Can script based languages, like DOORS® DXL, hack Natural Language Processing?

- Webinar rules:
  - The Webinar will start in few minutes
  - You'll be muted throughout the Webinar
  - There's a chat box for you to ask questions at any time during the webinar
  - Please address comments and questions to the user "The REUSE Company" and not to the presenter directly
  - If you have any technical issues please use this chat box, or mail us at: support@reusecompany.com
  - The Webinar will be recorded. A link to the recording will be sent to you in few days time

TRC

WEBINARS 2018

Can script based languages, like DOORS® DXL, hack Natural Language Processing?

Monday, 23 April 2018

Presenters' profile

Dr. SimonWright

Chief Systems Engineer

Simon Wright
simon.wright@reusecompany.com

Content

- Description of the Reuse Company
- Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Content

- Description of the Reuse Company
- Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Brief description of The Reuse Company

the REUSE company

T(he) R(euse) Q(ompany)

Trace + Retrieval + Quality (Reuse)

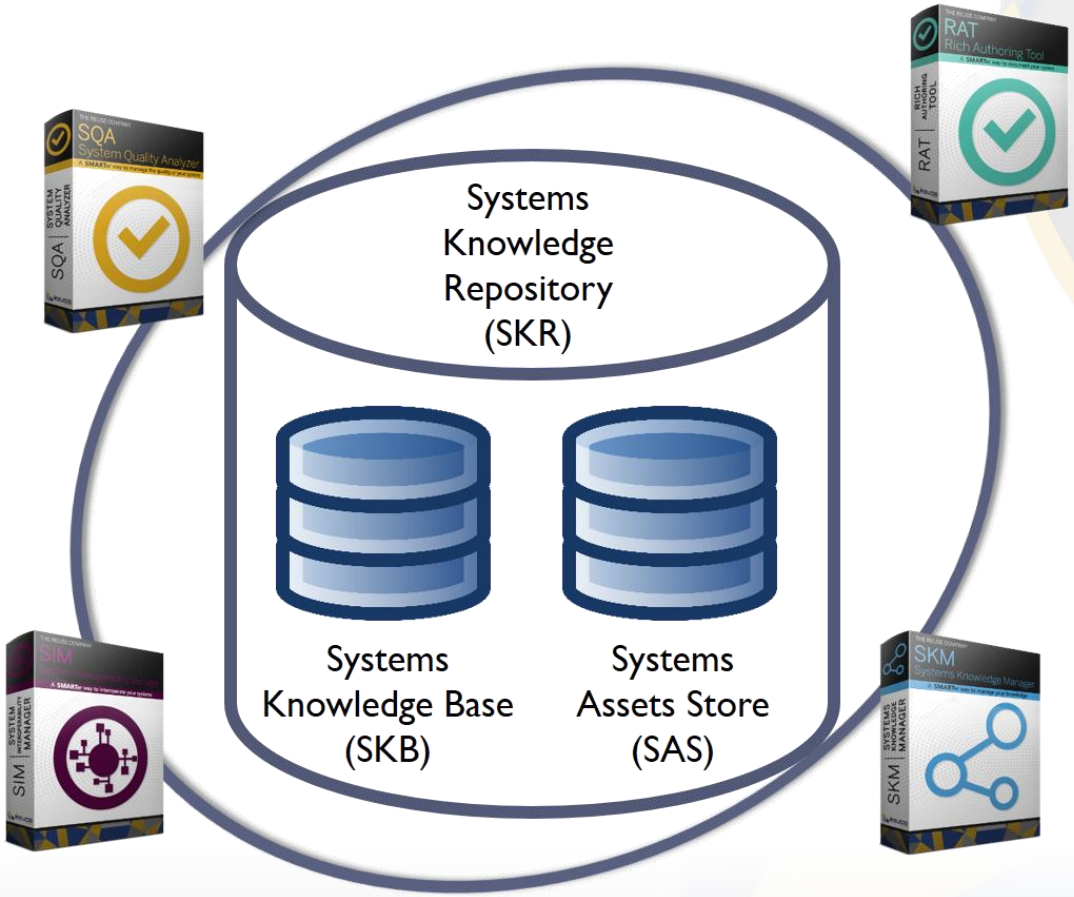
Aiming to **Improve Project performance**

By means of a: **Knowledge Centric Approach**

TRC Suite v18

SQA -System Quality Analyzer

Global Quality Management



RAT –Rich Authoring Tool

Smart text authoring

SIM –System Interoperability Manager

Tailorable Interoperability Platform

- R+ Manager

Managing requirements transformations

Managing models transformations

- T+ Manager

Managing traceability

- Reasoning Manager

Task based environment

SKM –System Knowledge Manager

Management of System Knowledge

Libraries

The REUSE Company in the World

Selected set of Customers

Aerospace and Defense

AIRBUS
AIRBUS DEFENCE & SPACE
THALES
SAFRAN Snecma
AIRBUS HELICOPTERS
EADS INNOVATION WORKS
MBDA MISSILE SYSTEMS
SAFRAN Sagem

Automotive

RENAULT

Energy

FUSION FOR ENERGY
iter the way to new energy

Consulting

indra
SHERPA ENGINEERING

Banking

BCR Banco de Costa Rica
Santander

Health care

THINK SURGICAL
Health Net

Other industries

sage
SIEMENS
tirant editorial

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Dr. Simon Wright

- For 10 years I earned a living as a Rational® DOORS® consultant and DXL programmer, (and from time to time I still do!)
  - Trusted Borders – Metrics reporting
  - SAAB Avitronics, Sweden – SACREM
  - Russian Atomic Energy Authority, Moscow – TOI
  - Novo Nordisk, Denmark – Document/Module Templates
  - Syntell Stockholm – RQPlus contributor
  - Norewgian Army – DEx Manager
- DOORS is a mature tool for managing requirements – I have my own license, this year's Maintenance and Support cost me £1000 – I am committed to the tool but I will not be biased for or against DXL.

Simon Wright
simon.wright@reusecompany.com

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Requirements Quality Papers

- Ivy Hooks - "Writing Good Requirements" Proceeding's of the Third International Symposium of the INCOSE Volume 2 1993
- P. Kar and M. Bailey "Characteristics of Good Requirements" Presented at the 1996 INCOSE Symposium.
- William M. Wilson – "Writing Effective Natural Language Requirements Specifications" The Journal of Defense Software Engineering February 1999
- Karl E. Wiegers "Writing Quality Requirements" published in Software Development, May 1999.

Early Requirements Quality Tools

1. ARM – NASA - proto-type 1997 – ASCI Text analysis.
   W. M. Wilson, L. H. Rosenberg and L. E. Hyatt. "Automated analysis of requirement specifications." Proceedings of the 19th International Conference on Software Engineering (ICSE), pages 161-171, 1997

2. Requirements Councillor – EU IT project "Precepts" available 1999 - WORD Document plug-in Text Analysis
   Dr. Simon Wright – "How Good Are Your Requirements" The 1999 RTM User Workshop
3. Boilerplates – DOORS Plugin – Jeremy Dick 1999 – Linguistic patterns
   "Using statement-level templates to improve the quality of requirements" An Integrate white paper October 2012.

DOORS and Req's Quality

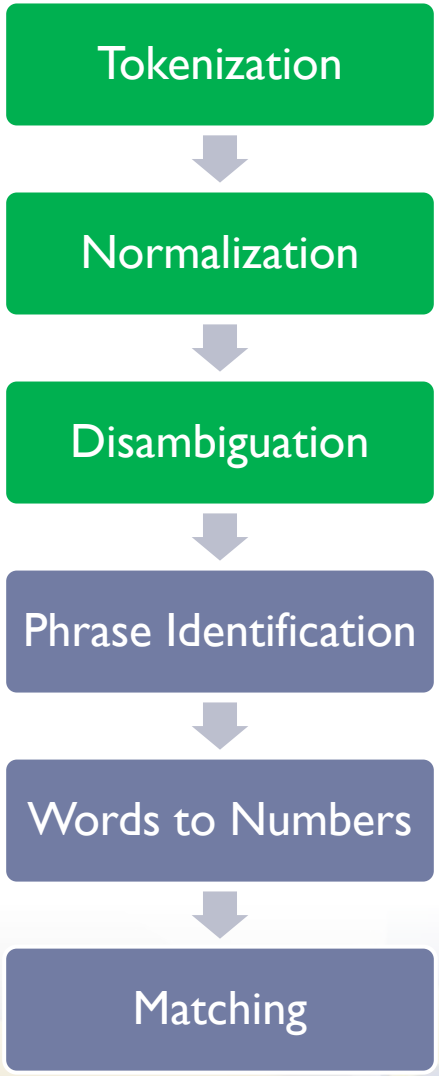
- arm.dxl
  - © Ian Alexander, 30 October 2006
  - "based loosely on NASA's ARM"
  - Text analysis
- smartQualityAnalysis.dxl
  - © Tony Goodman 22-MAR-2008
  - "extends on original work done at NASA"
  - Text analysis.

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Methods for NLP – just syntax

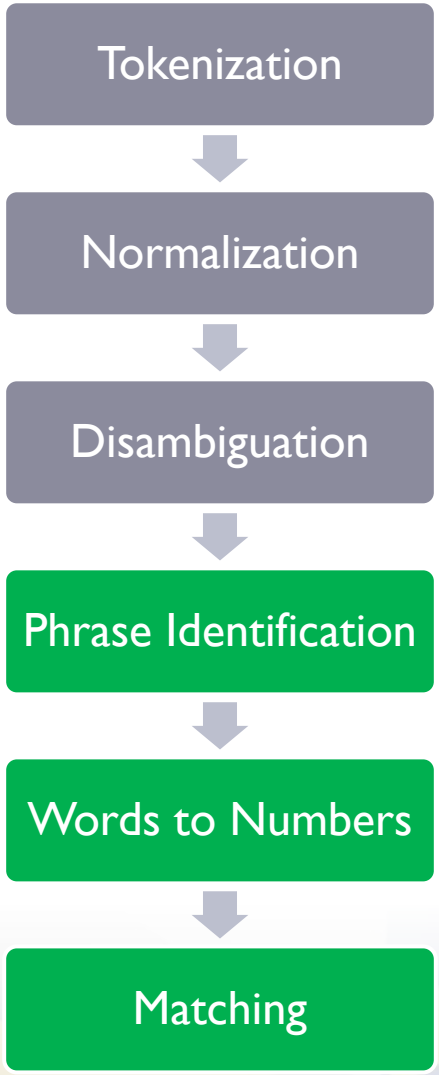
- Tokenization – chopping a sentence into pieces. How to tokenize "can't" or "L.E.D."?
- Stemming - reducing words to their word stem, base or root form "cats" "catty" "catlike" all stem from CAT – what is the stem of "better", (lemmatization).
- Part-of-speech tagging - determining the **parts** of speech for each word "Book a book from the book library"
- Disambiguation – from candidates term tags to only **one** term tag for each element



- Input:
  "The flight system shall have three engines"
- Output:
  list of simple terms with possible term tags
  - [The| DETERMINER],
  - [Flight| NOUN],
  - [System | NOUN, **PROPER NOUN**],
  - [Shall| VERB, **MODAL VERB**],
  - [Have| VERB],
  - [Three| NUMBER],
  - [Engine| NOUN]

Methods for NLP – just syntax

- Phrase Identification
  - combining words into phrases
  - 'Flight System' is a single term
- Words to Numbers
  - three to 3
- Matching Process
  - list of terms grouped by recognized patterns
  - Subject
  - Verb / Verbal phrase
  - Object
  - Predicate



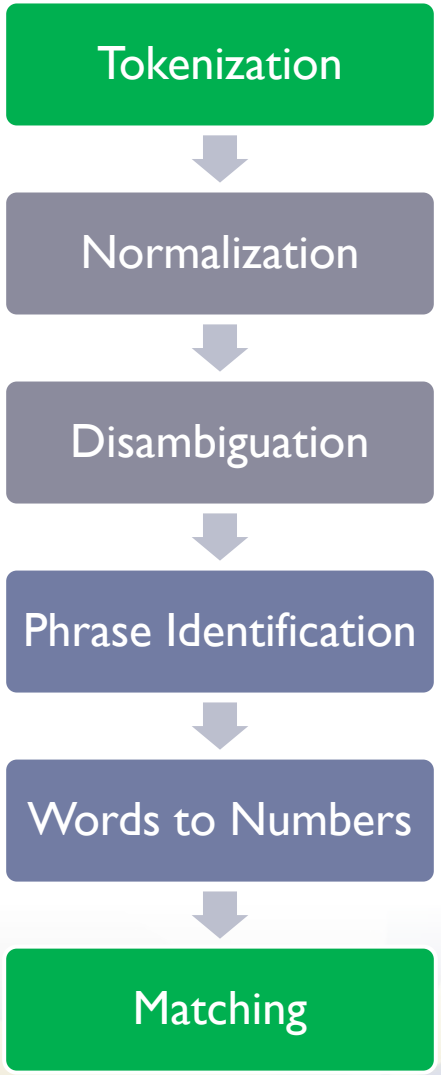
- Input:
  - [The| DETERMINER],
  - [Flight| NOUN],
  - [System | NOUN, PROPER NOUN],
  - [Shall| VERB, MODAL VERB],
  - [Have| VERB],
  - [Three| NUMBER],
  - [Engine| NOUN]
- Output:
  - [SUBJECT]
    - [The | DETERMINER]
    - [Flight System | PROPER NOUN]
      - [Flight| PROPER NOUN, NOUN],
      - [System| NOUN]
  - [VERB]
    - [Shall | VERB, MODAL VERB]
    - [Have | VERB]
  - [PREDICATE]
    - [3 | NUMBER]
    - [Engine | NOUN.

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Methods for Text Analysis

- Tokenization – chopping a sentence into pieces.
- Matching Process – found against stored
- Problems
  - "cats" "catty" "catlike" all the same
  - Book a book from the book library
  - Flight System – one phrase
  - Forty three to 43
  - Match 1000 requirement words checked against 100 words = one hundred thousand compares!.



- Input:
  "The flight system shall have three engines"
- Output:
  - {the,
    flight,
    system,
    shall,
    have,
    three,
    engines}

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Functions for textual Analysis

String functions

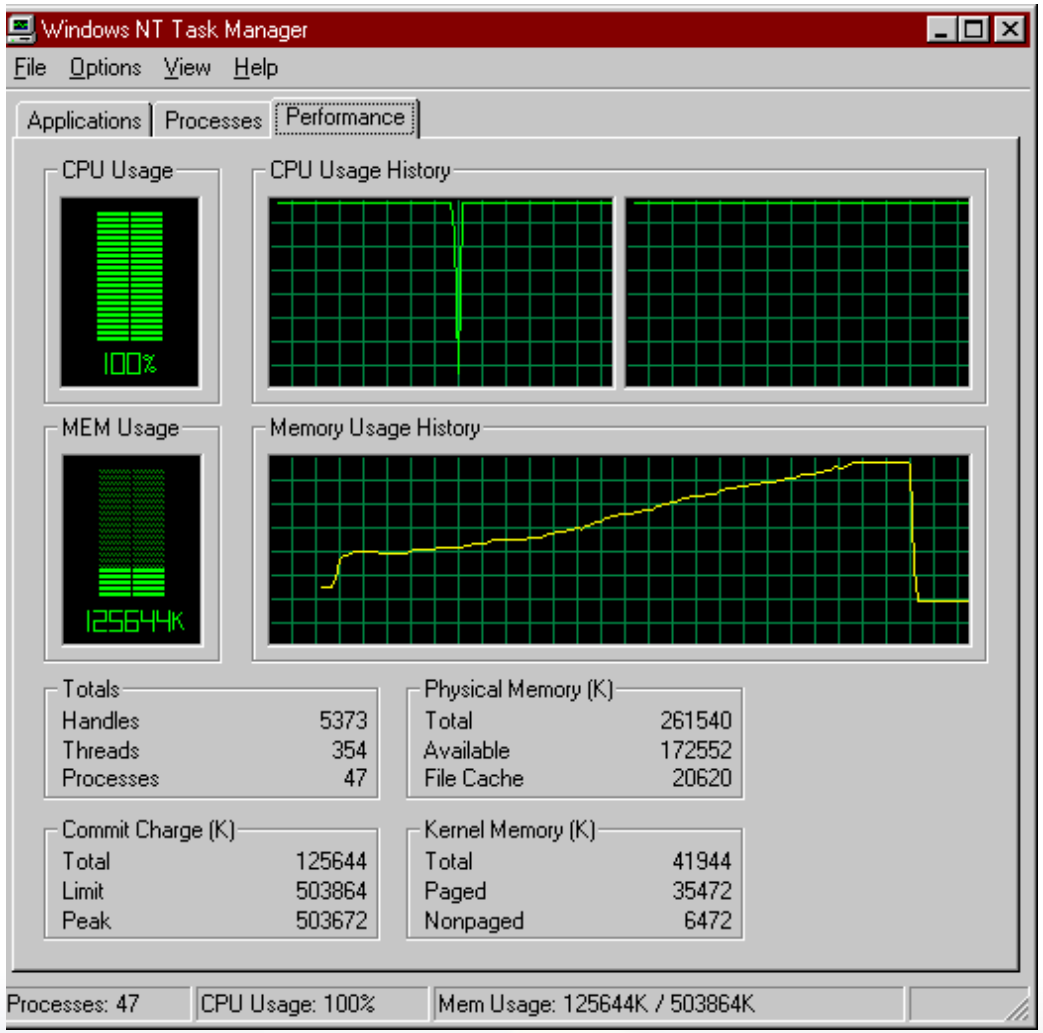
- Declaration – a String is an array of Characters
- Tokenize
  - Find the position of a delimiter Char in a String – 'space', full stop, parenthesis ….
  - Extract a sequence of Chars from a string
  - Regular expressions
- Match
  - Boolean Comparison / Contains
  - Regular expressions

String Data Structure functions

- Declaration – an array of strings, linked lists, skip lists, buffer
- Tokenize
  - Add a string to array/list/buffer
  - Remove a string from array/list/buffer
  - Searching – avoid duplication
- Match
  - Sorting for searching
  - Searching – find a match

DXL provides all these functions – as well as many other languages.

Some Issues for DXL Textual Analysis



Mitigation

- To avoid this issue use variable types, (Skip Lists, Array, Buffer), that have functions that explicitly de-allocate memory used by these variables
- But….

You can find an excellent dissertation on this by Mathias Mamsch in 2012 here:
https://www.ibm.com/developerworks/community/forums/html/topic?id=77777777-0000-0000-0000-000014886977&ps=25

Some Issues for DXL Textual Analysis

String Memory Management

- Tony Goodman points out -

"Memory de-allocation is not automatic for the dynamic types Skip, Array, Buffer, DB, OleAutoArgs, IPC and Stat

Repeated use of these types can consume memory and reduce performance of DOORS

In some cases this can even lead to DOORS crashing"

Mitigation

- To avoid such memory leaks you must explicitly de-allocate memory used by these variables. This means that you should have a delete() or destroy() function call for every create() function call in your program
- Alternatively pass data by reference rather than by value …

Some Issues for DXL Textual Analysis

String Memory Management

Referencing is supported by DXL. It allows a convenient way of limiting data replication in passing data values between functions

With Referencing you can totally get rid of this memory leak. The accessed Buffer or Skip list, is truly handled as that same Buffer or Skip

Thus, the memory area will be emptied, when that Buffer or Skip is emptied or deleted

Mitigation

- Spend many hours and thousands of Euros/ Dollars/ Pounds/ SEKS/ NOKS etc and reprogram your DXL to use buffers and skip lists and pass by reference not value
- BUT! There is another problem.

Some Issues for DXL Textual Analysis

Hidden formatting or different character set

- Below is a rich text string that contains the word "hello"
- {\rtf1 \deff0{\fonttbl {\f1 Times New Roman;}}{\stylesheet {\s1 Normal;}}{\s1 hello\par}}
- String1 == String2 may result in False. Depending which character sets you are using, or rich text format, or embedded hidden rich text

Mitigation

- Convert the rich text to ASCII for proper comparison…
- BUT this will add to the "string table"!
- And the cycle starts again, memory is consumed, leaks occur.

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

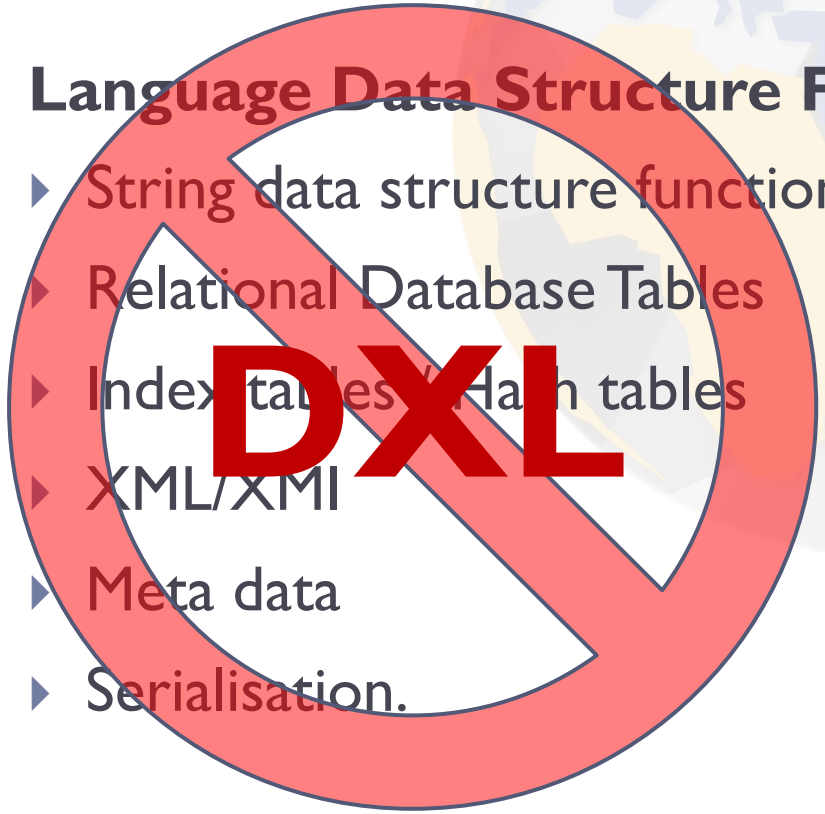
Functions for NLP Syntax Analysis

Language Functions

- String functions plus
- entity extraction, *
- sentiment analysis, *
- keyword extraction, *
- concept tagging, *
- relation extraction, *
- taxonomy classification, *
- language detection, *
- microformats parsing, *
- linked data support *

Language Data Structure Functions

- String data structure functions plus
- Relational Database Tables
- Index tables / Hash tables
- XML/XMI
- Meta data
- Serialisation.



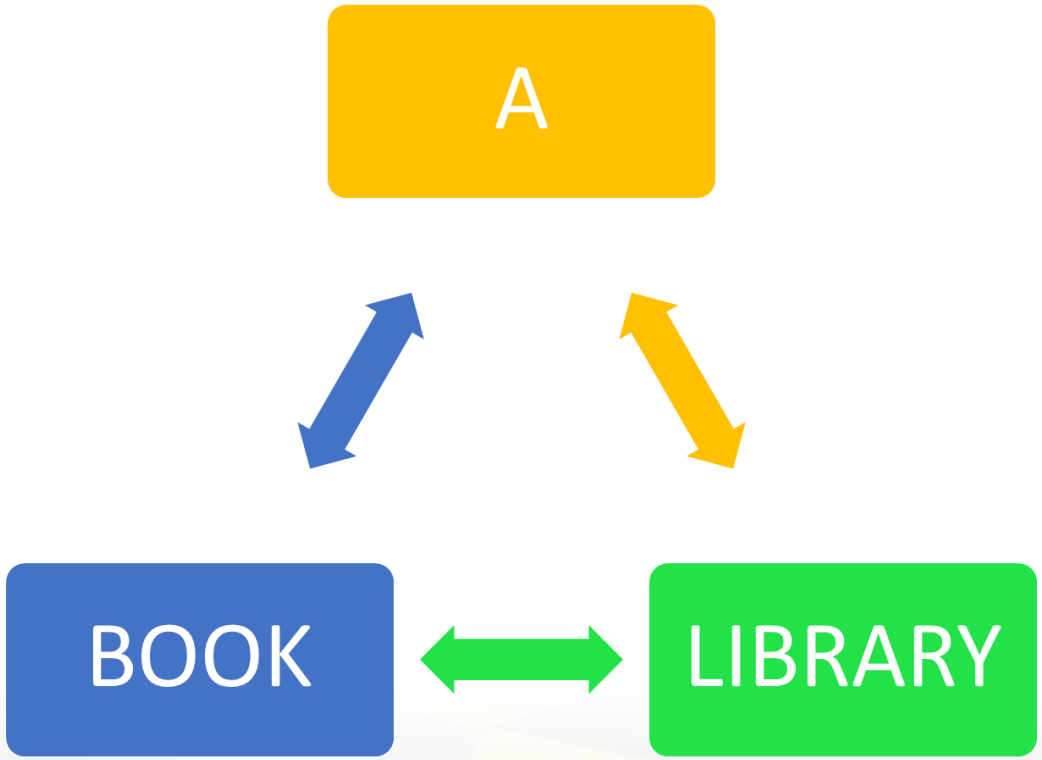
* Offered by AlchemyAPI – an NLP API service

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Natural Language Processing: Correctness and Consistency

- The three words 'book', 'library' and 'a' can be ordered in six ways
- Only three are syntactically correct



- "A library book"
- "A book library"
- "Book a library"

An object

A descriptor

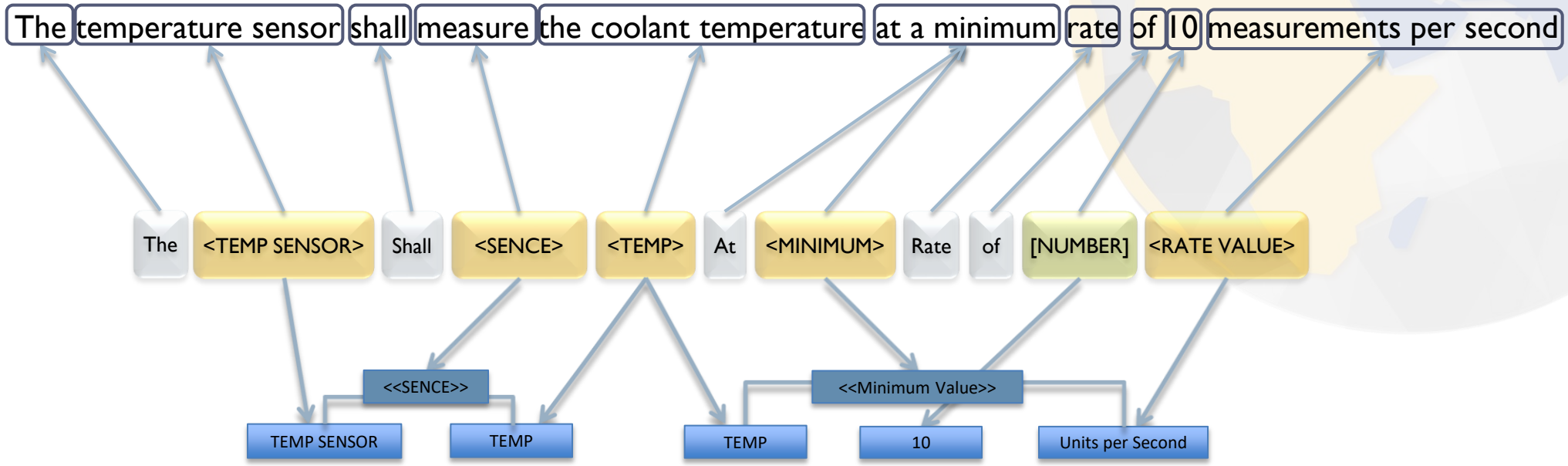
An action.

Natural Language Processing: Correctness and Consistency

Book the book from the book library

Phrase	Correct	Consistent	Total
"A library book"	0	0	0
"A book library"	0	1	1
"Book a library"	0	1	1
"Library a book"	1	0	1
"Library book a"	1	0	1
"book Library a"	1	1	2
Book the book from the book library	0	2	2
Reserve the book from the publications library.	0	0	0

Pattern with semantics



Classified TERMS allow meaning, (semantic), to be checked.

The relationships between the terms
are best handled by using a **Relational Database**.

Model Sentences

William M. Wilson – "Writing Effective Natural Language Requirements Specifications" The Journal of Defense Software Engineering February 1999

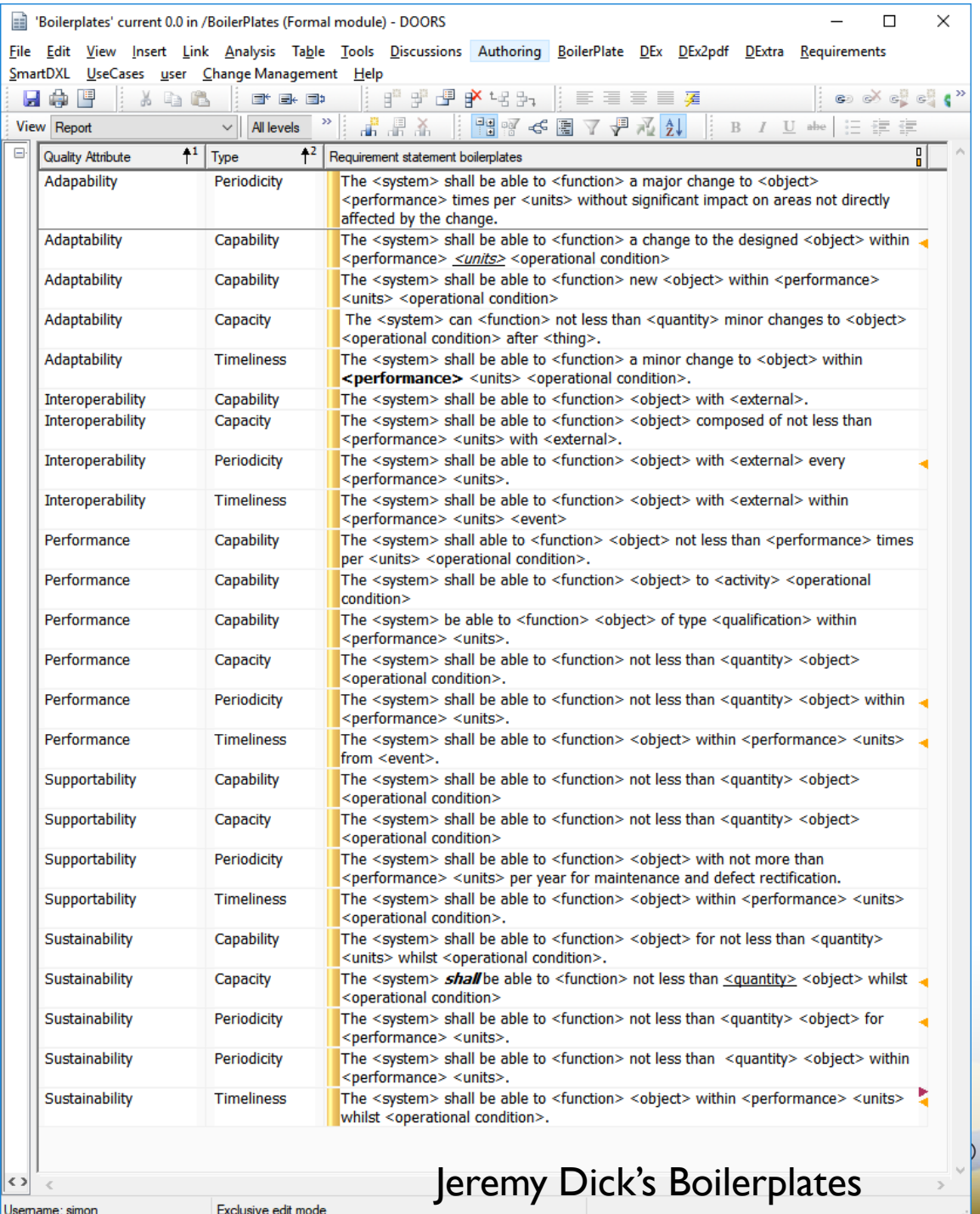
Each specification statement consists of four basic structural elements—entities, actions, events, and conditions. These elements can be used or modified by various cases such as the following:

- Owner.
- Actor.
- Target.
- Constraint.
- Owned.
- Action.
- Object.
- Localization.

The recommended model for a specification statement's structure is as follows:

- Localization.
- Actor/Owner.
- Action.
- Target/Owner.
- Constraint.

For example, "When three or more star trackers lose reference stars, the spacecraft shall immediately align its main axis on the earth-sun line unless the optical instrument's cover is not closed."



Jeremy Dick's Boilerplates

Boilerplates and EARS

EARS (Easy Approach to Requirements Syntax) - RE09, IEEE, August 2009, Alistair Mavin et al Rolls-Royce PLC

4.1 Generic requirements syntax

<optional preconditions> <optional trigger> the <system name> shall <system response>

4.2 Ubiquitous requirements

The <system name> shall <system response>

4.3 Event-driven requirements

WHEN <optional preconditions> <trigger> the <system name> shall <system response>

4.4 Unwanted behaviours

IF <optional preconditions> <trigger>, THEN the <system name> shall <system response>

4.5 State-driven requirements

WHILE <in a specific state> the <system name> shall <system response>

4.6 Optional features

WHERE <feature is included> the <system name> shall <system response>

4.7 Complex requirement syntax

For requirements with complex conditional clauses, combinations of the keywords When, While and Where may be required.

Standards

INTERNATIONAL STANDARD

ISO/IEC/ IEEE 29148

First edition
2011-12-01

Systems and software engineering

Life cycle processes

Requirements engineering

[Condition] [Subject] [Action] [Object] [Constraint]

EXAMPLE: When signal x is received **[Condition]**, the system **[Subject]** shall set **[Action]** the signal x received bit **[Object]** within 2 seconds **[Constraint].**

Or

[Condition] [Action or Constraint] [Value]

EXAMPLE: At sea state 1 **[Condition]**, the Radar System shall detect targets at ranges out to **[Action or Constraint]** 100 nautical miles **[Value]**.

Or

[Subject] [Action] [Value]

EXAMPLE: The Invoice System **[Subject]**, shall display pending customer invoices **[Action]** in ascending order **[Value]** in which invoices are to be paid.

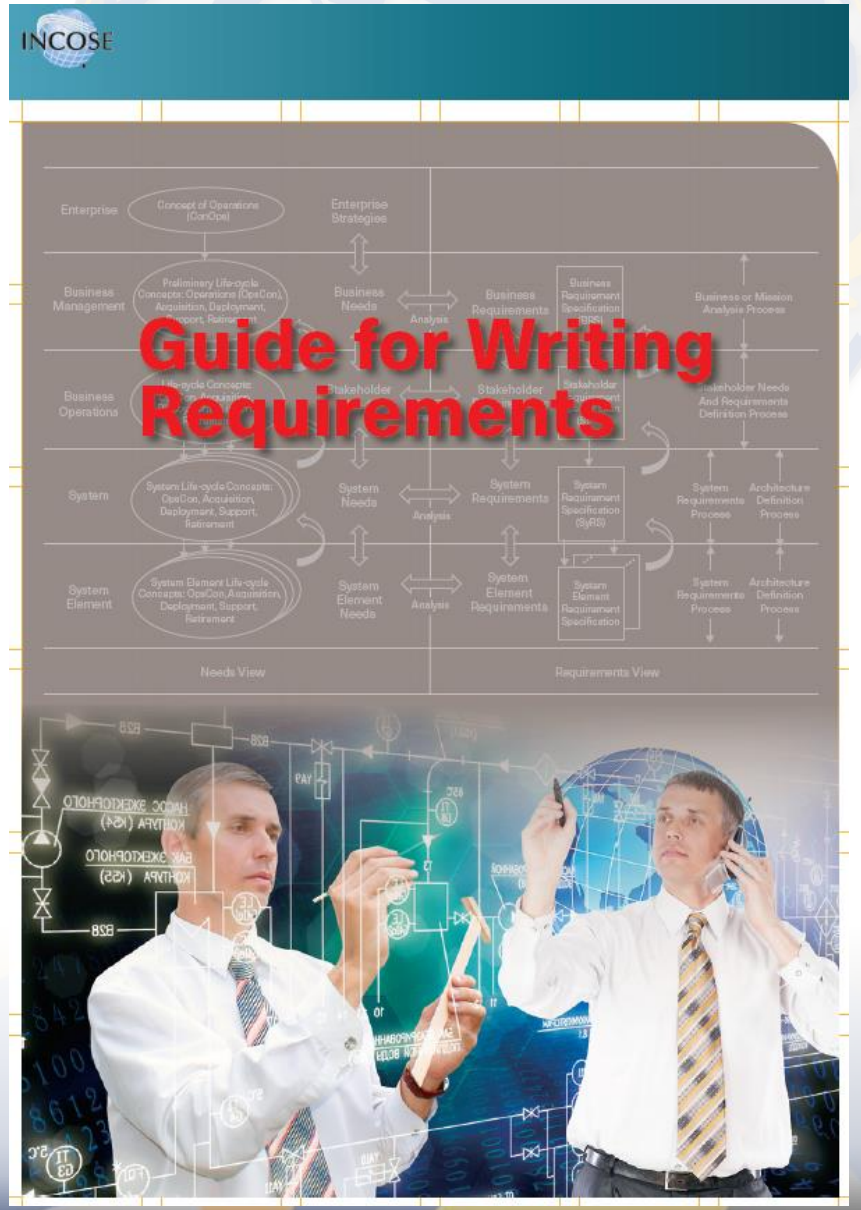
Figure 1 — Examples of Requirement Syntax

Some Benefits of NLP

- Mining – analysing older specifications for (compound) nouns and verbs (phrases)
- To build a domain specific relational database – or Knowledge Base, as a capital asset of intellectual property giving a competitive advantage for the future, which contains
  - Domain specific Terms
  - Agreed linguistic patterns
- Terms plus patterns = Rules

Term(s) + pattern = rule

- R1 – Use definite article "the" rather than the indefinite article "a."
- R2 – Use the active voice with the actor clearly identified.
- R3 – Make the subject of the requirement appropriate to the layer in which the requirement exists.
- R4 – Only use terms defined in the glossary.
- R5 – Quantify requirements precisely – Avoid imprecise quantifiers that provide vague quantification, such as "some," "any," "several," "many," "a lot of," "a few," "approximately," "almost always," "nearly," "about," "close to," "almost," "approximate."
- R6 – Use appropriate units, with tolerances or limits, when stating quantities – Explicitly state units for all numbers.
- R7 – Avoid the use of adverbs – words that end in -ly" .



The universal guide to writing requirements

Avoid the bad and Enforce the good

But exactly what is "bad" and what is "good"?

Here are some examples

Rule 01 Precision - Indefinite articles

Avoid

- a
- an

- The <system> shall provide a time display

Enforce

- The definite article

- The <system> shall display the Current_Time.

R05 Precision - Imprecise quantifiers

Avoid

- 'some', 'any', 'several', 'many', 'a lot of', 'a few', 'approximately', 'almost always', 'very nearly', 'nearly', 'about', 'close to', 'almost', 'approximate', 'significant', 'flexible', 'expandable', 'typical', 'sufficient', 'adequate', 'appropriate', 'efficient', 'effective', 'proficient', 'reasonable'.

- The Flight_Information_System shall display the current altitude to approximately 1 meter resolution

Enforce

- Quantification of performance criteria

- The Flight_Information_System shall display Current_Altitude plus or minus 1 meter.

R26 Consistency - Pronouns

Avoid

- 'it', 'this', 'that', 'he', 'she', 'they' and 'them`

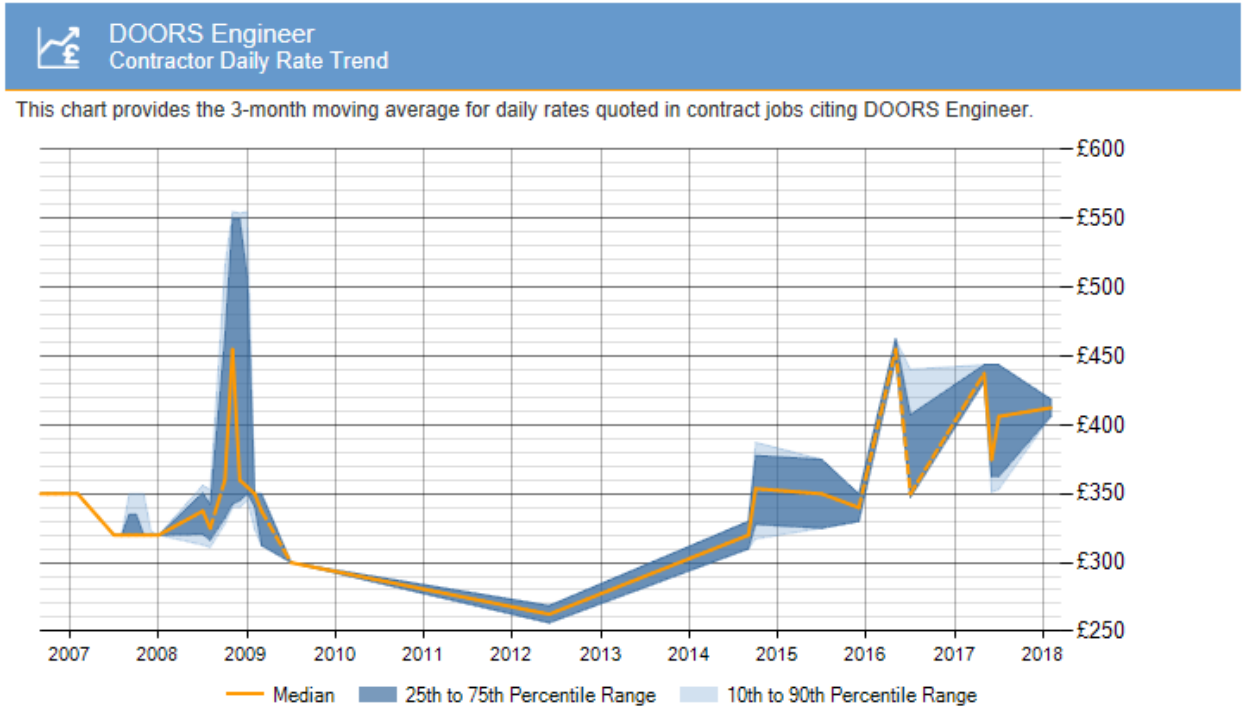
Enforce

- The name of the role of a Stakeholder e.g. The operator of the system , the maintainer of the system

- The controller shall send the driver his itinerary for the day. It shall be delivered at least 8 hours prior to his Shift

- The Controller shall send the Driver_Itinerary for the day to the Driver at least 8 hours prior to the Driver Shift.

Can I afford NLP of Requirements?



- For about €3995 you can get software to do comprehensive* quality analysis (yes, the Reuse suite)
- For €3995 @ €450 a day you can get maybe nine days of DXL development effort which will only give you
  - some textual analysis and
  - an ongoing maintenance burden

https://www.itjobswatch.co.uk/contracts/uk/doors%20engineer.do

*Text analysis, syntax analysis, semantic analysis, applying correctness, completeness and consistency rules

Content

- The Reuse Company
- The Presenters Background
- A Brief History of Requirements Quality
- The methods & functions needed to undertake:
  - textual analysis, and NLP syntax checking,
  - NLP for semantic checking, correctness, completeness and consistency
- Can script based languages hack NLP?
- Q&A

Can script based languages cope with Requirements Quality Analysis?

textual Analysis

- Yes, BUT …
- the DXL data structures are severely challenged and memory leaks and performance will be issues

Natural Language Processing

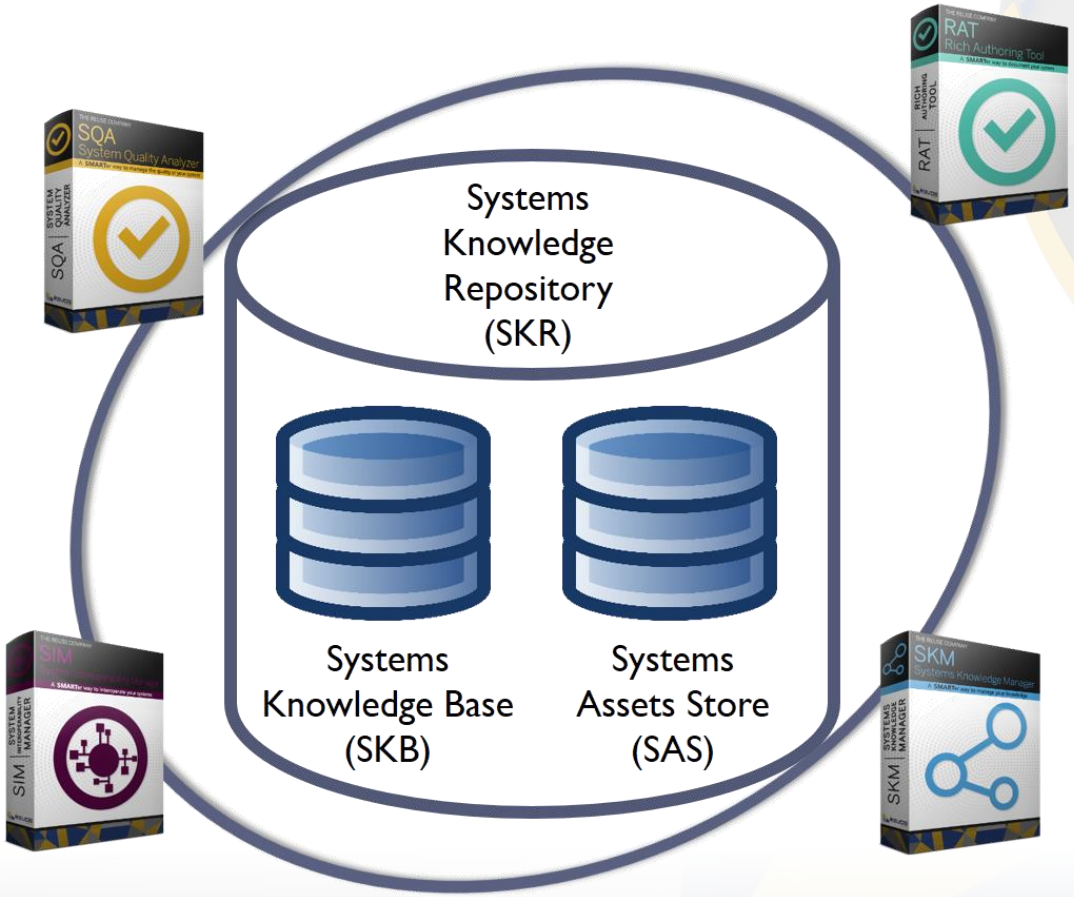
- NO!
- NLP extracts meaning, meaning is about relationships and so a relational database or equivalent is required

Do not use a hammer on a screw – it is not the tool for the job!

TRC Suite v18

SQA -System Quality Analyzer

Global Quality Management



RAT –Rich Authoring Tool

Smart text authoring

SIM –System Interoperability Manager

Tailorable Interoperability Platform

- R+ Manager

Managing requirements transformations

Managing models transformations

- T+ Manager

Managing traceability

- Reasoning Manager

Task based environment

SKM –System Knowledge Manager

Management of System Knowledge

Libraries


Q&A

TRC WEBINARS 2018

Next webinar

- **Topic:** Procuring systems: PQS for SMARTer acquisition?
- **Content:**
  Competition in a fair and transparent manner is the heart of procurement, that is why you probably already have a process in place to ensure the legal aspects of Fairness, Integrity and Transparency throughout the competition.
- But what about the two other corner stones and key guiding principles of Economy and Effectiveness, and not least the principle of Best value for money?
- This suite guarantees that the purchasing body is conducting the procurement process with maximum effectiveness in relation to the overall budget, and that the principle of Best value for money is indisputable and crystal clear from the initiation of the procurement project, through requirements definition, strategy selection and finally, bid evaluation and selection.
- **Dates:**
  - Tuesday 8th May 2018 at 5.00 pm CET
  - Wednesday 9th May 2018 at 9.00 am CET

WEBINAR ID	NAME	DATES	TIME
TRCW-01	Requirements Quality along the supply chain	16/01/2018 18/01/2018	5.00 pm CET 9.00 am CET
TRCW-02	Managing the quality ecosystem: DOORS, Rhapsody, Simulink and Modelica	20/02/2018 22/02/2018	5.00 pm CET 9.00 am CET
TRCW-03	Ontologies Configuration Management	13/03/2018 15/03/2018	5.00 pm CET 9.00 am CET
TRCW-04	Can script based languages, like DXL, hack Natural Language Processing?	10/04/2018 12/04/2018	5.00 pm CET 9.00 am CET
TRCW-05	Procuring systems: PQS for SMARTer acquisition	08/05/2018 09/05/2018	5.00 pm CET 9.00 am CET
TRCW-06	The SMARTER way to improve your requirement specifications	05/06/2018 07/06/2018	5.00 pm CET 9.00 am CET
TRCW-07	Knowledge and Quality management milestones in a SE organization	11/09/2018 13/09/2018	5.00 pm CET 9.00 am CET
TRCW-08	Automatic checking of quality metrics for logical and physical models	16/10/2018 18/10/2018	5.00 pm CET 9.00 am CET
TRCW-09	Following standards patterns in KCSE: An application to EARS patterns in RAT and SKM	03/07/2018 05/07/2018	5.00 pm CET 9.00 am CET
TRCW-10	Tracing system work products: T+ Manager	06/11/2018 08/11/2018	5.00 pm CET 9.00 am CET
TRCW-11	Defining your own quality rules in KCSE: A one-hour practical approach	11/12/2018 13/12/2018	5.00 pm CET 9.00 am CET
TRCW-12	The KCSE approach in a nutshell	15/01/2019 17/01/2019	5.00 pm CET 9.00 am CET
TRCW-13	Requirements Transformations	12/02/2019 14/02/2019	5.00 pm CET 9.00 am CET

the REUSE company