

# From Miners to Millionaires

Yanan Xiao

Masdar Institute of Science and Technology

Software Engineering Course Presentation

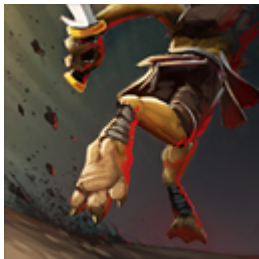
- 1 Introduction
- 2 Trustrace
- 3 Empirical Evaluation
- 4 Results
- 5 Discussion
- 6 Future Work

“It’s all about Mathematics.”

# Glossaries

- IR: Information Retrieval
- VSM: Vector Space Model
- JSM: Jensen-Shannon Model
- PCA: Principal Component Analysis
- Trustrace: Trust-Based Traceability
- TF/IDF: Text Frequency and Inverse Document Frequency
- MSW: Multiple-Static Weights

# Definition of Traceability



Traceability is the only means to ensure that the source code of a system is **consistent** with its requirements.

And that **all and only** the specified requirements have been implemented by developers.

# Traceability Sad Facts



During software maintenance and evolution, requirement traceability links become **obsolete**.

Because developers do not/cannot devote effort to updating them.



More sad: Recovering these traceability links later is a daunting and costly task for **developers**.

# State-of-the-Art Technique



The literature has proposed methods, techniques, and tools to recover these traceability links semi-automatically or automatically.

**Information Retrieval (IR)** techniques can **automatically** recover traceability links between free-text requirements and source code.



# A Mathematician's Approach

A set of requirements:

$$R = \{r_1, \dots, r_N\} \quad (1)$$

A set of classes:

$$C = \{c_1, \dots, c_M\} \quad (2)$$

A collection of sets:

$$T = \{T_1, \dots, T_P\} \quad (3)$$

where each  $T_i = T_1, \dots, T_{N_i}$  is a set of homogeneous pieces of information.

For each set  $T_i \in T$ , we build a set  $R2CT_{i,r_j,t_k}$  for each expert  $T_i$  as follows:

$$R2CT_{i,r_j,t_k} = \{(r_j, c_s, \sigma'_i(r_j, t_k)) | c_s \in \delta_{T_i}(t_k) \& t_k \in T_i\} \quad (4)$$

And we use the sets  $T_i \in T$  to build a set of trustable links  $T_r$ :

$$\begin{aligned} T_r = \{ & (r_j, c_s, \sigma'_i(r_j, t_k)) | \\ & \exists t_k \in T_i : (r_j, c_s) \in \alpha(R2CT_{i,r_j,t_k}) \\ & \& (r_j, c_s) \in \alpha(R2C) \} \end{aligned} \quad (5)$$

In  $TC_i(r_j, c_s)$  a new similarity  $\sigma_i^*(r_j, c_s)$  computed as:

$$\sigma_i^*(r_j, c_s) = \frac{\sigma_i(r_j, c_s) + \sum_{l \in TC_i(r_j, c_s)} \phi(l)}{1 + |TC_i(r_j, c_s)|} \quad (6)$$

Finally Trumo combine assigned value to each link  $T_r$  as follows:

$$\begin{aligned} \psi_{r_j, c_s}(T_r) = & \left[ \sum_{i=1}^P \lambda_i(r_j, c_s) \sigma_i^*(r_j, c_s) \right] \\ & + \lambda_{P+1}(r_j c_s) \frac{|T_r(r_j, c_s)|}{\max_{n,m} |T_r(r_N, c_M)|} \end{aligned} \quad (7)$$

# An Architecture's Approach

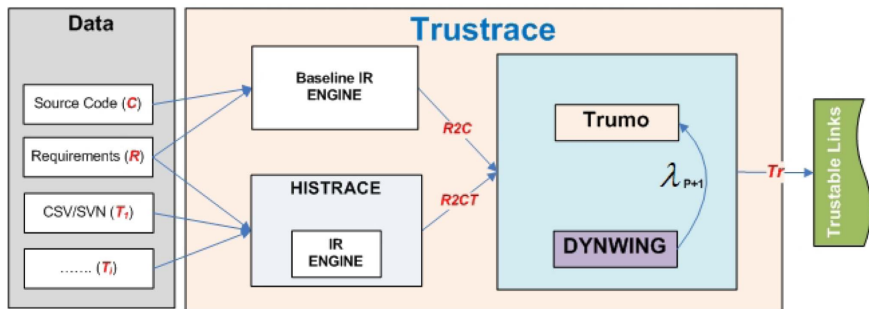


Figure : Trust-based requirement traceability process

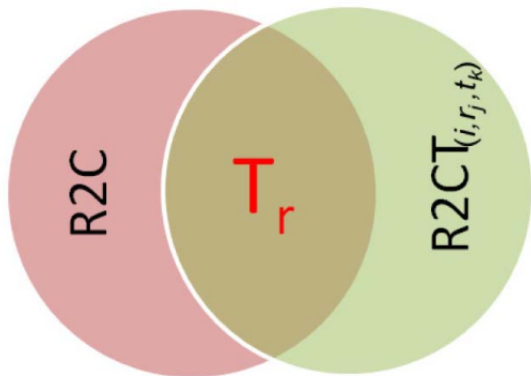


Figure : Overlapping of  $R2C$ ,  $R2CT_{i,r_j,t_k}$  and  $T_r$

# Trustrace Step-by-Step

- **Histrace**: Uses requirements' textual descriptions, CVS/SVN commit messages, bug reports and classes to produce experts.
- **Trumo**: Uses a web model of users' trust to discard and/or rerank the similarity of links in  $T_r$ .
- **DynWing**: Uses Expectation-Maximization approach to choose the right weight per link for different experts.

# Goal

- **Quality Focus:** The accuracy of Trustrace in terms of precision and recall. Also includes improvement by DynWing in terms of  $F_1$  score.
- **Perspective:** The perspective of practitioners interested in recovering traceability links with greater precision and recall values.

# Research Questions

## Accuracy of traceability links:

- **RQ1** recovered by Trustrace compare with JSM and VSM.
- **RQ2** recovered by DynWing compare with PCA.



# Analysis Method

- To answer **RQ1**, we perform several experiments with **different threshold values** on the recovered links to perform statistical tests on the precision and recall values.
- To answer **RQ2**, we use PCA and DynWing to **assign weights** to the traceability links recovered using Trustrace.

- **RQ1:** Trusttrace helps to recover more correct links than IR techniques alone. **When two experts are available, Trusttrace is always better.** In only one case and with just a single expert due to a lack of external source of information, did recall go down.
- **RQ2:** DynWing **provides better weights for different experts than a PCA-based weighting technique.** However, it is possible that in some cases PCA-based weighting provides the same (but not better) results as DynWing.

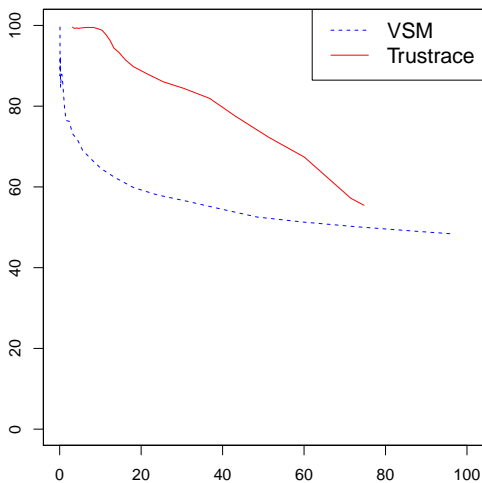


Figure : Precision and recall values of JSM & Trustrace, *Rhino* example

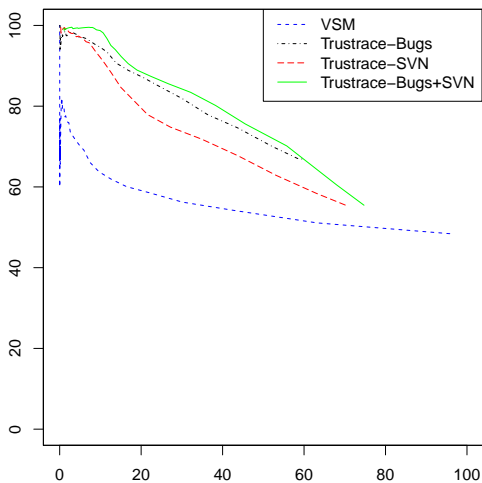


Figure : Precision and recall values of VSM & Trustrace, *Rhino* example

- Data Set Quality Analysis
- DynWing versus MSW versus PCA
- Number of Experts
- Other Observations
- Practical Applicability Trustrace
- Revisiting the Conjectures
- Threats to Validity

# Threats to Validity

- **Construct Validity:** Quantify the degree of inaccuracy by validation of the precision and recall using **manually built oracles**.
- **Internal Validity:** Mitigate this threat by using **MSW- and PCA-generated  $\lambda$  values**, and by using **the same setting** for all the experiments.
- **External Validity:** The research approach is **applicable to any other systems**.
- **Conclusion Validity:** Mitigate this threat by the appropriate nonparametric test **Mann-Whitney**. And applying **the Shapiro-Wilk** test to select data.

- *Histrace*: Implement more instances, using emails and forum discussions.
- *Trumo*: Use in other software engineering fields, in particular, test-case prioritization, anti-pattern detection and concept location.
- *Trustrace*: Deploy in a development environment. Perform experiments with real developers.
- *Regular Expression*: Use advanced matching techniques.



Figure : Year 2012 H1B Applicants





Figure : Year 2012 United States Base Salary, per Profession



Figure : No Question Off Limits!