Human Factors in Software and Systems Engineering, Vol. 61, 2022, 21–25
https://doi.org/10.54941/ahfe1002518

Procedure Parsing: A Method for Parsing Handwritten Documents into Computer-Based Procedures

Stacey Whitmore

Idaho National Laboratory, Idaho Falls, ID 83401, USA

ABSTRACT

The nuclear industry is heavily procedure-driven, where almost everything has a step-by-step instruction that is expected to be followed in detail. Historically, these procedures were printed on paper. Recently, the industry began transitioning towards electronic copies (e.g., PDFs on tablets). One major driver for this transition is the introduction of human error and loss of situation awareness when using paper copies. However, electronic copies of documents inherently have the same error traps as their paper cousins. Therefore, there is an increased interest in a way to utilize the information in the step-by-step guidance, but to present it in a dynamic manner that guides the user and adapts to any encountered conditions. Researchers at Idaho National Laboratory propose a flexible, automated method based on document parsing and augmented by natural language processing (NLP) techniques, to address these shortcomings and capitalize on these recent advancements in machine learning. The proposed method provides a cost-effective solution for computer-assisted procedure parsing of hand-written control room procedures, originally authored in Word or PDF formats, into instructions that can be displayed as computer-based procedures in a modern graphical user interface. The researchers devised, implemented, and demonstrated the Operating Procedure Extender for Novel Systems (OPENS) method in 2020. The key to OPENS is to map the original procedure text into a context-free grammar, tying content to equipment, locations, and other steps, actions, etc. This formal grammar is then used to isolate and define keywords and action verbs, such as "measure" or "evaluate" and tie them to specific equipment referenced within that step or located in other steps, sub-steps, actions, sub-actions, and tables throughout the procedure.

Keywords: Procedure parsing, Procedure compiling, Computer-based procedures, CBP, Context-free grammars, Abstract syntax tree, Dynamic procedures, OPENS, Operating procedure extender for novel systems

INTRODUCTION

Computer-based procedures (CBPs) take the logic contained within handwritten, paper-based procedures (PBPs) and present them in a computerized form with interactive controls and capabilities for branching, calculating values, and referencing other steps. While existing PBPs have a history of establishing safe operations at a nuclear power plant (NPP), CBPs may improve performance by reducing errors (Le Blanc & Oxstrand, 2013). Converting existing PBPs into CBPs may further enhance safety and optimize efficiency (Fink et al., 2009; Le Blanc & Oxstrand, 2012); however, one obstacle in making this transition is the cost and time involved in converting volumes of existing PBPs into a "self-describing", easy-to-understand, text formats, such as JSON ('JSON', no date) or XML ('XML', no date), that is language independent and can be standardized and read by a CBP application.

OPENS is a procedure parser that iterates over electronic, paper-based procedure documents (e.g., in Word or PDF format) and tokenizes its text to extract procedural information and compiles it into a meaningful data structure called an abstract syntax tree (AST). It then formats the contents of that AST into a text format (e.g., JSON) or a markup language such as XML so it can be displayed on a graphical user interface as an interactive CBP.

This is beneficial to nuclear and other industries that rely heavily on handwritten procedures as it provides a quick and inexpensive way to transition from PBPs to CBPs, sparing the operator or procedure writer the laborious process of converting these handwritten procedures into a computer-readable format and entering them into a computer by hand. This could be revolutionary in expediting this transition as it provides an opportunity for operators, to try CBPs and convert several handwritten procedures into CBPs in a matter of seconds.

OPENS METHODOLOGY

OPENS compiles handwritten PBPs in 4 main phases: the scrubbing phase, the scanning phase, the parsing phase, and the formatting phase.

Scrubbing Phase

A paper-based, procedure written and stored as a Microsoft Word document consists of several files zipped into one .docx file. A single .docx file, for instance, is actually a compressed file (like a .zip file) comprised of several Open XML documents embedded with data that instruct the application used to display the document with directives regarding its layout, styling, and font sizes (Microsoft, 2022). This is similar to the way JSON or XML data can inform a CBP application with data concerning the behavior or presentation of steps in a procedure.

The initial scrubbing phase consists of unzipping the Word document to dissect its Office Open XML (OOXML) file constituents. It then scans them for marked-up items such as heading numbers, bold and italicized words as well as referenced assets such as tables and diagrams (and the items that reference them). Next, it extracts only the necessary procedure data into a plain text format, ignoring irrelevant information such as font and margin sizes, in preparation for the subsequent scanning phase.

Scanning Phase

During the scanning phase, the scanner iterates through the scrubbed plain text, scanning for tokens matching the replacement rules defined by its context-free grammar (See Figure 1). The concept of phrase-structure

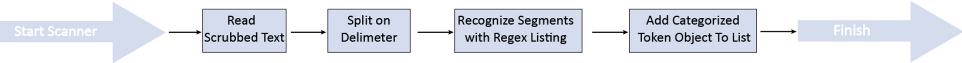
Start Scanner
Read Scrubbed Text
Split on Delimeter
Recognize Segments with Regex Listing
Add Categorized Token Object To List
Finish

Figure 1: Scanning/tokenizing stages.

grammars (or context-free grammars) was formalized by Noam Chomsky in the mid-1950s and is a set of production rules in the form

$$A \to \alpha$$

where A represents a single non-terminal symbol (i.e., a start point) and α a string of terminals and/or other non-terminals or an empty string (Chomsky, 1956). Any non-terminal on the left-hand side of the arrow can be replaced by the defining rules on its right-hand side regardless of any context created by surrounding symbols. This is what sets context-free grammars (CFGs) apart from more general context-sensitive grammars—they are essentially context agnostic.

For instance, suppose the non-terminal Sn (SubstepNumber) is defined as a string of any lower-case, alphabetic character (α) followed by a period, followed by a closing parenthesis, and the non-terminal W (Word) is defined as any string of one or more (*) alphabetic characters not separated by spaces (α*).

$$Sn \to \alpha.$$

$$W \to \alpha *$$

A non-terminal on the left-hand side (LHS) of an arrow, e.g., Substep, could then be defined using these non-terminals as right-hand-side (RHS) production rules.

$$Substep \to Sn\ W$$

These rules will then always be able to replace any Substep, or vice versa, even if that Substep is enclosed within another Step or surrounded by other Substeps. The context does not change this definition. The RHS terminals become the tokens that the parser can employ to identify some of the LHS non-terminals that make up the leaves it uses to construct the AST in the next phase.

Parsing Phase

During the parsing phase, the OPENS parser iterates through the tokens collected by the scanner in the previous stage to generate an AST. It does this by combining these tokens according to the production rules defined in the procedure grammar to form objects (See Figure 2). These objects can represent the LHS non-terminals in the formal grammar and the individual scanner tokens represent the terminal and non-terminal rules on the right-hand side of the production. Each node or leaf on the AST represents an instance of a step, substep, or other instruction object and can contain information regarding the relation of those objects to one another (e.g., steps and substeps to notes, cautions, or warnings [NCWs] or their referenced tables or figures, etc.).

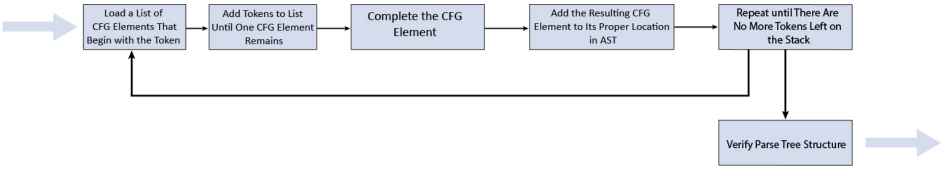
Load a List of CFG Elements That Begin with the Token
Add Tokens to List Until One CFG Element Remains
Complete the CFG Element
Add the Resulting CFG Element to Its Proper Location in AST
Repeat until There Are No More Tokens Left on the Stack
Verify Parse Tree Structure

Figure 2: Parsing stages.

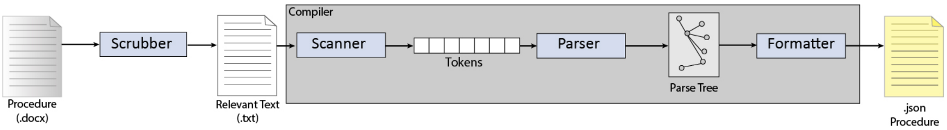

Figure 3: OPENS procedure lifecycle.

Formatting Phase

During the final formatting stage, the AST is validated, and all information contained in the AST is deserialized into formatted JSON or XML. (See Figure 3 for an overview of all of the phases.) This single text file, representing the procedure, has a small footprint and can also be used to quickly track a state variable within the computer-based procedure application such as the currently active step, each step's completed time and duration or branching information, etc.

The XML is useful in preserving the relational aspects of the procedure for referencing tables and branching information so the user can be directed to the next appropriate active step based on the values entered for that step and previous steps. The JSON, being lighter, less verbose and easier to follow, is useful for storing and exchanging data objects used to track responses to previous steps and state changes. (Although, either format could be used.)

The techniques the researcher developed could further be improved by the integration of recent advancements in machine learning. NLP methods could standardize documents, correct grammatical errors, and provide automated semantic validation. The researcher expects that self-supervised techniques applied to collections of natural language instructions could strengthen the model with a broader context.

All these methods together give us a practical way to automatically extract protocols from documents and user interactions, empowering researchers, procedure writers, and nuclear operators while moving the industry forward.

CONCLUSION

The concepts proven by OPENS could automate the process of converting PBPs to CBPs, significantly reducing the time and cost involved. By using NLP methods to glean, and store, contextual and relational information regarding the objects that form the procedure's AST, on the objects themselves, protocols could robustly be extracted from procedures with varying design requirements and style guidelines across various industries and even provide correction for grammatical errors. These features would greatly benefit researchers, NPP operators, and procedures writers in industries where safety is a concern.

ACKNOWLEDGMENT

The author would like to acknowledge Katya LeBlanc and Johanna Oxstrand for their contributions in the field of computer-based procedures and their efforts in creating standardized JSON and XML formats for use in computer-based procedure applications (Le Blanc, 2021) and for making the development of OPENS possible. The author would also like to acknowledge Jardain Kenner for assisting in the development of the OPENS software and Jacob Lehmer for assisting with the diagrams used in this paper.

REFERENCES

Chomsky, Noam (Sep 1956). “Three Models for the Description of Language”. *IRE Transactions of Information Theory.* VOL 2 (3), 113-124. doi:10.1109/TIT.1956.1056813.

Fink, R., Killian, C., Hanes, L., & Naser, J. (2009). Guidelines for the design and implementation of computerized procedures. Nuclear News. VOL 52 (3), 85–88, 90.

‘JSON’ (no date) Available at https://www.w3schools.com/js/js_json_intro.asp (Accessed: 9 Apr 2022).

Le Blanc L., Katya. Oxstrand, Johanna H. (2013) “Computer-Based Procedures for Nuclear Power Plant Field Workers: Preliminary Results from Two Evaluation Studies”, Proceedings of the Human Factors and Ergonomics Society 57th Annual Meeting, San Diego, CA.

Le Blanc, K.L. Oxstrand, J.H. (2021). *Method To Convert A Written Procedure To Structured Data, and Related Systems and Methods.* U.S. Patent 11,126,789.

Microsoft. (April 04, 2022) Word Processing (Open XML SDK). The Microsoft Online Documentation for Open XML: https://docs.microsoft.com/en-us/office/open-xml/word-processing

‘XML’ (no date) Available at https://www.w3schools.com/xml/xml_whatis.asp (Accessed: 9 Apr 2022).