

Automated Traceability using IRSim: A Preliminary Empirical Study

Dharma Ganesan

Mikael Lindvall

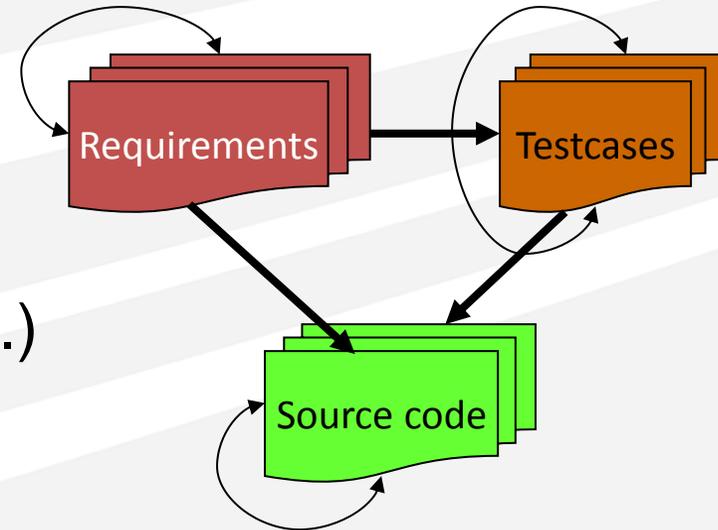
Jan-Philip Quirnbach

Outline

- Problem and Motivation
- IRSim Tool
- Results of Preliminary Empirical Study

Problem

- Many NASA projects require traceability
- Problem: It is tedious to create and maintain traceability
- Traceability between
 - requirement and source code
 - requirement and test cases
 - other docs(bug reports, user manuals, etc.)



Key motivations for retrieving traceability links

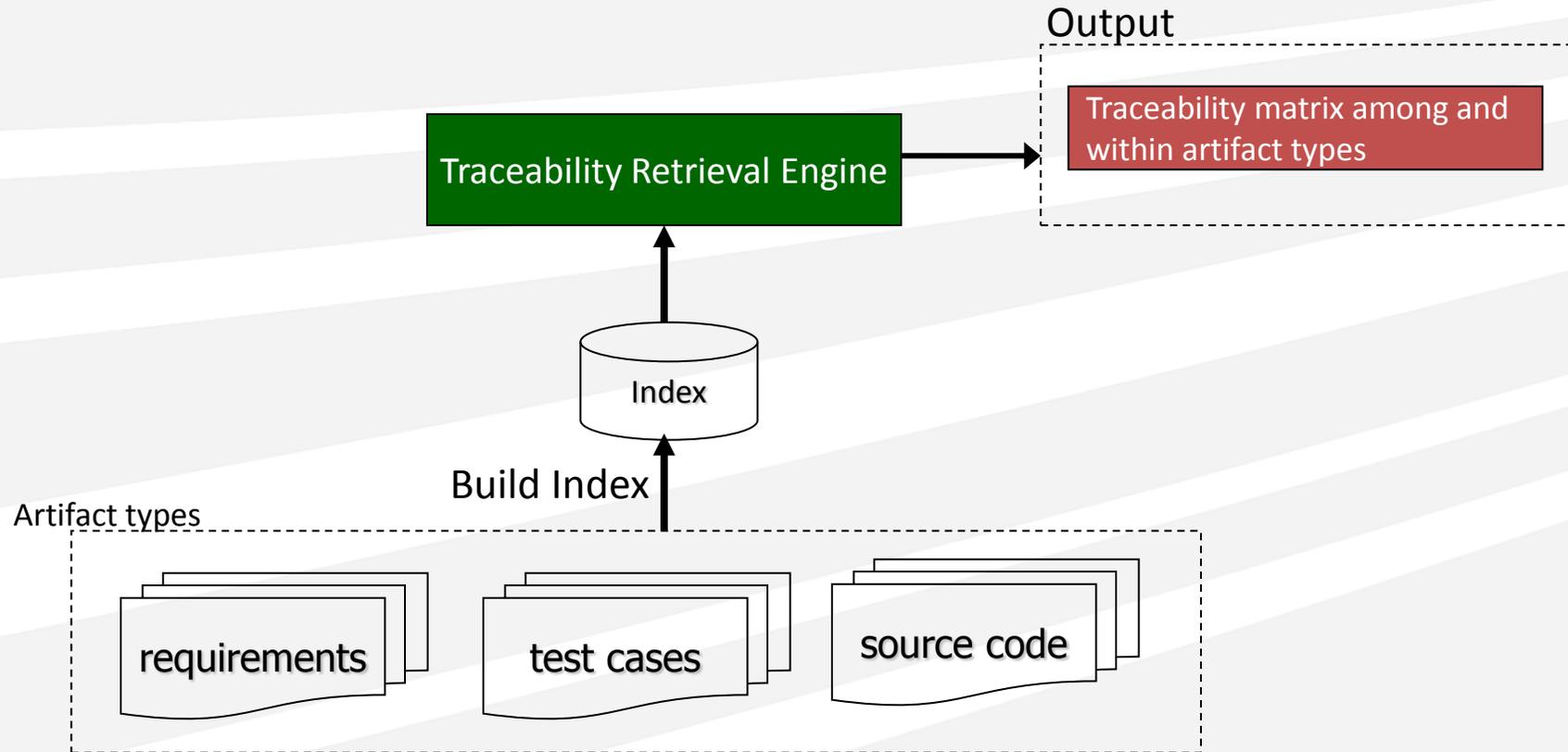
- **Verification and Validation**

- To check completeness of an implementation w.r.t stated requirements
- To verify whether requirements are correctly implemented, the corresponding source code should be identified

- **Program Understanding**

- To understand how a requirement is realized, the corresponding source code should be identified
- Bug fixes or feature enhancements require tracing from requirements/testcases to code

Retrieving traceability links



IRSim Tool

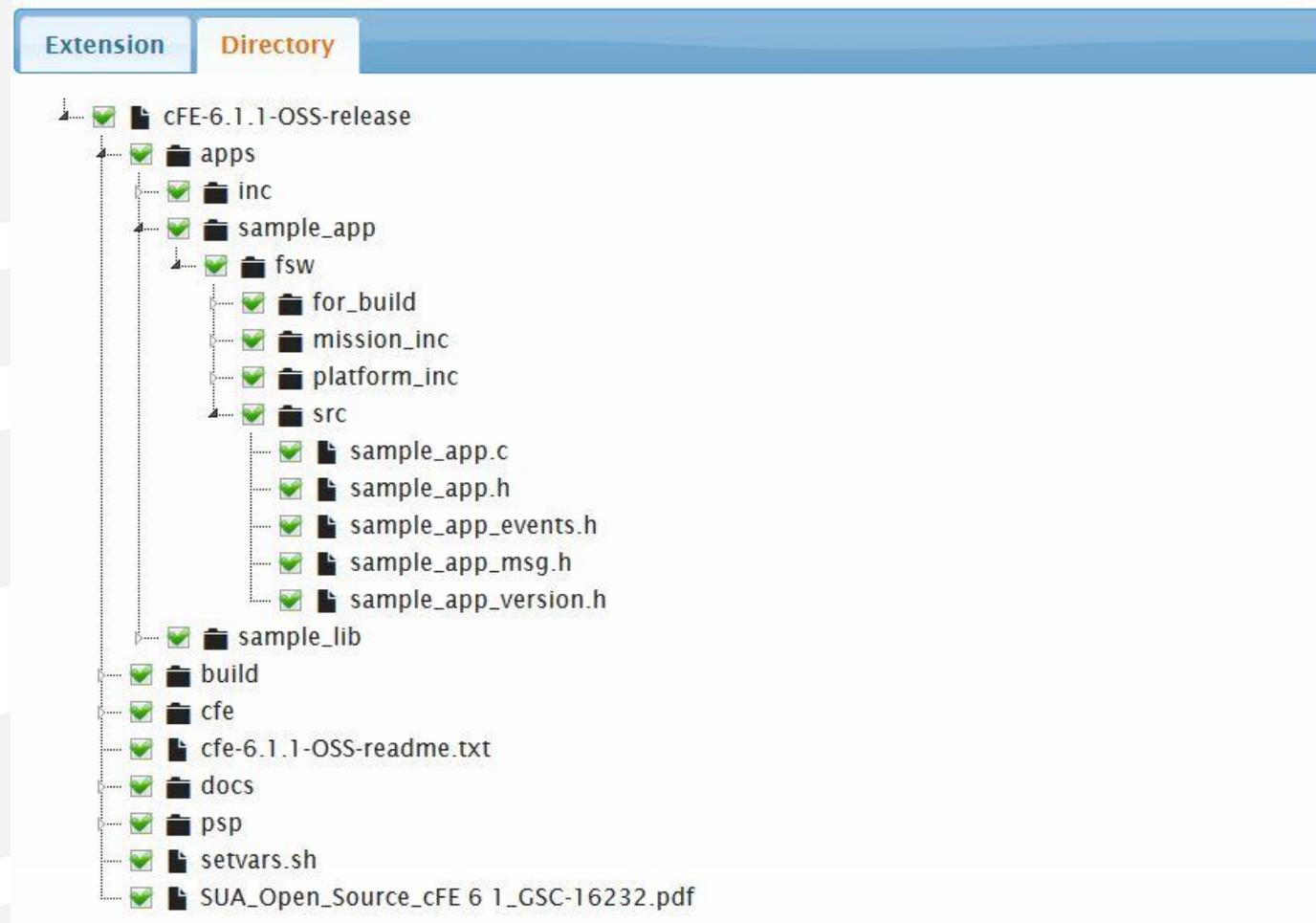
IRSim Tool

- Web application
- Upload archives
 - User can select files for indexing:
 - by extensions
 - by directories
- Automatically produce traceability matrix
- Based on Information Retrieval Methods
 - Vector Space Model and Lucene

Explore Uploads by Extension

Extension		Directory		
Extension ↓↑	Occurrences ↓↑	Enable(2642/2642)	Example File	
 html	531	<input checked="" type="checkbox"/>		
 dat	263	<input checked="" type="checkbox"/>		
 c	154	<input checked="" type="checkbox"/>		
 log	145	<input checked="" type="checkbox"/>		
 doc	142	<input checked="" type="checkbox"/>		
 loge	126	<input checked="" type="checkbox"/>		
 logf	126	<input checked="" type="checkbox"/>		
 logp	126	<input checked="" type="checkbox"/>		
 logr	126	<input checked="" type="checkbox"/>		
 h	118	<input checked="" type="checkbox"/>		
 prc	111	<input checked="" type="checkbox"/>		
 zip	111	<input checked="" type="checkbox"/>		

Explore Uploads by Directories



Text Parsing Methods in IRSim

- CamelCase: splitting camel cased words
 - i.e. CamelCase = camel, case, camelcase
- Standard: Using a List of English stop words
 - i.e. and, but, if, not, or, the,...
- Simple: Using a Lower Case Tokenizer
 - i.e. Hello World = hello, world

Matrices, Vector Spaces, and Traceability Retrieval

- In 1960's, Gerard Salton proposed idea of modeling document collection as a matrix – Vector Space Model

<i>Terms</i>	<i>Documents</i>					
	D_1	D_2	D_m
W_1	C_1^1	C_2^1	...	C_j^1	...	C_m^1
W_2	C_1^2	C_2^2	...	C_j^2	...	C_m^2
\vdots						
W_n	C_1^n	C_2^n	...	C_j^n	...	C_m^n

$$= \mathbf{C}$$

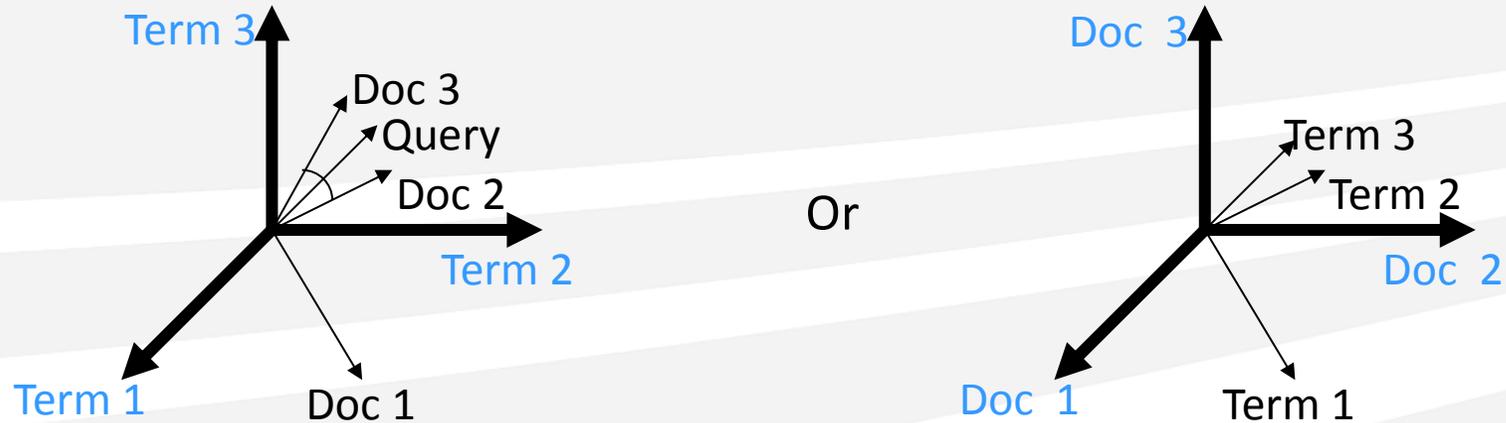
Typical term-document incidence matrix \mathbf{C} ($C_j^i = n \leftrightarrow$ document D_j contains term W_i exactly n times)

- IRSim tool builds the matrix automatically
- Similarity between documents: cosine of angle between them

Weight options for the Matrix

- Boolean: 1 term present, 0 term not present
- TF: The frequency of the term in the document
- TF-IDF: important of a word to a document in a collection

Geometry and Traceability Retrieval



- If two vectors are nearly orthogonal (90 degree) then they are treated conceptually dissimilar
- If the angle between two vectors are towards 0 degree then they are considered conceptually similar
- User defines a threshold on the angle to select similar vectors of a vector

Vector Space Model Boolean Example

- Document to a vector with terms (Boolean)

$$\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \\ t_4 \end{array} \begin{pmatrix} d_1 & d_2 & d_3 & d_4 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Calculate cosine of angle for similarity

$$\cos \theta = \frac{d_1 \cdot d_4}{\|d_1\| \cdot \|d_4\|} = \frac{1 \cdot 1 + 0 \cdot 0 + 1 \cdot 1 + 0 \cdot 1}{\sqrt{2} \cdot \sqrt{3}} = 0.8164$$

Example of generated traceability links

Show entries Showing 1 to 10 of 500 entries

[<](#) [>](#)

	Score ↑↓	Document 1 ↑↓	Document 2 ↑↓	Tokens ↑↓	Tokens ↑↓
	0.1866	\abc\requirement\cSB4301.txt	\abc\source\cfe_sb_api \CFE_SB_CreatePipe.txt	14	181
	0.18552	\abc\requirement\cSB4311.txt	\abc\source\cfe_sb_api \CFE_SB_Subscribe.txt	25	37
	0.18399	\abc\requirement\cSB4303.1.txt	\abc\source\cfe_sb_task \CFE_SB_SetSubscriptionReporting.txt	14	27
	0.18399	\abc\requirement\cSB4311.1.txt	\abc\source\cfe_sb_task \CFE_SB_SetSubscriptionReporting.txt	14	27

These “raw” traceability links can be used to generate a traditional traceability matrix on any level

Questions for the Study

- Which is the *best* weight type for the term-doc matrix?
- Which is the *best* text parsing strategy?

Results of a preliminary empirical study using NASA's Core Flight Software

NASA's Core Flight Software

- Core Flight Executive
- Used as basis for satellite data system and instruments and for embedded systems
- Written in C with a software library called Operating System Abstraction Layer (OSAL)

Requirements

Module	# of Requirements
Execution Service	148
Software Bus	34
Time Service	38
Table Service	52
Total	272

ReqID	Requirement Text	Rationale
	Subsection Title = 3.4.1 Operational Interface	
cSB4000	Upon receipt of a NOOP command, the cFE shall increment the command counter.	Useful for verifying communication

Source Code

Module	LOC
Execution Service	2478
Software Bus	589
Time Service	574
Table Service	1344
Total	4985

Golden Model

- Manually traced Requirements to the Source Code (.h,.c,.mak)
- Used as basis for the Traceability (Compare with IRSim)
- Goal is to evaluate the *precision* of IRSim
- Fraunhofer has analyzed the code for several years
 - Golden model was validated

Golden Model

- ID, Requirement, File, Function / Name

	A	B	C	D	E
1	ID	Requirement	File		Function / Name
2	cSB4000	Upon receipt of a NOOP command, the cFE shall increment the command counter.	cfe_sb_task.c		void CFE_SB_ProcessCmdPipePkt(void)
3	cSB4001	Upon receipt of Command the cFE shall set to zero the following counters in housekeeping telemetry: - Valid command counter - Invalid command counter - No subscriptions counter - Message send error counter - Message receive error counter - Create Pipe error counter - Subscribe error counter - Pipe Overflow error counter - MsdID-to-pipe limit error counter	cfe_sb_task.c		void CFE_SB_ResetCounters(void)

Evaluation procedure

- Looked at the Top 5 Results of IRSim
- If the expected functions are present, then it is considered as success
- Otherwise as failure

Precision

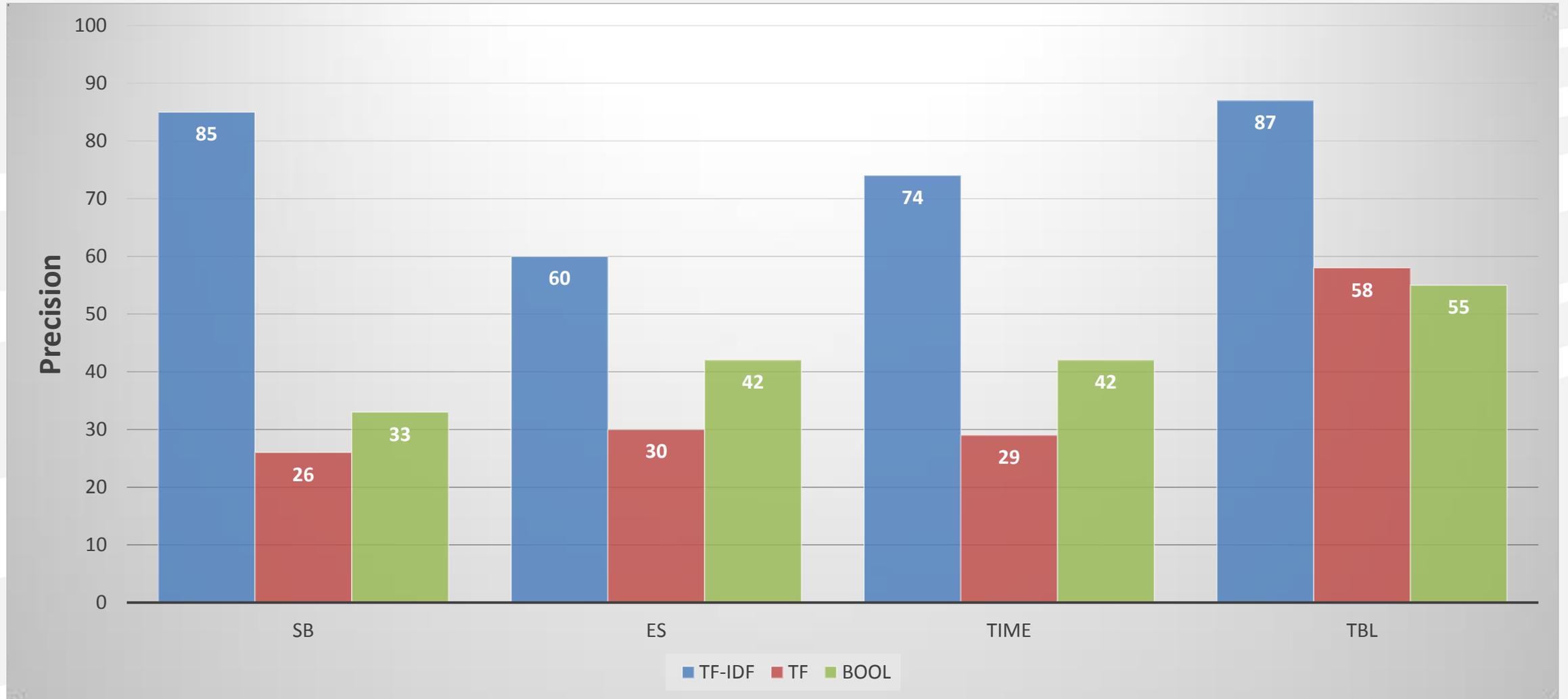
- **Precision** is the ratio of the Top 5 traced requirements with the IRSim Tool to the total manually traced requirements. It is expressed as a percentage.

- $$\textit{Precision} = \frac{\textit{\# of Results in Top 5}}{\textit{\# of Requirements}}$$

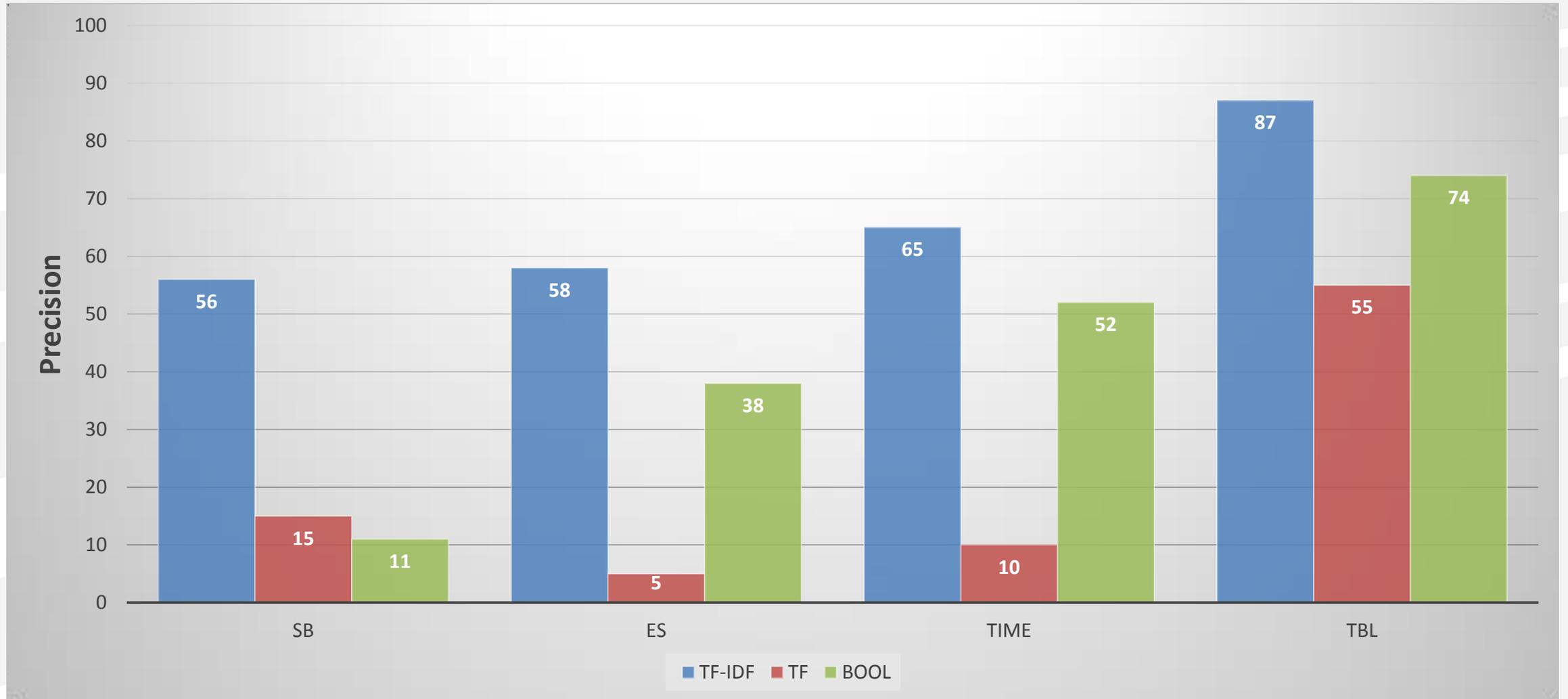
- Example: Requirements: 4, Results in Top 5: 3

- $$\textit{Precision} = \frac{3}{4} = 0.75 = 75\%$$

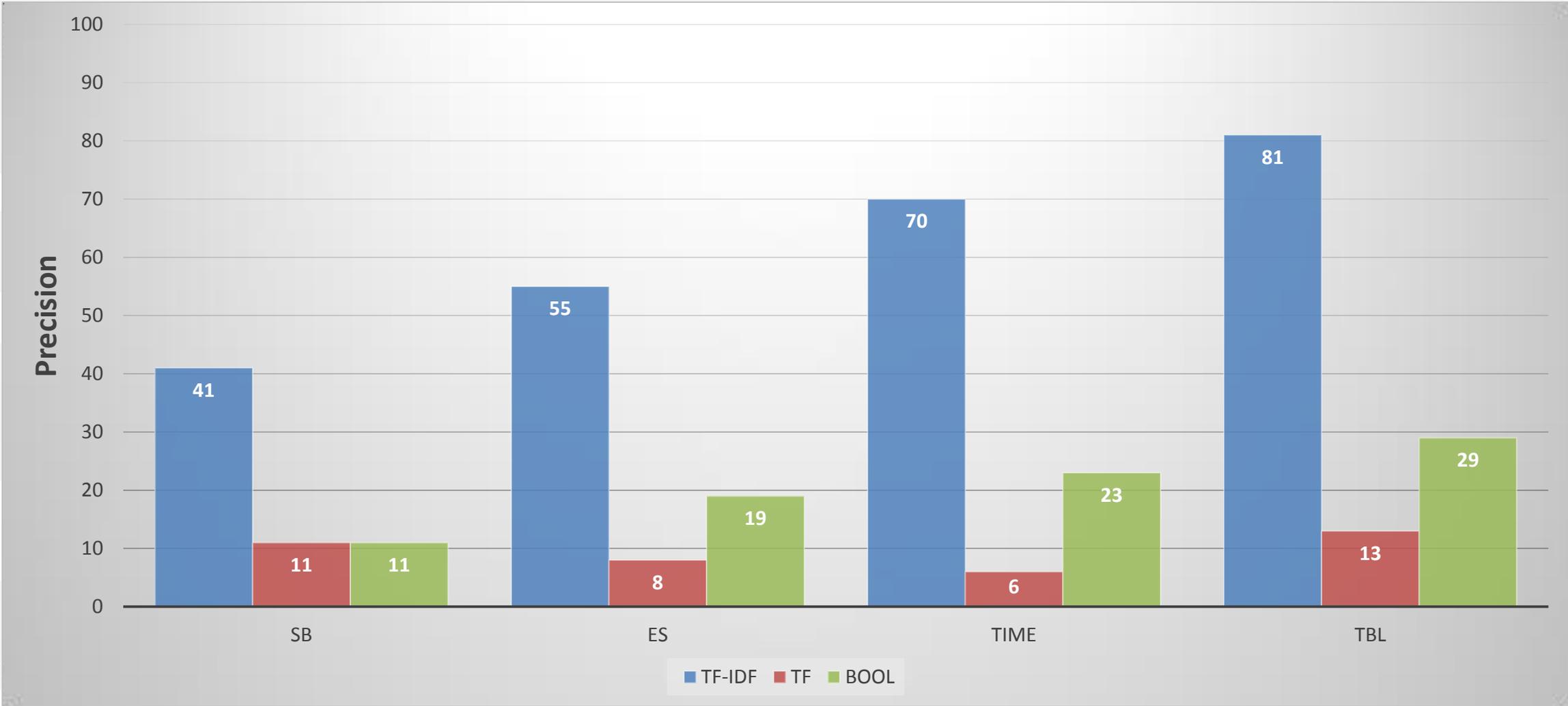
CamelCase Analyzer



Standard Analyzer



Simple Analyzer



Why IRSim is not tracing all requirements?

- Typo in Requirement or Source Code
- Requirement is not well enough worded
- Chopping of Sub-Requirements
 - Parent-child requirement hierarchy
- Heavy use of acronyms in the source code

Summary

- IRSim can be used to generate Traceability links
- Depends on the wording of the Requirement
- Combination of CamelCase and TF-IDF is the best one
- IRSim is completely independent of programming language and can be used on any text
- Need to replicate on other systems
- Need further validation

Acknowledgements?

- NASA CFS/CFE team
- Fraunhofer Interns

Thank you for your Attention

dganesan@fc-md.umd.edu