Recovering Traceability Links in Requirements Documents

Zeheng Li  Mingui Chen  LiGuo Huang
Department of Computer Science and Engineering
Southern Methodist University

Vincent Ng
Human Language Technology Research Institute
University of Texas at Dallas
What is a Software Requirement?

A software requirement is a description of a software system to be developed, laying out functional and non-functional requirements.
What is Requirements Traceability?

**Given:**
- a set of high-level (coarse-grained) requirements
- a set of low-level (fine-grained) requirements

**Goal:**
- Identify all the low-level requirements that refine each high-level requirement

**An important task in Software Engineering**
## An Example

| High-level requirements | Low-level requirements |
|-------------------------|------------------------|
| **HR01**                | **UC01**               |
| The underlined character in each menu shall be a shortcut key. | **Use case name:** store a contact’s info |
| **HR02**                | **Summary:** the address book should store a contact’s name, email and address |
| The system shall have an address book to store contacts. | **Description:** 1. enter “pine” in terminal 2. enter “a” to make address book 3. enter “@” 4. enter nickname and fullname 5. press ctrl+x to save the entry |
# An Example

## High-level requirements

| HR01 | The underlined character in each menu shall be a shortcut key. |

## Low-level requirements

| UC01 | **Use case name:** store a contact’s info  
  **Summary:** the address book should store a contact’s name, email and address  
  **Description:**  
  1. enter “pine” in terminal  
  2. enter “a” to make address book  
  3. enter “@”  
  4. enter nickname and fullname  
  5. press `ctrl+x` to save the entry |
## An Example

| High-level requirements | Low-level requirements |
|-------------------------|------------------------|
| **HR01**                | **UC01**               |
| The underlined character in each menu shall be a shortcut key. | **Use case name:** store a contact’s info |
| **HR02**                | **Summary:** the address book should store a contact’s name, email and address |
| The system shall have an address book to store contacts. | **Description:** |
|                         | 1. enter “pine” in terminal |
|                         | 2. enter “a” to make address book |
|                         | 3. enter “@” |
|                         | 4. enter nickname and fullname |
|                         | 5. press ctrl+x to save the entry |
An Example

| High-level requirements | Low-level requirements |
|-------------------------|------------------------|
| **HR01**                | **UC01**               |
| The underlined character in each menu shall be a shortcut key. | **Use case name:** store a contact’s info |
|                         | **Summary:** the address book should store a contact’s name, email and address |
|                         | **Description:** 1. enter “pine” in terminal 2. enter “a” to make address book 3. enter “@” 4. enter nickname and fullname 5. press ctrl+x to save the entry |

Goal: induce a many-to-many mapping
A very challenging NLP task

- ... for at least two reasons
  - Only a small portion of a document is relevant to the establishment of a link
  - Information relevant to the establishment of a link can be irrelevant to the establishment of another link
Previous Approaches

- **Manual approaches**
  - Identify traceability links by hand

- **Automatic approaches**
  - Establish a link between two requirements if their cosine similarity exceeds a certain threshold
    - Each document is represented as a bag of words or a bag of LDA-induced topics
Our Approach

- A supervised, knowledge-rich approach
  - Extends a baseline that uses only word pairs as features with two types of human-supplied knowledge
    - Word/phrase clusters
    - Annotator rationales
Word Clusters

- Two clusterings provided by domain experts
  - a verb clustering and a noun clustering
  - cluster-based features provide better generalizations
Word Clusters

- Two clusterings provided by domain experts
  - a verb clustering and a noun clustering
  - cluster-based features provide **better generalizations**

| Category              | Terms                          |
|-----------------------|-------------------------------|
| System Operation      | evoke, operate, set up, activate, log |
| Message Search        | search, find                  |
| Contact Manipulation  | add, store, capture           |
| Message Manipulation  | compose, delete, edit, save, print |
| Folder Manipulation   | create, rename, delete, nest  |
| Message Communication | reply, send, receive, forward, cc, bcc |
| User Input            | input, type, enter, press, hit, choose |
| Visualization         | display, list, show, prompt, highlight |
| Movement              | move, navigate                |
| Function              | support, have, perform, allow, use |

| Category  | Terms                             |
|-----------|----------------------------------|
| Message   | mail, message, email, e-mail, PDL, subjects |
| Contact   | contact, addresses, multiple addresses |
| Folder    | folder, folder list, tree structure |
| Location  | address book, address field, entry, address |
| Platform  | windows, unix, window system, unix system |
| Module    | help system, spelling check, Pico, shell |
| Protocol  | MIME, SMTP                        |
| Command   | shortcut key, ctrl+c, ctrl+m, ctrl+p, ctrl+x |
Word Clusters

- Two clusterings provided by domain experts
  - a verb clustering and a noun clustering
  - cluster-based features provide better generalizations

- Also attempted to induce the clusterings to reduce human effort in cluster creation
Annotator Rationales

- Proposed by Zaidan et al. (2007)

- Manually identify the words/phrases in each training document that are relevant to the establishment of a link (the rationales)

- Create additional training instances based on rationales
  - Allow the learner to train a better classifier by focusing on the relevant material
Evaluation

- Two datasets
  - Pine
    - 49 high-level requirements, 51 low-level requirements
    - Only 11% pairs have links
  - WorldVistA
    - 29 high-level requirements, 317 low-level requirements
      - 3.5 times larger than Pine
    - Only 5% pairs have links
Main Results

- When using both
  - annotator rationales (to create additional training instances)
  - word/phrase clusters (to create new features)
  to train a SVM classifier, our approach reduces relative error by 11-20% in comparison to the word-pair supervised baseline

- Results obtained using manual clusters are as good as those obtained using induced clusters
For details, please come visit our poster!