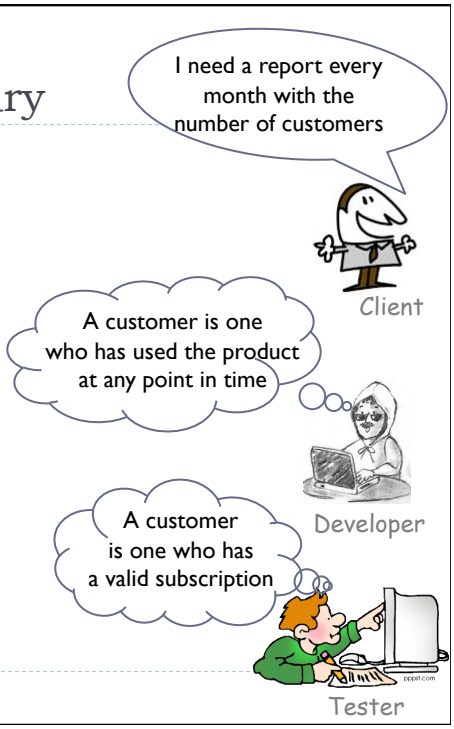
Automatic Extraction of Glossary Terms from Natural Language Requirements

Anurag Dwarakanath, Roshni R. Ramnani, Shubhashis Sengupta
Accenture Technology Labs
Bangalore, India

The Need for a Glossary

- Most Software Projects have their requirements documented in Natural Language (Colin and Laplante, 2003)
- A glossary contains all domain concepts mentioned in the requirement document and their definitions (Ricardo, Sawyer & Gervasi, 2011)
- A glossary ensures consistent interpretation by all stakeholders thus reducing rework



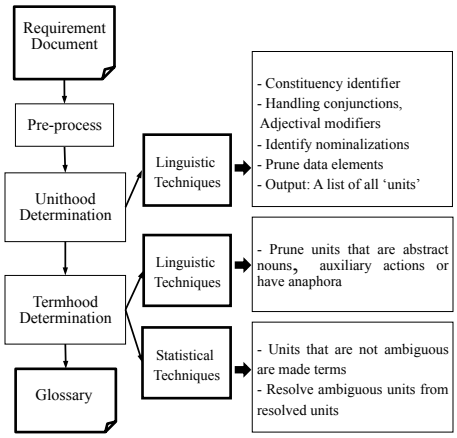
Automatic Glossary Extraction

- We describe a method for the automatic extraction of Glossary Terms (not definitions) from a single Business Requirements' document.
- Term Extraction has been studied in Information Retrieval & Requirements Engineering for long now.
- This is a difficult problem to solve:
  - Need to identify Terms from *any* domain (i.e. Insurance, Telecom, etc). There is no domain specific 'normative' corpus
  - Requirements are written in a style that *repeats* sentence structures.
    - This repetition is incidentally promoted by Requirements Best Practices.
    - Having same/similar sentence structures throughout the document causes 'frequency depended' techniques to suffer since they pick up 'frequent' words/patterns.
  - Requirements contain 'co-ordinations' (and/or) between terms. Identifying atomic terms requires solving 'co-ordination ambiguity'. (Chantree, Willis, Kilgarriff & De Roeck, 2007)
  - Requirements contain 'adjectival ambiguity'
  - Requirements contain 'nominalization' – i.e. an 'action term' being used in the grammatical function of an 'entity term'. (eg: synchronisation) (Rupp & Goetz, 2000)

High Level Idea

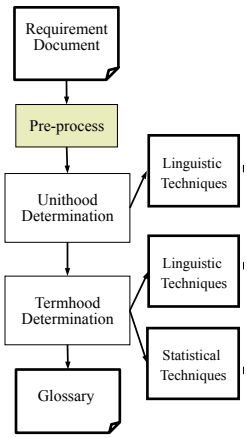
- Use NLP techniques to identify candidate terms
- Use Linguistic attributes to Prune candidates
  - Physical / Abstract attributes of Nouns
  - Concrete / Auxiliary attributes of Verbs
- Use 'frequency' to solve ambiguity

Automatic Glossary Term Extraction



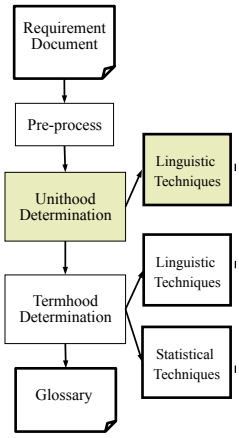
- The glossary creation is driven by two basic modules:
  - Unithood
  - Termhood
- Unithood: Identifies candidate terms for the glossary
- Termhood: Selects among these candidate terms

Preprocessing



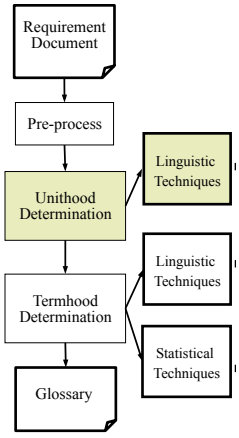
- Pre-processor extracts sentences which have requirement labels
  - Eg: Req-01
- Identifies Acronyms, URLs, content in quotes
  - Exceptions: 'ONLY', 'NOT'
- Cleans up the sentence
  - Removes special characters, brackets
- Every sentence is then parsed using 'Link Grammar'
  - An Open Source dependency parser from CMU (Sleator & Temperly, 1993) .

Unithood Determination (1/6)



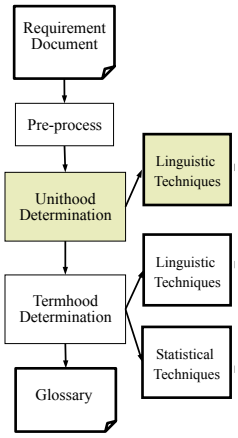
- Unithood identifies all 'units' in a sentence
  - A unit is a word or phrase that behaves as a structural block.
  - Every unit is a candidate for a Term.
  - Eg: 'password should be' is not a unit.
  - Eg: 'valid password', 'its strength'
- The definition of 'Units' is driven from that of a 'constituent' in a parse of Natural Language sentence.
- The 'units' are the Noun Phrase Constituent and Verbs (not verb phrase) from the parse of LG

Unithood Determination (2/6)



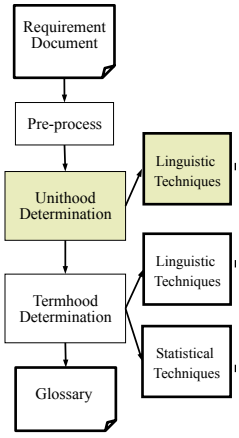
- We also interpret the parse of a sentence to identify 'multiple' units from a single constituent
  - Multiple units align with multiple interpretations possible (i.e. ambiguity)
  - The correct unit from the multiple options is chosen in 'Termhood'.
- Handling Co-ordinating Conjunctions
  - Co-ordinations (and/or) can occur between most Parts-of-Speech, most common being among nouns (Chantree, Willis, Kilgarriff & De Roeck, 2007)
  - We use the 'dictionary' of LG to identify all possible uses of co-ordinations.
  - We developed rules to handle each case explicitly

Unithood Determination (3/6)



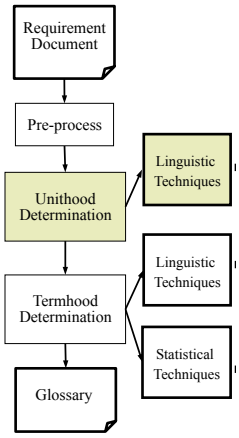
- Co-ordinating Conjunctions around Nouns (LG link 'Sj')
  - Eg: Constituent - 'sales and marketing user'
  - Units created:
    - 'sales', 'marketing user'
    - 'sales user', 'marketing user'
    - 'sales and marketing user'
- Co-ordinating Conjunctions around Adverbs (LG link 'Rj')
  - Eg: Constituent – 'efficiently and securely compute'
  - Units created
    - 'compute'
- Similarly, we generate units for co-ordinations between Adjectives (Link Aj), Verbs (Link Vj), Prepositions (Link Mj) and Independent Clauses (Link CC and Link XX)
- The interpretation of 'or' is exactly similar to 'and', though we need to note a subtle difference:
  - 'admin or superuser rights'

Unithood Determination (4/6)



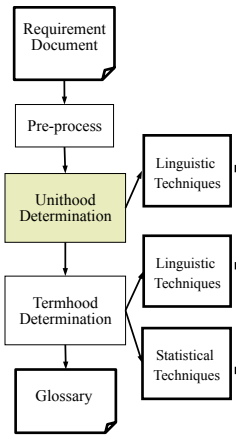
- Handling Adjectival Ambiguity
  - We identify patterns in the parse where the constituent is modified by an Adjective
  - Eg: Constituent - 'patient monitoring system'
    - Units generated:
      - 'monitoring system' , 'patient monitoring system'
  - Eg: Constituent – 'numeric keypad'
    - Units generated:
      - 'keypad, numeric keypad'

Unithood Determination (5/6)



- Identifying and Handling Nominalization
  - Nominalization is a noun semantically pointing to a verb
  - We identify nominalization by checking the 'Infinitive' of the head word of the constituent
    - The 'Infinitive' of a nominalization is a verb
    - Algorithm provided in the paper
  - Multiple Units are created:
    - Eg: Constituent – 'email synchronisation'
    - Units Created:
      - 'email'
      - 'synchronisation'

Unithood Determination (6/6)

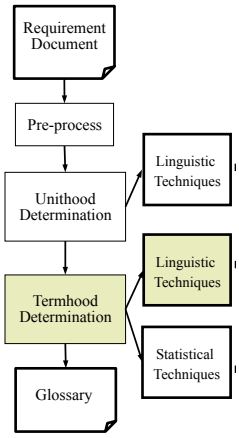


- The generated Units are maintained in a data structure (below).

$$U = \begin{cases} u_{11}, u_{12}, u_{13} \cdots u_{1m} \\ u_{21}, u_{22}, u_{23} \cdots u_{2m} \\ \vdots \\ u_{k1}, u_{k2}, u_{k3} \cdots u_{km} \end{cases}$$

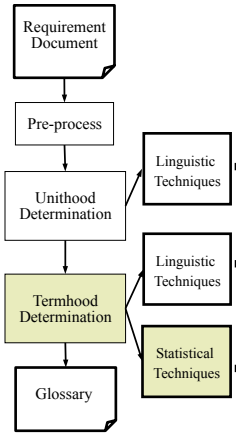
- Eg: Constituent – 'sales and marketing user'
- u_{11}= 'sales', u_{12}= 'marketing user',
- u_{21}= 'sales user', u_{22}= 'marketing user', u_{31}= 'sales and marketing user'

Termhood Determination (1/3)



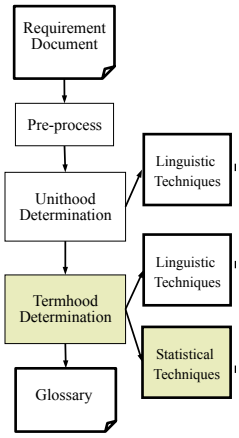
- Termhood either accepts a Unit or rejects a Unit as a term
  - Prune Abstract Nouns
    - Abstract nouns are those that cannot be perceived by the senses
    - The infinitive of the head word of the Unit is an adjective
    - Eg: Unit – 'capability'
  - Prune Auxiliary Actions
    - Auxiliary actions are those that support other content bearing actions
    - Identified through a look-up (auxiliary actions are few in number)
    - Eg: Unit – 'have', 'be'
  - Algorithm provided in the paper

Termhood Determination (2/3)



- We use a Statistical metric to resolve ambiguity (by choosing the right Unit from the multiple options)
  - The Metric is inspired by in-document statistical approach (Matsuo & Ishizuka, 2004) or unsupervised ambiguity detection (Ratnaparkhi, 1998)
  - Use unambiguous cases to resolve ambiguous ones
- Recall – 'sales and marketing user'
  - If elsewhere in the document, 'sales user' has been used, we can resolve the ambiguity

Termhood Determination (3/3)



$$g(a) = \begin{cases} 1 \; if \; f(a) > 0 \\ 0 \; otherwise \end{cases}$$

$$Term\ (U) = \begin{cases} option\ U_i \; if \; argmax_{i=1\ to\ k-1} \sum_{n=1}^{n=m} g(U_{in}) \geq 1 \\ else\ option\ U_k \; if \; f(U_{k1}) \geq \frac{\sum_{\forall i} f(T_i)}{i} \end{cases}$$

- Where U is the data structure made by Unithood

Experimental Results

- We test our algorithm over 5 real-life requirements' document
- Our approach is compared against a base algorithm
  - Base algorithm identifies all Noun Phrases (NPs) and the Verbs as Terms

		Requirements' Documents					Total/Avg
		1	2	3	4	5	
Number of Requirements		108	165	183	110	147	566
number of Terms in Entity list by manual inspection		341	593	564	511	452	2461
number of Terms in Action list by manual inspection		118	313	285	248	218	1182
Accuracy of LG		.96	.82	.85	.91	.95	.89
Our Method	F1 measure Entity list	.90	.85	.80	.83	.83	.84
	F1 measure Action list	.86	.86	.89	.89	.81	.86
Base Algorithm	F1measure Entity list	.85	.76	.67	.78	.71	.75
	F1 measure Action list	.43	.47	.55	.56	.48	.50

Performance on Individual Steps of our algorithm

Aspect	Actual Count	No. identified / resolved	Precision	Recall
Abstract Nouns	154	170	.66	.73
Process Nouns	185	213	.65	.74
Auxiliary Actions	891	721	.98	.80
Co-ordination ambiguity	59	2	1	.03
Adjectival Ambiguity	200	119	.73	.43

- Identification of Auxiliary Actions have worked well, with Identification of Abstract and Process being moderate
- Adjectival Ambiguity resolution has worked moderately while co-ordination ambiguity resolution has been very poor

Conclusion & Future Work

- Our method of Term Extraction has shown superior results over the base algorithm
- Shortcomings of our Technique
  - Requirements contains many IT terms
    - E-mail, Adobe PDF
  - English Functional Words are being selected as Terms
    - 'data', 'color'
  - The linguistic properties checked ignores the 'sense' of the word
    - Eg: 'commitment'
  - Co-ordination Ambiguity Resolution remains a challenge
- Future Work
  - An NER (Named Entity Recognition) technique to identify IT Terms
  - A better approach for Linguistic Classification
  - A quantitative evaluation against related algorithms

References

- (Colin and Laplante, 2003) Neill, Colin J., and Phillip A. Laplante. "Requirements engineering: the state of the practice." Software, IEEE 20, no. 6 (2003): 40-45.
- (Ricardo, Sawyer & Gervasi, 2011) Gacitua, Ricardo, Pete Sawyer, and Vincenzo Gervasi. "Relevance-based abstraction identification: technique and evaluation." Requirements Engineering 16, no. 3 (2011): 251- 265.
- (Rupp & Goetz, 2000) Rupp, C., and R. Goetz. "Linguistic Methods of Requirements-Engineering (NLP)." In Proceedings of the European Software Process Improvement Conference (EuroSPI), pp. 7-11. 2000.
- (Chantree, Willis, Kilgarriff & De Roeck, 2007) Chantree, Francis, Alistair Willis, Adam Kilgarriff, and Anne De Roeck. "Detecting dangerous coordination ambiguities using word distribution." Recent Advances in Natural Language Processing. Current Issues in Linguistic Theory, 4 (292). (2007): 287-296.
- (Matsuo & Ishizuka, 2004) Matsuo, Yutaka, and Mitsuru Ishizuka. "Keyword extraction from a single document using word co-occurrence statistical information." International Journal on Artificial Intelligence Tools 13, no. 01 (2004): 157-169.
- (Ratnaparkhi, 1998) Ratnaparkhi, Adwait. "Statistical models for unsupervised prepositional phrase attachment." In Proceedings of the 17th international conference on Computational linguistics-Volume 2, pp. 1079-1085. Association for Computational Linguistics, 1998.

References

- (Sleator & Temperly, 1993) Sleator, Daniel DK, and Davy Temperley. "Parsing English with a link grammar." In Third International Workshop on Parsing Technologies (1993)
- (Infinitive 2012) Infinitive. http://en.wikipedia.org/wiki/Infinitive (retrieved on Dec 7, 2012)

Overview of Related Work

- In the field of Information Retrieval, studied as 'Automatic Term Extraction (ATR)'
- Typically Statistical Techniques are developed here
  - Frequency counts, Tf-idf, chi-squared
- An over-dependence on frequency of occurrence can cause non-domain terms to be selected:
  - Requirements being written by a single analyst has similar grammar (or sentence structure) in a large number of sentences
  - For example, in our test we witnessed the same sentence structure in 62% of the requirements

Overview of Related Work

- Eg: Consider a requirement: 'the customer should *be able to* change passwords after login'
- Linguistically, 'be' is an 'auxiliary verb', while 'change' is a concrete verb (a semantic content bearing verb)
- Consider a requirement: 'the customer should have the capability to change passwords'
- Here 'capability' is an abstract noun, while 'customer'; is a concrete noun
- A glossary term consists of concrete nouns & verbs

- Requirements also contain co-ordinations ('and' and 'or')
  - Eg: 'create and delete'
  - In our tests, co-ordinations occurred in 22% of the requirements
- However, handling co-ordinations requires solving 'co-ordination ambiguity'
  - 'commerce & finance websites'
- We also attempt to solve adjectival ambiguity
  - 'patient monitoring system'

- We also attempt to identify 'nominalization' (Rupp & Goetz)
  - A Nominalization (or a process noun) is a word that has the gramatical function of a noun, but semantically points to an action
    - Eg: 'creation', 'validation', etc