A Model-Based Approach to Extract Health Information from Textual Data

Diego Mandelli[1] and Congjian Wang[1]

[1]*Idaho National Laboratory* (*INL*), *1955 N Fremont Ave., Idaho Falls, ID, 83415, USA*

Diego.Mandelli@inl.gov
Congjian.Wang@inl.gov

Abstract

Current nuclear power plants generate a large amount of condition-based data that is stored to assess and monitor component health and performance. The format of this data can be either numeric (e.g., pump vibration data) or textual (e.g., condition report that assesses component health). While assessing component health from numeric data can be performed with a large variety of methods, extracting information from textual data still remains a challenge. Natural language processing methods are starting to be deployed in current power plants mainly to filter out non-safety-related incident reports by employing supervised machine-learning methods. However, these methods do not really provide the quantitative information that might be contained in textual data. This paper presents an approach to extract information from textual data (e.g., from maintenance reports) based on data analytics methods coupled with model-based system engineer models. Through a specific set of functions, our methods can identify whether a sentence contains health information of a component (e.g., degraded performance, anomaly behavior) or the causal relationship between two events (i.e., a cause-effect pair). An innovative element of our approach is that our analysis relies on models to identify links between textual elements. Such models are diagrams designed to represent system and component dependencies (from both a form and functional point of view). In our approach, these models emulate system engineer knowledge about component and system architecture. This paper presents in detail how the integration of Natural language processing methods and model-based system engineer models is performed, and it presents a few analysis examples focusing on centrifugal pumps.

1. Introduction

Industry equipment reliability and asset management programs are essential elements that help ensure the safe and economical operation of nuclear power plants (NPPs). The effectiveness of these programs is addressed in several industry-developed and regulatory programs.

The Risk-Informed Asset Management (RIAM) project (Mandelli, 2020) is tasked with developing tools in support of the equipment reliability and asset management programs at NPPs. These tools are designed to create a direct bridge between component health and lifecycle data and decision making (e.g., maintenance scheduling and project prioritization). Here, the primary focus is on supporting typical system engineer decisions regarding maintenance activity scheduling and component aging management. This is performed in a risk-informed context, where "risk" broadly includes both plant reliability and economics. This project combines data analytics tools to analyze equipment reliability (ER) data with reliability methods designed to support system engineer decisions (e.g., maintenance and replacement schedules, optimal maintenance posture) in a customizable workflow.

A RIAM research area focuses on the analysis of ER data with a particular emphasis on condition-based data, such as test and surveillance reports and component monitoring data. This article focuses on the analysis of ER textual data (i.e., incident reports [IRs] and work orders [WOs]), and it presents methods to assess component health information by merging two perspectives: a system engineer and a data scientist perspective (see Section 2).

The analysis of textual data has been investigated only recently using ML methods (Young et al. 2018) designed to assess their nature (e.g., safety or non-safety related). Instead, we aim to solve a different class of problems that requires reasoning rather than data learning. We are in fact addressing the analysis of ER data by focusing on causal reasoning and knowledge extraction from textual data.

2. EQUIPMENT RELIABILITY DATA TAXONOMY

Typically, a generic system structure component (SSC) is a part of plant (see Figure 1) designed to provide a specific function: its *emergence* (e.g., electric power generation for a power plant).

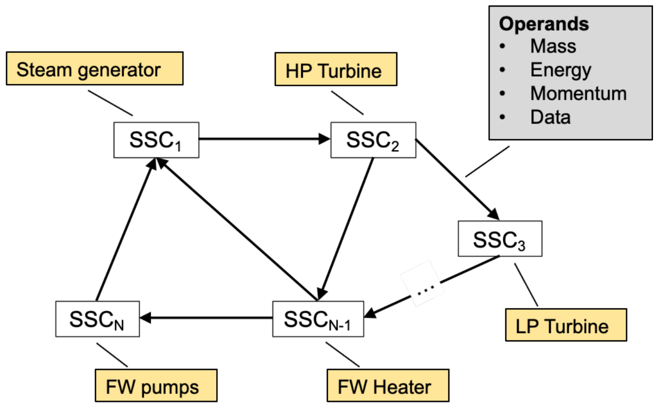


Figure 1. Graphical representation of SSC communicating and supporting each other through operands to provide plant emergence.

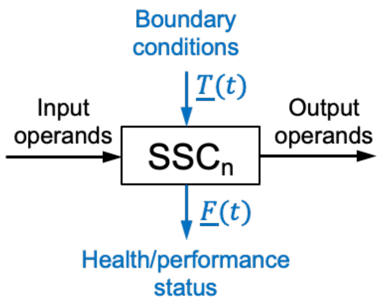


Figure 3. System engineer representation of an SSC where operands (see Figure 1) and SSC health related parameters (in blue) are monitored.

Each SSC contributes to the system emergence by providing a specified functionality used by other SSCs through a set of connections where operands (e.g., mass, energy, or data) are exchanged. The goal of a system health program is to monitor not only the correct operation of each SSC but also their health parameters, such as aging and degradation (indicated as $\underline{F}(t)$ in Figure 3). In addition, a system health program is designed to perform appropriate actions to assure component functionality (indicated as $\underline{T}(t)$ in Figure 3) (Xingang, 2021). In this article, $\underline{T}(t)$ also includes all the external stressors that contribute to altering component aging and degradation (e.g., workload, humidity).When analyzing the data generated by a generic SSC (i.e., $\underline{T}(t)$ and $\underline{F}(t)$ of Figure 2), it is vital to understand and capture the functional relationship between monitoring data, maintenance activities, and failure modes.

This can be accomplished by complementing the system engineer representation of an SSC with a data scientist representation of such an SSC. This is shown in Figure 2 where three levels are identified: the component level (which would correspond to Figure 3), a sensor and monitoring level (which retrieves and records portions of $\underline{T}(t)$ and $\underline{F}(t)$ in digital form), and data level. Data retrieved from $\underline{T}(t)$ (i.e., $\underline{\theta}(t)$ of Figure 2) can be textual (e.g., work orders) or numeric (e.g., environment temperature). We indicate here with "num" the numeric portion of $\underline{\theta}(t)$ while we indicate with "NL" the textual portion of $\underline{\theta}(t)$ (NL here stands for natural language). Data retrieved from $\underline{F}(t)$ has been portioned into two portions, component health and performance monitoring ($\underline{\varrho}(t)$ and $\underline{\gamma}(t)$), which can be numeric or textual in nature as well.

3. MBSE MODELING

ER data can have heterogenous data formats: textual, numeric, image, etc. Given system engineers' knowledge of SSC dependencies and architecture, they have many ways to

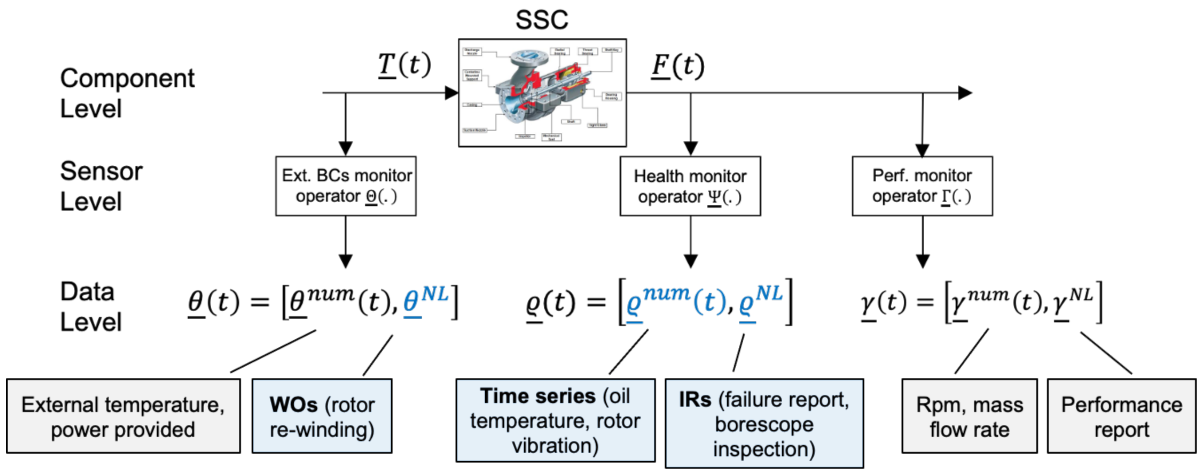

SSC
Component Level
Sensor Level
Data Level
$\underline{T}(t)$
$\underline{F}(t)$
Ext. BCs monitor operator $\underline{\Theta}(.)$
Health monitor operator $\underline{\Psi}(.)$
Perf. monitor operator $\underline{\Gamma}(.)$
$\underline{\theta}(t) = [\underline{\theta}^{num}(t), \underline{\theta}^{NL}]$
$\underline{\varrho}(t) = [\underline{\varrho}^{num}(t), \underline{\varrho}^{NL}]$
$\underline{\gamma}(t) = [\underline{\gamma}^{num}(t), \underline{\gamma}^{NL}]$
External temperature, power provided
WOs (rotor re-winding)
Time series (oil temperature, rotor vibration)
IRs (failure report, borescope inspection)
Rpm, mass flow rate
Performance report

Figure 2. Graphical representation of a data scientist perspective of a generic SSC.

interpret ER data, identify the causal links between event, and plan recovery actions. Is it possible for a machine to perform these tasks?

Currently, data analysis methods are based on machine-learning techniques that focus on finding patterns from data. While such approaches are valuable for some use cases, not all patterns provide insights about the system. This is often translated as: "correlation does not imply causation."

Here, we are looking at computational methods designed to support system engineers into the analysis of ER data using machine reasoning (rather than machine learning) methods. Machine reasoning is based on the construction of causal relations between data elements. This construction process cannot rely solely on data, but it requires models. These models represent specific aspects of the "real world" and are the foundations to perform such causal reasoning. When dealing with ER data, these models need to emulate system engineer knowledge about component and system architecture and dependencies.

Model-based system engineer (MBSE) practices (Borky, 2018) provide several solutions to model component from both *form* (i.e., which elements are part of the SSC) and *functional* (i.e., how SSC elements interact with each other, and which functions they support) points of view.

These solutions are based on MBSE languages that model system and SSC form and functional elements through a set of diagrams. The most commonly used languages are the object process methodology (OPM) (Dori, 2002) unified modeling language (William 2004), and systems modeling language (Friedenthal, 2008). We have chosen the OPM language because it provides the basic modeling elements we are looking for. For the scope of the analysis of ER data, elements of OPM diagrams will be used for elements contained in IRs and WOs.

Figure 4 provides an example of functional and form description of a generic SSC by employing an OPM diagram. An SSC OPM diagram provides an essential description of the SSC from both a form and functional perspective. This diagram explicitly indicates how SSC internal functions and processes act upon form elements and how form elements support these functions. From an ER perspective, monitoring activities (i.e., $\underline{F}(t)$ of Figure 3) act on both SSC functions (i.e., rotational frequency recorded for an induction motor) and form (i.e., blade corrosion of centrifugal pump) elements. On the other hand, degradation processes (i.e., $\underline{T}(t)$ of Figure 3) directly alter the form elements of the component that consequently affect SSC functional elements. Typically, from a reliability perspective, component failure modes are described in terms of loss of function; hence, in the OPM diagram, failure modes are only directly linked to the functional elements of the component. Lastly, note that maintenance activities (such as component replacement, refurbishment, or reconditioning) act on the form elements of components.

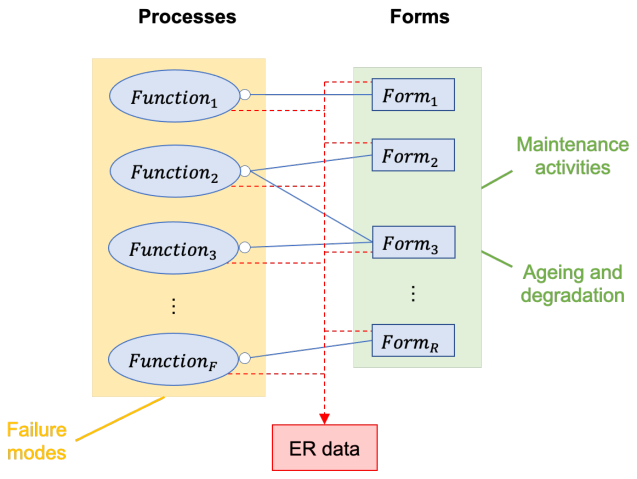


Figure 4. Generic presentation of an SSC OPM diagram and its link to SSC failure modes, aging degradation, maintenance activities, and ER data.

The OPM diagram of a component represents the key point to automatically understand and analyze health data $\underline{F}(t)$ (e.g., IRs). In particular, it clearly links monitored data with failure models that might affect component performance and maintenance activities that would restore component functionality. We are employing model-based data analysis methods to link component models with data rather than using machine-learning methods, which solely rely on available data to perform diagnostic and prognostic operations. Note that an OPM diagram extends failure modes and effects analysis tables by providing a form and functional description of the considered system in a graphical form.

In Appendix A, we provide a list of the main elements of an OPM diagram along with their semantic description.

We chose the OPM modeling language because:

- The OPM language is relatively simple in nature and provides the most basic functionalities required to extract causal relations between data elements.
- OPM diagrams can be easily digitally processed, and graph structures can be generated out of them.
- A direct link between OPM model and ER data (of any form, e.g., textual, or numeric) along with aging and degradation can be uniquely established.

An example of an OPM model for a centrifugal pump is in Figure 5. This simple representation includes the most basic elements that can be found in most OPM models and provides the following information:

- The form element "centrifugal pump" is composed by four elements: shaft, impeller, bearing, and motor (through the composition link).

- The function "increase fluid pressure" requires the form element "centrifugal pump" (through the instrument link).
- The function "increase fluid pressure" transform "fluid pressure" from low to high (through the transformation link).
- "Fluid pressure" is an attribute of the form element "fluid" (through the characterization link).

4. AN MBSE APPROACH TO PLANT HEALTH MANAGEMENT

As indicated in Section 1, the goal is to extract information from plant text data (e.g., maintenance reports, WOs, or IRs). The approach described in this article is not based on the identification of the *correlation* between data elements using machine learning. Instead, the goal is to identify and trace the *causal* relationship between events. Our proposed approach is based on causal inference (Pearl, 2009). Causal inference differs from classical statistical inference in that it is not based solely on data but requires a model that provides insights on the causal relationship between stochastic variables. We employ SSC OPM models (see Section 3) as a base of our causal analysis.

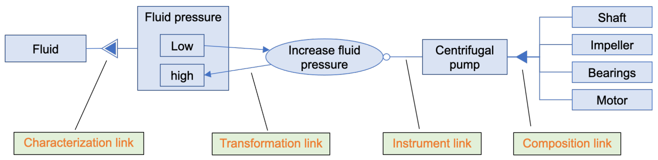


Figure 5. Simplified representation of a centrifugal pump using OPM.

The outcome of our causal analysis is graphical in nature where representation of the causal relationship between events is performed though directed acyclic graphs (DAGs). DAGs consist of nodes that represent stochastic variables and arrows that connect nodes and represent causal relationships between the nodes themselves.

In a typical NPP setting, several SSCs are constantly monitored, and relevant events are recorded in the plant monitoring and diagnostic center. As an example, specific events (e.g., SSC failure) might be caused by a process that results in SSC deterioration (in condition, performance, or both). Figure 6 displays a very basic relationship between cause and effect for one single SSC.

When dealing with complex systems (e.g., an NPP), multiple SSCs are linked together (see Figure 1) to support an emergence function (e.g., electricity production for an NPP); consequently, the DAG representation in this situation might be very complex.

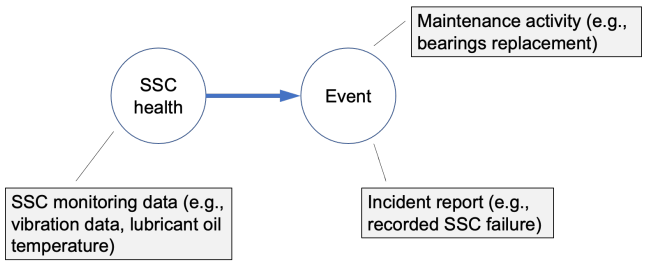

Maintenance activity (e.g., bearings replacement)
SSC health
Event
SSC monitoring data (e.g., vibration data, lubricant oil temperature)
Incident report (e.g., recorded SSC failure)

Figure 6. DAG representation between cause and effect in an NPP setting between SSC health (where monitoring condition-based data is available) and recorded event (e.g., SSC failure).

In this context, a DAG diagram represents the causal relationship between events and SSC health; it recreates the "story" behind observed events and data. We are in fact moving away from current methods that aim to identify correlations between events and data. However, note that the DAG diagram is unavailable and needs to be created. Our methods are designed to create a DAG diagram based on ER data. Note that, to achieve this objective, possessing ER data alone is not sufficient, we also need:

- Models that can provide insights on how SSCs operate and how they are connected to each other (see system engineer perspective indicated in Section 2)
- Links between ER data and SSC models (see data scientist perspective indicated in Section 2).

As anticipated, these elements are addressed by SSC OPM models that capture SSC form and functional elements and by data mining methods that capture the order, duration, and coincidence of events. Our data analytics methods employ OPM models and advanced data mining methods to:

- Capture information contained in available ER data (text and numeric)
- Explore causal relationship between ER data elements
- Exploit the generated relationships for anomaly detection, diagnostic, and prognostic purposes.

5. ANALYSIS OF TEXTUAL DATA

Most methods found in the literature (Young, 2018) process textual reports using supervised learning to predict the report nature (e.g., failure, operating). In this article, we are following a different path, to analyze the sentence structure of logs and reports, organize information in a structured form, and create a structural relationship among text objects (i.e., understand who and what did what, when, why, where). This is being accomplished by employing natural language processing (NLP) methods[1] (Lane, 2019) to perform two main tasks: syntactic and semantic analyses.

As a starting point, we characterize the content of a generic IR or maintenance report. Note that maintenance report content is fairly straightforward since it reports component replacement or restoration activity. On the other hand, IRs (along with SSC monitoring data) provide insights among the nodes of a complex DAG diagram. In other words, an IR describes a portion of a DAG: a node or the causal relation between two nodes. From an initial assessment of a dataset of textual data generated by an NPP, we were able to identify three classes of SSC health related IRs:

- Class 1 IR: *Health status*. The IR reports a DAG node, either an event (e.g., SSC malfunction) or data regarding component health (e.g., excessive corrosion on pump impeller).
- Class 2 IR: *Cause-effect relation*. The IR reports a causal relation between two DAG nodes; the content of these nodes can be any combination between events and SSC health information linked by a causal relationship.
- Class 3 IR: *Time-based relation*. The IR reports the time occurrence of multiple events (i.e., multiple DAG nodes) without explicitly specifying any causal relationship among them.

This classification scheme defined by these three mutually exclusive class needs is being validated with actual NPP data to measure its validity (i.e., the degree to which the three classes are in fact mutually exclusive). In the validation process, we can measure the percentage of actual NPP IRs that falls in each class, and more importantly, the percentage of IRs that do not fall in either of the three classes. Note that, the classification provided above are relevant to IRs related to plant equipment performance.

This article focuses on the first two classes presented above and presents the main NLP analysis workflows for both classes (see Sections 5.1–5.3).

5.1. NLP Analysis Pipeline

The first step in the analysis of textual data is to perform a syntactic analysis (Lane, 2019) of the raw text by employing the rules of formal grammar. Here, we assumed that the text is in a digital form (typically a string form). The syntactic

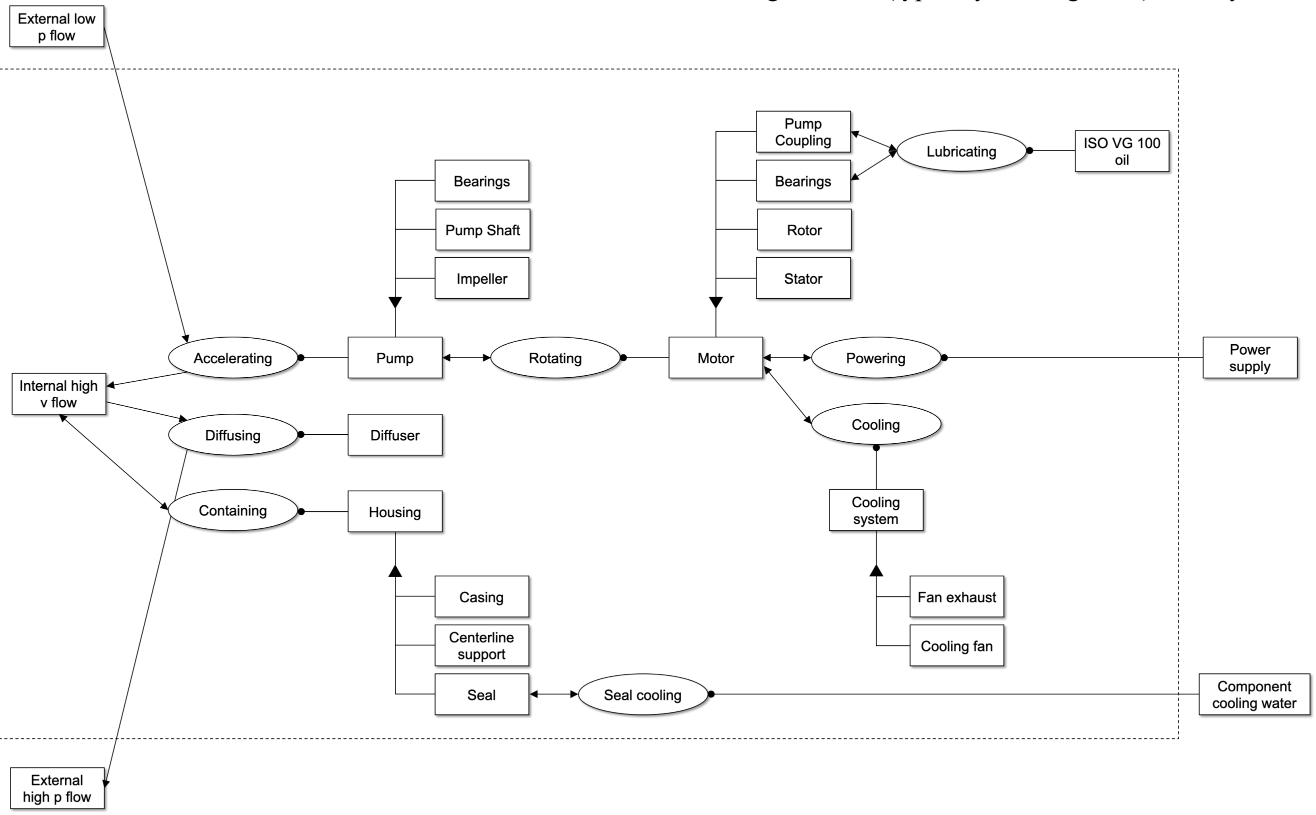


Figure 7. Reference OPM diagram of a centrifugal pump.

[1] In this work, we are employing three main `Python` libraries: `STANZA` (https://stanfordnlp.github.io/stanza/), `NLTK` (www.nltk.org), and `SPACY` (https://spacy.io).

analysis follows these main steps (see Table 1 for a more detailed list of NLP analysis steps):

1. Sentence segmentation and word tokenization: each sentence is translated into a list of string elements.
2. Part of speech (POS) tagging: the grammatic elements of each string (e.g., nouns, verbs) are identified using POS tags developed in the Penn Treebank project.[2]
3. Named entity recognition: text entities (e.g., names, dates, events) are classified and identified (e.g., component ID, event occurrence time).
4. Relation extraction: a knowledge graph is created where entities identified in Step 3 are linked together in a graph that reflects the structure of the original sentence.

Steps 1–8 in Table 1 are common in any NLP analysis (Lane, 2019). Our approach deviates from the standard NLP method in Steps 9 and 10. In Step 9 of Table 1, we identify elements of the SSC OPM model (i.e., operands, forms, or functions as indicated in Section 3). From each SSC OPM model, we can generate a set of textual elements that lists not only all OPM elements but also their relationship. In Step 10, we infer the causal relationship between elements in the IR (see Sections 5.2 and 5.3). These relationships are *cause* and *consequence*. Here, we exploit the observations reported in the IR by plant system engineers and trace back causal relationship with other IRs using the SSC OPM models.

Table 1. List of the main NLP analysis steps.

ID	Steps	Note
1	Retrieve raw text	
2	Cleaning	Process of cleaning raw text data from non-text-related elements[3]
3	Segment sentence	Each sentence is analyzed separately
4	Clean punctuation	Punctuation is removed
5	Tokenize sentence	Each sentence is split into a set of words
6	Stemming and lemmatization	Each word is converted into its own dictionary form or to its stem or root form
7	POS tagging	Process of marking each word as corresponding to a particular POS using grammatical rules
8	Entity recognition	Process designed to identify and classify named entities into predefined classes such as: SSC type, systems, locations, time values
9	OPM entity recognition	Process designed to identify OPM elements (functions or forms)
10	Information extraction	Process of extracting information content from text (see Sections 5.2 and 5.3)

5.2. Analysis of Health Status Reports

The methods designed to extract information from Class 1 IRs have been structured in a similar way to the one presented in (Doan, 2019). We, in fact, based our methods on a set of rule templates based on specific trigger words and relations. We focused on the development of status nouns and verbs that would indicate a degradation of SSC functions or internal elements.

The chosen set of status words includes verbs, adjectives, and nouns obtained from the `WordNet`[4] database. For Class 1 IRs, we have identified three categories of status words (negative, anomalous, and positive), shown in Table 2, Table 3, and Table 4, respectively. Table 5 provides an initial list of status relations encoded using `STANZA` Python library.

Table 2. Subset of negative status nouns, verbs, and adjectives.

Status Nouns	Status Verbs	Status Adjectives
Failure	Fail	Unable
Degradation	Degrade	Ineffective
Breach	Break	Anomalous
Fracture	Decline	
Decline	Go bad	
Decay	Rupture	
Loss	Breach	
	Reduce	
	Increase	
	Decrease	
	Fracture	
	Aggravate	
	Worsen	
	Lose	

Table 3. Subset of positive status nouns, verbs, and adjectives.

Status Nouns	Status Verbs	Status Adjectives
Operation	Function	Operating
Functioning	Work	Operational
	Operate	Functional

[2] Penn Treebank project official website: https://catalog.ldc.upenn.edu/LDC99T42

[3] For this task we have employed `Beautiful Soup` (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)
[4] WordNet official website: https://wordnet.princeton.edu/

	Run	Usable

The status relations indicated in Table 5 were coded in a Python-based code that relies on the Stanford NLP library `STANZA`[5]. This library provides a set of algorithms for linguistic analysis that can be used to construct fairly complex NLP analysis pipelines. Once the IR has been processed using all steps listed in Table 1, a set of tuples is created from each sentence in the form (SSC, form and function, health status). These tuples are designed to represent in digital form the DAG node as follows:

(SSC; subject = ‘OPM function/form’; health status = ‘ok, ‘degraded’ or ‘anomalous’)

An example of a Class 1 IR is “Oil puddle was found in proximity of CCW Pump 1B.” By using the NLP analysis Steps 1 through 7 listed Table 1 using `STANZA` and `NLTK` Python libraries, the resulting grammatical structure of the IR is shown in Figure 8. This figure shows the POS tags represented on top of each word, and the grammatical dependencies between words (represented with arrows).

Table 4. Subset of anomalous status nouns, verbs and adjectives.

Status nouns	Status verbs	Status adjectives
Observation	Find (out)	Unchanged
Detection	Observe	Unaltered
	Detect	Constant
	Determine	Consistent
	Discover	Stable
	Get	Unaffected
	Notice	
	Become	
	Record	
	Register	
	Show	

Table 5. Subset of status relations.

Relation
A (noun) “status verb” “status adjective”
A (noun) “status verb” “status verb-ing”
“Status adjective” B (noun) “status verb”
“Status noun” “status verb” prep. B (noun)

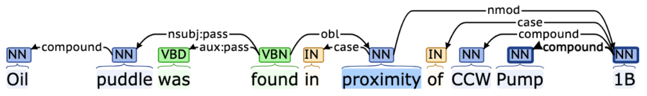


Figure 8. Grammatical decomposition and analysis of the example Class 1 IR.

Step 8 in Table 1 is accomplished by looking in the IR for specific SSC tags (i.e., CCW Pump 1B). It is here assumed as well that SSC tags are unique and given. Once the SSC has been identified, its OPM model (see Figure 7) is employed to identify OPM elements in the sentence that refer to such model (see Step 9 in Table 1). In this case, the word “oil” is linked to the OPM form element “ISO VG100 oil”.

Next, Step 10 of Table 1 is performed where the verb “find” is identified (i.e., verb being part of anomalous status, see Table 4) and a relation (see Table 5) is matched. The following tuple is constructed:

(SSC=CCW Pump 1B; subject=ISO VG100 oil; health status=anomalous)

Note that now the OPM model of Figure 7 can propagate anomalous behavior contained in the IR to other OPM elements such as:

Oil → motor → rotating → pump → accelerating function

5.3. Analysis of Health Status Reports

For the extraction of the causal relationship between sentence elements of a sentence, we identified the works presented by Doan (2018) as candidates to effectively perform such a task since it provides robust and explainable analysis results. As described in Section 5.2, this method is again based on the identification of a set of rule templates based on specific trigger words and relations. The chosen set of words includes verbs and nouns obtained from the `WordNet` database[6] and is shown in Table 6.

Table 6. Subset of trigger causal verbs and nouns.

Causal Nouns	Causal Verbs
Result	Cause
Reason	Stimulate
Cause	Make
	Derive
	Trigger
	Result
	Lead
	Increase
	Decrease

Similarly, the chosen set of causal relations has been constructed from common English syntactical rules as indicated in Table 7. These causal relations were coded in a Python-based code that relies on the Stanford NLP library `STANZA`.

Once the IR has been processed using all steps listed Table 1, a set of tuples is created from each sentence in the form cause=A → consequence=B. These tuples are designed to represent the DAG node in digital form as indicated in Figure 6:

[5] STANZA official website: https://stanfordnlp.github.io/stanza/

[6] WordNet official website: https://wordnet.princeton.edu/

(SSC, OPM form/function, health status) → (SSC, OPM form/function, health status)

As an example of Class 2 IR is:

Bearing failure of CCW Pump 1B caused reduced flow.

By using the NLP analysis Steps 1–7 listed in Table 1 using `STANZA` and `NLTK` Python libraries, the resulting grammatical structure of the IR is shown in Figure 9. This figure shows the POS tags on top of each word and the grammatical dependencies between words (represented with arrows).

Table 7. Subset of causal relations, extended from (Doan, 2018).

Relation
A (noun) "causal verb" B
A (verb) "causal verb" B
B was "causal verb" A
A is a "causal noun" of B
B was "causal verb" by A (verb)
A "causal verb" in/to/from B

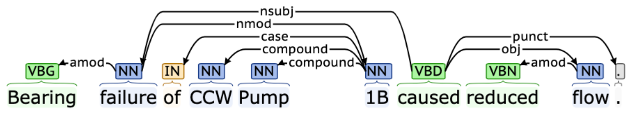


Figure 9. Grammatical decomposition and analysis of the example Class 1 IR.

Step 8 in Table 1 is accomplished by looking in the IR for specific SSC tags (i.e., CCW Pump 1 B). Again, we assumed that SSC tags are unique and given. Once the SSC has been identified, its OPM model (see Appendix A) identifies OPM elements in the sentence that refer to such a model (see Step 9 in Table 1). In this case, these OPM elements in the text have been identified as "bearing" and "flow."

Next is Step 10 of Table 1; here, the causal verb "cause" (see Table 6) and a specific causal relation of the sentence (see Table 7) are identified that produce the causal relationship as follows:

(CCW Pump 1B, bearing, degraded) → (CCW Pump 1B, high internal v flow, degraded)

6. CONCLUSIONS

In this paper, we have presented a series of methods and models designed to analyze ER data with particular focus on textual data. We have introduced an approach to extract quantitative information from ER textual data, such as IRs. Rather than focusing on machine-learning heuristics, the system view of the SSC (through a OPM diagram) provides knowledge required by our data analysis methods to extract knowledge from the textual data retrieved by IRs and WOs and identify possible causal links again using the SSC OPM diagram. The immediate applications for this kind of methods range from component diagnostic to history retrieval. In the first application, these methods are designed to integrate multiple data elements, and identify the causes of any anomalous behavior. In the second application, we are targeting the retrieval of the historic performance of a component to capture trends and reliability measures such as component unavailability.

ACKNOWLEDGEMENT

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 with the U.S. Department of Energy. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

REFERENCES

Borky, J., Bradley, T. (2018). *Effective model-based systems engineering*. Springer.

Dori, D., Crawley, E. (2002). *Object-process methodology: A holistic systems paradigm*. Springer.

Doan, S., Yang, E. W., Tilak, S. S., Li, P. W., Zisook, D. S., Torii, M. (2019). Extracting Health-Related Causality from Twitter Messages Using Natural Language Processing. *BMC Medical Informatics and Decision Making*, 19, 71–8. https://doi.org/10.1186/s12911-019-0785-0.

Friedenthal, S., Moore, A., Steiner, R. (2008). *A practical guide to SysML: The systems modeling language.* Morgan Kaufmann.

Young, T., Devamanyu, H., Poria, S., Cambria, E. (2018). Recent Trends in Deep Learning Based Natural Language Processing. *IEEE Computational Intelligence Magazine*, 13(3), 55–75. https://doi.org/10.1109/MCI.2018.2840738.

Lane, H., Hapke, H., & Howard, C. (2019). *Natural language processing in action: Understanding, analyzing, and generating text with Python*. Manning Publications.

Mandelli, D., Wang, C., Cogliati, J., Smith, C., Hess, S., Sugrue, R., Pope, C., Miller, J., Ercanbrack, S., Cole, D., & Yurko, J. (2020). Integration of data analytics with plant system health program. Idaho National Laboratory Technical Report, INL/EXT-20-59928.

Pearl, J. (2009). Causal Inference in Statistics: An Overview. *Statistics Surveys*, 3, 96–146. https://doi.org/10.1214/09-SS057.

William, S. (2004). *The object primer: Agile model driven development with UML 2*. Cambridge University Press.

Xingang, Z., Kim, J., Warns, K., Wang, X., Ramuhalli, P., Cetiner, S., Kang, H. G., & Golay, M. (2021). Prognostics and Health Management in Nuclear Power

Plants: An Updated Method-Centric Review with Special Focus on Data-Driven Methods. *Frontiers in Energy Research*, 9. https://doi.org/10.3389/fenrg.2021.696785.

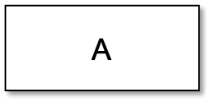
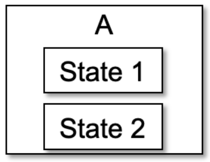
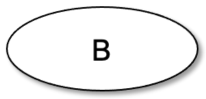
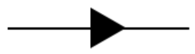
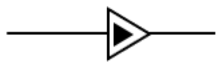
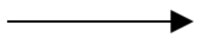
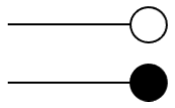
BIOGRAPHIES

Diego Mandelli is a research and development scientist in the Reliability, Risk and Resilience Sciences Department at Idaho National Laboratory. His areas of expertise include risk, reliability, and system health management. His research focuses on the development of probabilistic methods based on machine learning, data mining, and optimization algorithms. He is currently employing these methods to perform state-of-the-art simulation-based safety assessments (also known as dynamic PRA), system health management and stochastic resource optimization. His developed methods range from data preprocessing, system modeling, system analysis, data mining and visualization, and decision-making. He holds a Ph.D. in nuclear engineering from The Ohio State University (2011). He is a member of the American Nuclear Society.

C. Wang Dr. Congjian Wang is a computational nuclear scientist at Idaho National Laboratory. He received his B.E. in engineering physics and his Ph.D. in nuclear engineering and carried out postdoctoral research focused on machine learning, advanced sensitivity and uncertainty quantification, verification, validation, and data assimilation. He has more than 10 years of experience in developing and implementing artificial intelligence algorithms, including physics-based reduced-order modeling, surrogate modeling, data mining techniques, and deep neuron networks. He is the main developer of the Risk Analysis Virtual ENvironment framework, and he is mainly responsible for developing and implementing supervised and unsupervised learning algorithms, cross validations, and verification and validation metrics within this framework.

APPENDIX A

A subset of the basic elements (Dori & Crawley, 2022) of an OPM diagram are shown below along with their description.

OPM Element	Description
	Object A is a tangible entity that exists
	Object A has two states: State1 and 2
	Process B transforms an object
	Link designed to decompose an object into its basic elements
	Link designed to define attributes of an object
	Link designed to indicate a transportation activity between a process and an object
	Link designed to represent objects that support a process