available at month 24 (D24). The ontology and the thesaurus will be used in further development of the RKB and the methodology of resilience-explicit computing.

The present report presents a summary of all work of Task IT-T3 that was completed in 2007.

The work reported has benefited from contributions of the Institute of Applied Information Sciences (IAI) of the University of Saarland, Germany, partially supported via Vytautas Magnus University.

1. The project: building a thesaurus and an ontology of dependability and security

Research in the fields of dependability and security is of significant interest in most disciplines of computer science and engineering. As a consequence the diverse disciplines have introduced different terminologies to describe the concepts of dependability and security. The growing complexity of those fields makes it imperative to build a thesaurus and an ontology of concepts that relate the multiple terminologies. A taxonomy of dependability and security that evolved over the past 25 years within the IFIP Working Group 10.4 and IEEE Computer Society Technical Committee on Fault-Tolerant Computing has been presented in (Avizienis et al. 2004) and serves as the starting point for this research.

2. The Corpus

The corpus of text used in this research is composed of nearly 2000 papers presented at the 29 annual International Symposia on Fault-Tolerant Computing (1971-1999) and at their successors, the 7 International Conferences on Dependable Systems and Networks (2000-2006), The text files were created from PDFs using the free pdftotext tool delivered with the Unix program XPDF. Calculated by the UNIX word count tool, it contains 10,949,250 tokens.

3. The Tools

3.1 Linguistic analysis

The linguistic analysis of the ReSIST articles is the very basis of each following step. The analysis is performed with the program MPRO, a morphological analyser developed at the IAI. MPRO yields a feature bundle analysis of every token[1] in the text, for instance (1) which shows an analysis of the word "operational".

(1) `{ori=operational, ds=operate~ion~al,...}`

On top of the morphological analysis, grammars for shallow parsing, disambiguation and term detection can be applied to the morphologically annotated text (Carl et al. 1997, Carl & Schmidt-Wigger 1998). The disambiguation and shallow parsing grammar detects phrases, clauses and sentence boundaries.

3.2 Term extraction grammar

The term detection grammar is a grammar which uses heuristics for term and proper name detection[2], conceived in the MULTIDOC project[3] and used for practical purposes in CLAT[4]. Other

[1] Words as well as punctuation marks, internet addresses etc.

[2] A typical but simple heuristical rule is that unknown words after a title like „professor" will probably be names.

[3] http://www.iai.uni-sb.de/iaide/en/multidoc.htm

versions exist in the form of AUTOTERM (Haller 2006; Hong et al. 2001). It detects term candidates, i.e. noun phrases that may – by the interpretation of our heuristic rules – be real terms. Whether they actually are terms must be judged by a different procedure, possibly performed by human experts.

3.3 Special requirements

While these tools are readily available at the IAI, they have to be adapted to the special requirements of each task and each kind of texts they are applied on. Until now automatic natural language processing tools have not been applied to study the terminology of the field „dependability and security in computing“, further abbreviated as “D&S”. For instance, the tools that were used in automatic term extraction require a *stop word* list. Stop words are words that cannot be part of a term. For example, if “one” is a stop word, a term extraction application will (or should) not yield the term „one world idea“.

Having a stop word list as a first requirement, the second is that of finding high quality terms, as the extracted terms shall serve for building an ontology of the dependability and security (D&S) field.

4. Term extraction – the process

4.1 Choosing the extraction set

Earlier, we stated that the D&S terminology was to be extracted from the corpus described in section 1.1 above. This is only half the truth. First extraction experiments were conducted on the whole set of articles. It soon became clear, though, that this would yield a too great number of term candidates, including quite a number of very general terms such as “large system”. We then opted to only use the abstracts of the article collection, the assumption being that abstracts contain only the most important information and a set of rather meaningful term candidates. The abstracts subset of the corpus contains only 181,548 tokens.

4.2 Defining stop words

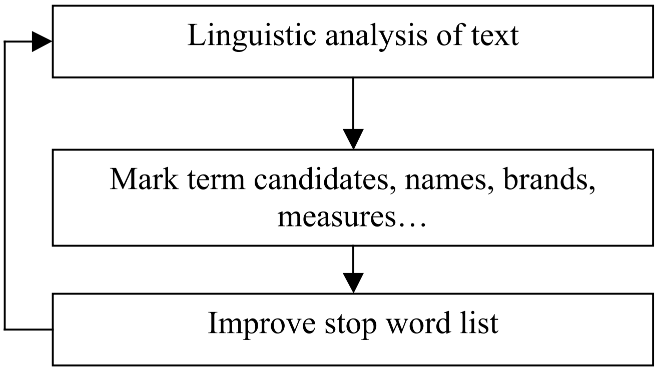

Linguistic analysis of text
Mark term candidates, names, brands, measures…
Improve stop word list

Figure 1: Building stop word lists.

[4] http://www.iai-sb.de/en/produkte/clat.htm

In the previous section, we have already introduced the notion of stop words. There are two ways how to define stop words:

1. Make up a list of *a priori* stop words.
2. Extract terms, look at the data and then define stop words.

We opted for a mixture of the two methods. First we chose a number of very general stop words that have been used in several applications of the IAI. These are stop words like "always" and "any". Then we ran the term extraction on our data set[5] and looked at the data manually, in order to find words that would appear regularly, but not contribute to the meaning of a term from the perspective of the D&S terminology. Such a stop word would be "large", as in "large system". Once we had defined a list of stop words, we iterated the process and looked at the terms once more in order to find out further stop words. This process yielded a total of 416 stop words, the vast majority of them being adjectives and adverbs, only few of them nouns like "illustration". The process was iterated a number of times until a random selection of term candidates showed only candidates that could not be ruled out as terms.

4.3 Term extraction

Once the stop word list had been completed, the term extraction rules were re-run on the abstracts. As previously stated, we expected the extraction to be a lot more efficient in terms of term candidate numbers and term quality. This eventually proved true. The extraction from the abstracts yielded 6,818 term candidates as opposed to >107,000 that had been extracted from the whole set of articles.[6]

4.5 Judging term candidate informativity

As we said in the beginning, one of the requirements of the thesaurus building project is to acquire terms with high quality. For the first phase of the project we opted for a rather simple mechanism of judging *term informativity*, i.e. whether a term candidate shows some discriminatory power. A term with good discriminatory power will not be too general, i.e., when the term occurs in a document, it will be a reliable indicator for what topic the article is about. On the other hand, we are also not interested in too specialised terms, i.e. such terms that only occur in very few documents and about whose term status we thus cannot be sure.

In order to apply these criteria automatically, we compute the *inverse document frequency* (IDF) for each term t where $|D|$ is the number of all documents in the collection and d a single document from the collection:

$$idf(t) = \log\left(\frac{|D|}{\{d : t \in d\}}\right)$$

[5] The stop word list was constructed on the set of complete articles, and reused as such for extraction from the abstracts.

[6] This strikingly high number of term candidates is owed to two factors: First, the system chose many formulae parts as term candidates. Second, the article set was a conversion from PDF to text; many words contained wrong characters, and the system would mark them as "unknown word" but "possibly a term".

From the IDF values obtained, we set thresholds stating that terms with *2 > idf(t) > 7* should not be part of the candidate set.

This very first and simple measure reduced the term candidate number from 6,818 to 5,710, ruling out singletons and terms like “system” which appear in virtually every document. We will investigate more recent methods, though, like the one suggested in (Park et al. 2002), where the frequency of a term candidate in the domain-specific corpus is compared to its frequency in a more general corpus.

For current purposes, the term candidate list is further reduced by manual inspection of the terms. For this, the terms are split into groups (e.g. by common root), and then inspected by a human D&S specialist with regard to other members of their group. 4,959 term candidates remained after manual inspection.

5. Building a basic taxonomy

The final goal of the project described here is to build an ontology of terms related to the Dependability and Security domain. The problem of building an ontology is that of finding term relations such as synonymy or antonymy. There are standard methods proposed e.g. by (Manning & Schütze 1999), which we haven't explored yet. Instead, we use a method to create a hierarchy from general to more special terms. For this we use another simple method; namely, we sort terms by their roots as follow:

- non-compound terms are top level hierarchy nodes;
- for a term t_x with *n* compound parts, we look up whether there is a term t_y consisting of the *n-1* rightmost term parts; if so, the term t_x becomes a subterm of t_y, such as “peer-to-peer storage system” is a subterm of “storage system” which in return is a subterm of “system”.

This sorting-by-roots is of course linguistically somewhat debatable. On the one hand, it only produces hierarchies, not the ontology we aim to create. On the other hand, the root is not always necessarily the rightmost part of a compound. If we take the term “operating system”, describing some abstract device that is meant to run a machine, and compare it to its human counter-part, the “operator”, we can at least think about whether the semantic focus weren't rather on the “operating” aspect than on the “system” aspect; in the term “operator”, the quality of being human is already encoded by the ending “-or”, yet we want to emphasise that somebody is running something rather than he or she is human.

6. Document Retrieval

An important part of the ReSIST research effort is the creation of a Resilience Knowledge Base (RKB) that combines diverse information sources. The RKB provides user-friendly interfaces and serves as a central repository of information on multiple aspects of dependable and secure computing, including researchers, active and completed projects, case studies of system failures with associated symptoms and post-mortem assessments, and of course a collection of research publications that has accumulated over many years.

As for every larger collection of documents, the following questions arise: How to make finding a document simple, and how to make finding documents simple that are related to the document we searched for? And, how can we visualize relations between documents in an easily graspable way?

Information retrieval has come up with a number of more or less standard approaches for more or less effective document retrieval, like hierarchical clustering techniques or automatic classification approaches. These techniques often rely on statistical analysis, sometimes linguistic analysis of documents, and say nothing about how to visualize the results.

In the following, we present a system which integrates all the stages that are useful for document retrieval: term extraction, linguistic analysis, clustering and classification techniques and finally visualization of results. At the time of writing, the system is in development.

7. Indexation

We have previously discussed a method for extracting terminology from a corpus based on linguistic analysis with extensive lexica and a fine-grained grammar. Once the terminology is extracted, the extracted terms are collected into a terminology database. Documents are reprocessed linguistically. For each document, we build what we call an *index* (cf. Ripplinger & Schmidt 2001), and contains the list of terms for each document and a term weight ranging from 0 (appears, but not important for this document) to 100 (very important for this document), as in the following, simplified example:

```
Descriptors: Network[100]; Error Message[54];
Free descriptors: Input-Output [100]; Network-Pair[89]
Possible Terms: redundant rs;
Toponyms: England [100];
Acronyms: SCHEME;
```

In our applications, we rely on the terms listed in the field "Descriptors". The descriptors are terms from the term database[7] that were found in the indexed document.

8. Computing vector correlation

We need a measure to compute the similarity between two documents. One of the standard measures is *correlation*; the correlation measure again is based on *covariance*. While covariance gives us direction of relations between two vectors, it tells us nothing about its strength. Correlation standardises this to a scale from +1 (perfect match) to -1 (perfect contradiction).

$$corr_{xy} = r_{xy} = \frac{\operatorname{cov}(x,y)}{\sqrt{\operatorname{var}(x)}\sqrt{\operatorname{var}(y)}}$$

In order to be able to apply this vector measure, we need to somehow interpret the documents as vectors. We use the document index for this purpose, or more precisely the descriptors that have been assigned to a document together with their weights, as in the following example:

[7] http://resist.ecs.soton.ac.uk/classifier/manual/

Document *a*:

computer system[100],
operating system[35],
network[20]

Document *b*:

network[100],
system message[56],
microprocessor[45]

We then parallelise the vectors of *a* and *b*. We use the descriptors as keys to the vector dimensions, much like in a hashtable, and first sort them alphabetically, their weights being the values for the dimensions. Then, we compare whether all descriptors from *b* are present in *a* and vice versa. If not, the descriptor is inserted with a value of zero. Finally, the vectors are resorted.

a = (**computer system[100]**, microprocessor[0], **network[20]**, **operating system[35]**, system message[0])
b = (computer system[0], **microprocessor[45]**, **network[100]**, operating system[0], **system message[56]**)

$$a: \quad \begin{pmatrix} 100 \\ 0 \\ 20 \\ 35 \\ 0 \\ 0 \end{pmatrix} \qquad b: \quad \begin{pmatrix} 0 \\ 45 \\ 100 \\ 0 \\ 0 \\ 56 \end{pmatrix}$$

Figure 2: The documents *a* and *b*, represented as vectors and parallelised.

Now we can apply the vector correlation measure which serves as basis for the classification and clustering techniques described in the following sections.

9. Automatic classification

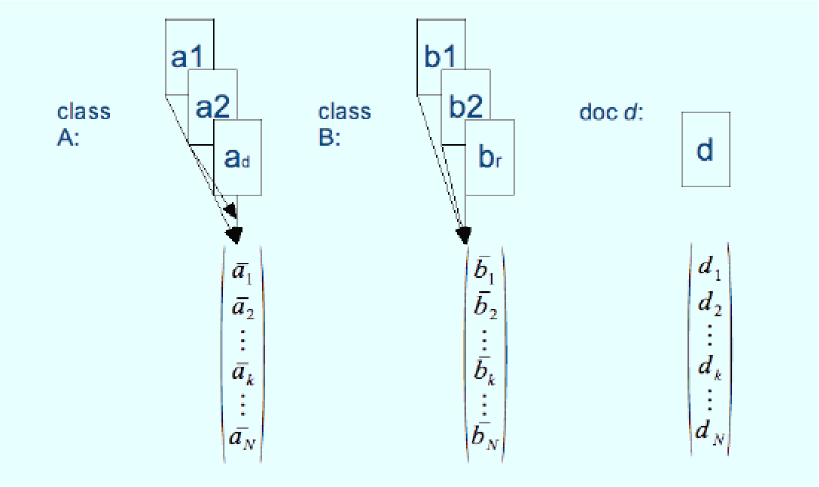
Automatic classification refers to the task of assigning documents to one or more from a predefined set of classes. A class definition for each class tells us which documents should belong to it. While this helps for manual classification, we need some other means for an automatic classification to be able to build up classification criteria.

One common method of building such classification criteria is to use a training set. To form the training set, a minimum number of documents is manually assigned to each class. Some algorithm is run over this manual assignment in order to “learn” or somehow else to extract classification criteria.

We employ the vector representation for the latter purpose. Each document from the manual classification set is assigned a vector. For each class, we then compute a mean vector as displayed in figure 3: $\bar{a}_1$ is the mean of all values of the slot *a* in each document etc. This mean vector represents the whole class. An automatic classification then reduces to the two following steps:

- for each document *d* and all classes *C*, calculate their vector correlation;

- insert *d* into all classes, for which the vector correlation is higher than some threshold *t*, but maximally to the *n* best classes.
-



-

Figure 3: Classes represented as vectors, very much like documents.

10. Clustering

Using the document similarities calculated by vector correlation, we could apply a simple hierarchical clustering algorithm (Manning & Schütze 1999). The biggest disadvantage of this algorithm is the distribution of cluster sizes; hierarchical clustering algorithms produce a smaller number of very big classes and a high number of very small classes. An alternative clustering method is *correlation clustering* as described in (van Gael & Zhu 2007). Correlation clustering takes into account correlations between one document and all other documents at once, as well as such constraints as *must-link* or *cannot-link*. As we have no constraints in our set to be clustered, we have not used correlation clustering as such, but have incorporated the former idea of using all correlations of one document to all others into our hierarchical clustering method.

A simple correlation measure, when visualised in a simplified two-dimensional scheme, gives us the correlation between two points in space.

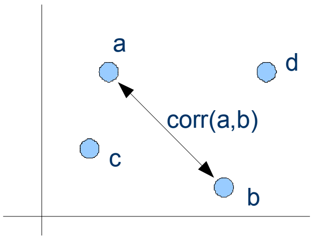


Figure 4: Correlation between two points in space.

Yet we gain more information if we correlate the correlations of either document to all other documents.

Figure 5 shows the documents *a* and *b* with the correlation arrows to the remaining documents *c* and *d*. Taking into account the correlation between one document to all other documents we gain information about the document's position in relation to the other documents.

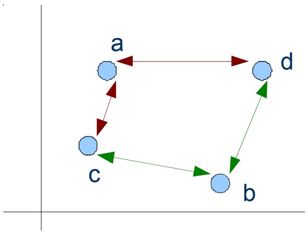


Figure 5: The correlation between a and c and all other documents

We can hold this information against the corresponding information for some other document and thus gain more than only knowing their position in relation to each other.

For two documents d_i and d_j, we thus compute the correlation as follows:

$$corr_{ext}(d_i, d_j) = corr\left(\begin{pmatrix} corr_{1i} \\ corr_{2i} \\ \ldots \\ corr_{ij} = 0 \\ \ldots \\ corr_{ji} = 0 \\ \ldots \\ corr_{ni} \end{pmatrix}, \begin{pmatrix} corr_{1j} \\ corr_{2j} \\ \ldots \\ corr_{ji} = 0 \\ \ldots \\ corr_{jj} = 0 \\ \ldots \\ corr_{nj} \end{pmatrix} \right)$$

The advantage of this extended correlation measure $corr_{ext}$ is that the similarity values obtained in this way are distributed differently than by the standard correlation measure. The contrast between most similar documents and not-so-similar documents is a lot higher, as shown in the following examples.

Table 1 shows the list of the 5 most similar documents, calculated by standard correlation for a random document.

Standard correlation		Extended correlation	
20060901177.txt	0.334	20061006303.txt	0.624
20060902360.txt	0.164	20060901174.txt	0.428
20061006303.txt	0.114	20060903142.txt	0.366
20061101073.txt	0.065	20060901179.txt	0.303
20061103707.txt	0.016	20060902422.txt	0.274

Table 1: Vector correlation values for standard (left) and extended correlation measure (right) between document 20060701125.txt and the top 5 most similar documents.

We can clearly see the difference between the first list and the second list. First, the two lists only have one document in common (highlighted in light grey) which is not even in the same position on both lists. Second, the extended correlation does not only produce higher values, but also a higher contrast. The difference e.g. between the first and the fifth document on the list is around 0.32, while it is 0.35 on the second list. This being true for quite a number of documents, we can more clearly include or exclude documents from a cluster – resulting in more balanced cluster sizes than when using standard correlation.

11. Visualization

As we said in the beginning, the system we are describing here not only takes care of the clustering viz. classification, but is also meant to visualize the results in a manner easily graspable by users, rather than e.g. in a grouped list of document names.

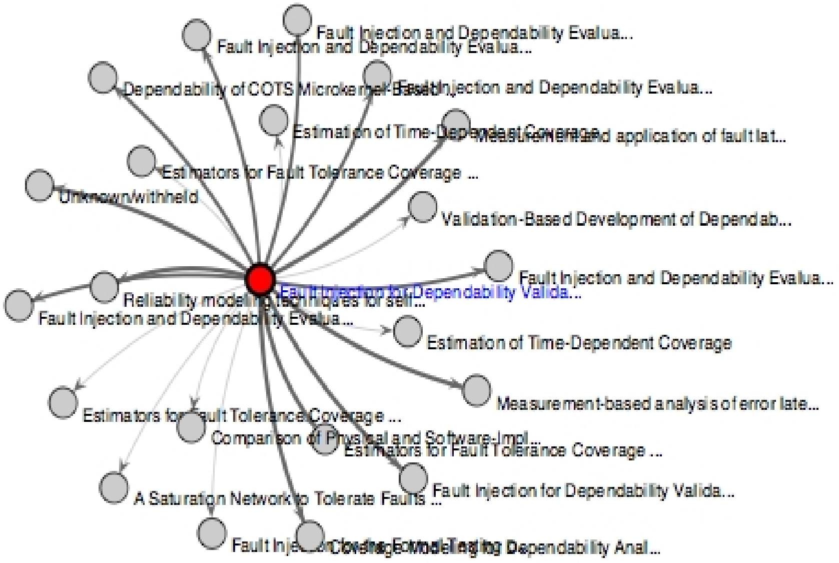

Fault Injection and Dependability Evalua...
Fault Injection and Dependability Evalua...
Dependability of COTS Microkernel-Based...
Fault Injection and Dependability Evalua...
Estimation of Time-Dependent Coverage
Measurement and application of fault lat...
Estimators for Fault Tolerance Coverage ...
Unknown/withheld
Validation-Based Development of Dependab...
Fault Injection and Dependability Evalua...
Fault Injection for Dependability Valida...
Reliability modelling techniques for self...
Fault Injection and Dependability Evalua...
Estimation of Time-Dependent Coverage
Measurement-based analysis of error late...
Estimators for Fault Tolerance Coverage ...
Comparison of Physical and Software-Impl...
Estimators for Fault Tolerance Coverage ...
Fault Injection for Dependability Valida...
A Saturation Network to Tolerate Faults ...
Fault Inje
Coverage Modeling for Dependability Anal...

Figure 6: Proposed RKB visualisation: A selected document, and documents related to it.

While at the time of writing the system is not yet ready for use, a basic setup is already available in the RKB[8] which will later be linked to the clustering results.

Figure 6 shows such an envisioned visualization. This assumes that a user has chosen an article in the database, because he or she knows the article (represented by the circle in the center). Clicking on the article, the user will be shown a net, where the original article is linked by arrows to other articles that are thematically related to it.

12. Conclusions

The first stage of the D&S thesaurus and ontology building project had two goals:

- create domain specific resources (stop word lists, adaption of English grammar) for D&S term extraction;
- for an extracted list of term candidates, find automatic methods of judging term quality.

[8] The ReSIST knowledge browser, available under http://resist.ecs.soton.ac.uk/explorer/

While point one has been completed, point two has only been achieved by very basic methods so far, which have proven to be effective to some point. The goal of the second project stage will be to explore further methods for judging term candidate quality, and to explore automatic methods for creating ontologies. Still, the results were good enough to serve for creating statistic methods for clustering and automatic classification. (Gindiyeh et al., 2007).

Currently using the described methods we have produced 800 clusters. Out of almost 2000 documents, only 60 remain ungrouped. Manual inspection of the cluster suggests a more than acceptable quality of the results.

While the project has achieved the linking of standard and non-standard methods both from term extraction and document retrieval, it currently lacks both an evaluation procedure and thus a decision: which procedure, clustering or classification, will produce the better results? This question will remain to be answered in the second half of the project.

13. Further work

As a consequence of the work having been done in 2007, a mini project was proposed by VMU and Ulm University to the ReSIST EB and has been granted for the year 2008. Staff members of the IAI will take part in the work as Affiliate members of VMU Kaunas. The following tasks will be performed during 2008:

1. **Term extraction.** The term extraction process will be performed for two other sets of documents: (a) the abstracts of the papers presented at the IEEE Oakland Conference on Security and Privacy, and (b) the technical documents generated by the ReSIST NoE project up to the present time. The additional terms obtained will be used to enrich the thesaurus generated from the FTCS+DSN corpus.
2. **Classification experiment.** The results of the currently ongoing manual classification procedure will be used to implement an automatic classification experiment for the FTCS+DSN corpus. The results will be validated by D&S domain experts and used to build the upper nodes of the thesaurus.
3. **Thesaurus enrichment.** Synonymy and other relations will be determined and used to enrich the thesaurus of the D&S domain. The hierarchical thesaurus and clustering results will be used in experiments on computational techniques for creating ontologies. Currently there are 4 approaches for automatic synonymy (and possibly other relation) extraction: (a) The word window approach, (b) Term relation discovery via clusters, (c) Term coherence approach, (d) Re-translation lookup applied to the thesaurus. Comparison of the results of two or more approaches will serve to validate the results.
4. **Ontology representation.** A logic representation of the D&S domain ontology will be created using the enriched thesaurus. Separately, the ALRL taxonomy will be manually restructured by introducing the logic representation of other relations.
5. **Visual analysis.** The interactive visualization and analysis techniques developed at Ulm university will be employed to represent and to analyze the D&S domain ontology.

The research effort in 2008 will be led by co-leaders Gintare Grigonyte (VMU Kaunas) and Thorsten Liebig (Ulm), with Olaf Noppens (Ulm) and Affiliate members Oliver Culo and Mahmoud Gindiyeh taking part. Consulting participants will be Friedrich von Henke (Ulm), Algirdas Avizienis and Ruta Marcinkeviciene (VMU Kaunas), Johann Haller and Michael Carl (affiliate members from IAI). We expect active interest from LAAS (J.-C. Laprie), Newcastle (B. Randell) and Southampton (H. Glaser and I. Millard).

References

Avizienis, Algirdas, Jean-Claude Laprie, Brian Randell & Carl Landwehr 2004. Basic concepts and taxonomy of dependable and secure computing. In: *IEEE transactions on dependable and secure computing*. IEEE Computer Society Press, vol.1, no.1, January-March 2004, pp.11-33.

Avizienis, Algirdas, Oliver Culo, Gintare Grigonyte, Ruta Marcinkeviciene 2007. Building a Thesaurus and an Ontology of the Concepts of Dependability and Security. In: *DSN 2007.*

Carl, Michael & Antje Schmidt-Wigger 1998. Shallow Post Morphological Processing with KURD. In: *NeMLaP '98*. Sydney

Carl, Michael, Antje Schmidt-Wigger & Munpyo Hong 1997. KURD - A Formalism for Shallow Post Morphological Processing. In: *Proc. of NLPRS'97*

Culo, Oliver, et al. 2007. Building a Thesaurus of Dependability and Security: a Corpus Based Approach in *Proc. Of 3rd Baltic Conf. On Human Language Technologies*, Kaunas, Lithuania

Gindiyeh, Mahmoud, et al. 2007. A Document Classification tool for the Resilience Knowledge Base of the ReSIST NoE Project, in *Proc. Of 3rd Baltic Conf. On Human Language Technologies*, Kaunas, Lithuania

Haller, Johann 2006. AUTOTERM – automatische Terminologieextraktion Spanisch-Deutsch. In: *Multiperspektivische Fragestellungen der Translation in der Romania*, Alberto Gil/Ursula Wienen (eds.). Peter Lang Verlag, Frankfurt. 229-242.

Hong, Munpyo, Sisay Fissaha & Johann Haller 2001. Hybrid Filtering for Extraction of Term Candidates from German Technical Texts. In: *Terminologie et intelligence artificielle*. Rencontres No4, Nancy , FRANCE. 223-232.

Maas, Heinz Dieter 1998. Multilinguale Textverarbeitung mit MPRO. In: *Europäische Kommunikationskybernetik heute und morgen '98*. Paderborn.

Manning, Chris & Hinrich Schütze 1999, *Foundations of Statistical Natural Language Processing.* MIT Press Cambridge, MA.

Park, Youngja, Roy J. Byrd & Branimir K. Boguraev 2002. Automatic glossary extraction: beyond terminology identification. In: *Proceedings of the 19th international conference on computational linguistics*. Taipei, Taiwan.

Ripplinger, Bärbel & Paul Schmidt 2001. Automatic Multilingual Indexing and Natural Language Processing. submitted to *SIGIR'01*.

Schmidt-Wigger, Antje 1998. Building Consistent Terminologies. In: *Proceedings of COMPUTERM '98.*

Van Gael, Jurgen & Xiaojin Zhu 2007. Correlation clustering for crosslingual link detection. In: *International Joint Conference on Artificial Intelligence (IJCAI) 2007.*