Copyright ⓒ 2019 KSII

# Building a Rule-Based Goal-Model from the IEC 62304 Standard for Medical Device Software

**DongYeop Kim[1], Byungjeong Lee[2], Jung-Won Lee[1*]**
[1] Department of Electrical and Computer Engineering, Ajou University
Suwon, KS 002 - KR
[e-mail: dongdongy@ajou.ac.kr, jungwony@ajou.ac.kr]
[2] Department of Computer Science and Engineering, The University of Seoul
Seoul, KS 013 - KR
[e-mail: bjlee@uos.ac.kr]
*Corresponding author: Jung-Won Lee

*Received February 27, 2019; revised April 3, 2019; revised May 13, 2019; accepted June 13, 2019; published August 31, 2019*

## Abstract

IEC 62304 is a standard for the medical device software lifecycle. Developers must develop software that complies with all specifications in the standard for licensing. However, because the standard contains not only a large number of specifications, but also domain-specific information and association relationships between specifications, it requires considerable effort and time for developers to understand and interpret the standard. To support developers, this paper presents a method for extracting the contents of the IEC 62304 standard as a goal model, which is the core methodologies of requirements engineering. The proposed method analyzes the grammar of the standard to robustly extract complex structures and various information from standard specifications and define rules that extract goals and links from syntactic element units. We validated the actual extraction process for the standard document experimentally. Based on the extracted goal model, developers can intuitively and efficiently comply with the standard and track specific information within the medical software and standard domains.

***Keywords:*** Goal Model, IEC 62304, Medical Device, Software, Requirements Engineering

A preliminary version of this paper appeared in KSII The 10th International Conference on Internet (ICONI) 2018, December 16-19, Phnom Penh, Cambodia. This version includes a concrete analysis and supporting implementation results on extracting Goal-Model from IEC 62304. This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning (NRF- 2014M3C4A7030504) and supported by the MSIT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2019-2016-0-00309) supervised by the IITP(Institute for Information & communications Technology Promotion).

## 1. Introduction

Medical device software must be strictly controlled in terms of safety and performance. Therefore, Korea, the United States, and many other countries have developed strict licensing procedures for medical device software. IEC 62304 has been established as an international standard for such software and major medical device certification bodies, such as the Conformité Européenne and US Food and Drug Administration, have legislated the use of this standard for the certification of medical device software [1]. Specifically, IEC 62304 is a standard for the medical device software lifecycle [2]. It contains five processes related to research and development (R&D) that ensure the safety and performance of medical device software through proper lifecycle progress. Software manufacturers are required to develop and manage software according to the standard specifications from the initial design stage to the post-launch maintenance stage of a product for licensing. In addition to simply developing software, a developer must understand all specifications in the standard, comply with them, and plan development accordingly. The development process must be documented carefully for a significant number of specifications in the certification process.

However, there are various difficulties for software developers attempting to adhere to this standard. First, the standard is very long at 80 pages and 420 specifications. It is difficult in terms of time and labor for software developers to identify, comply with, and document all the content of the standard specifications. Specifically, this process is difficult because the standard provides developers with specific information related to licensing in the form of sentences containing specific expressions in addition to the basic content related to software R&D. For example, the 420 specifications in the standard are divided into different levels of importance, such as mandatory, recommended, and possible, in terms of licensing. This information regarding the required compliance for various specifications is expressed only through auxiliary verbs in the corresponding sentences. Specifically, the sentiment expressed by the auxiliary verb "shall" is that a requirement is mandatory and will be inspected during the certification process. Therefore, a developer must recognize all sentences using "shall" and other auxiliary verbs for licensing purposes. In this manner, information that is important for licensing, such as “compliance with safety level” and “whether or not to be documented,” is implied through specific expressions within sentences. In this paper, we refer to such information as standard domain-specific or medical domain-specific information. This type of information is not expressed intuitively in the standard, meaning it requires interpretation by developers. Additionally, because certain specifications in the standard have reference relationships with other specifications, developers must understand the contents of multiple specifications to satisfy a single specification. For example, there is a development process and risk management process among the processes proposed in IEC 62304. Although these two processes are documented separately in chapters five and seven, they both affect the software lifecycle and contain concurrent specifications. In IEC 62304, any specification for the development process may refer to other specifications because certain specifications can only be satisfied based on compliance with other risk management process specifications. Therefore, the specifications in all five processes in the standard have reference relationships with each other, making it difficult for developers to identify complicated specification relationships and adhere to the standard.

**Fig. 1** presents an example of goal model extraction from the IEC 62304 standard, where (a) is an excerpt from the standard and (b) is a visualization of the goal model extracted from (a). G1, G2, and G3 in (a) are process, activity, and task specifications, respectively, and G4–G7 are requirement specifications. G9 is a note specification.

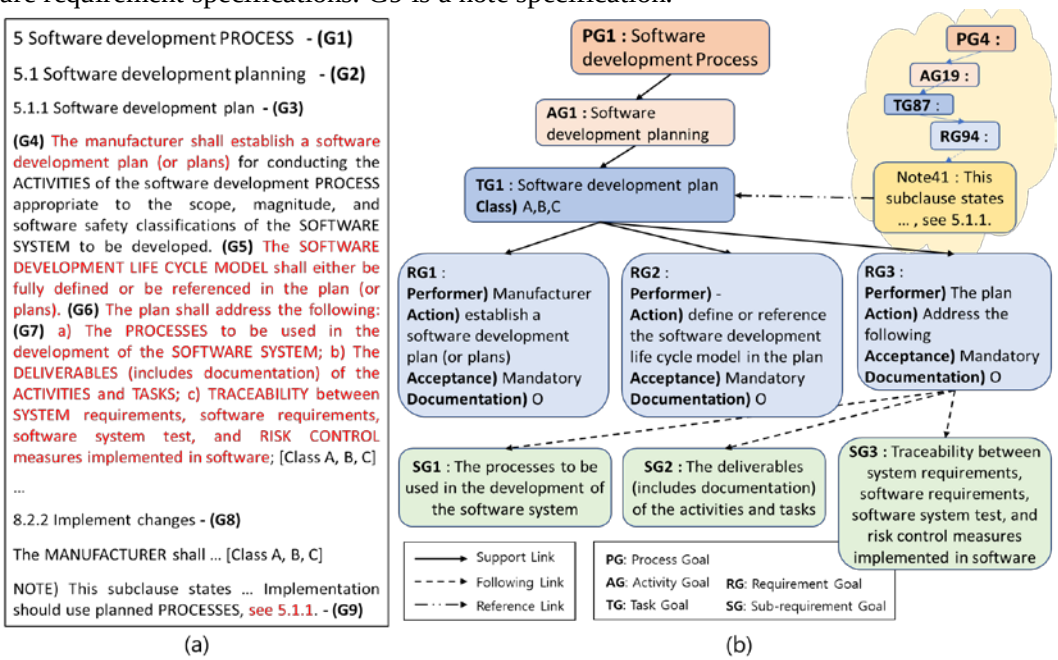


**Fig. 1.** Example of goal model extraction from IEC 62304

To reduce the difficulty that medical device software developers face in complying with IEC 62304, we developed a method for providing the contents of this standard in the form of a goal model. Providing standard content as a goal model allows developers to understand the vast content of a standard and the important information contained within each sentence intuitively. A goal model, sometimes referred to as goal-structuring notation(GSN), is a core methodology in the field of requirements engineering that facilitates system design during early development by allowing developers to understand, analyze, and organize software requirements [3]. A goal model can define all the information required for a system from its business purposes to a software unit implementation plan as goals. All the relationships between goals can be expressed as links. To design such a goal model, we extracted the specifications from every statement in the IEC 62304 standard, from process-level abstract specifications to software-unit-level concrete implementation specifications, as goal items. Additionally, all associations between standard specifications were extracted as link items. The resulting goal model effectively presents the important information from the standard to software developers.

**Fig. 1** presents a basic example of this process. It shows how the standard document in (a) can be transformed into the goal model in (b). However, to transform a standard into a goal model, the following points should be considered. First, a standard specification consists of a hierarchical structure in which content is formatted as “Process-Activity-Task-Requirement.” Because these specifications have different representations for each type, such as G1, G2, G3, G4, and G9 in **Fig. 1(a)**, standard text should be classified according to the specification type. Second, the core information of a specification is masked by the various expressions that decorate the sentence. Based on the nature of the medical software standard, each specification

contains many important considerations for licensing, such as documentation requirement information, required compliance levels, safety-level information, and reference requirements, which must be recognized. Third, all specifications are semi-structured text because the forms of the clauses or phrases used in all sentences in the standard are limited. To transform the standard into a high-quality goal model, as shown in **Fig. 1(b)**, we propose a rule-based goal model extraction method considering all of the points above. Based on analysis of the syntax and grammar of the standard, we define a rule that can recognize key information in the standard hierarchy and sentences. We then extract important information required by the medical software standard without any omissions. In Section 2, we discuss research related to medical software standards and the study of goal model extraction. Previous methods are compared to the method proposed in this paper. Section 3 analyzes the configuration and grammar of standards and specifications. A specification model then is created for each type of specification based on syntactic analysis. We define an extraction rule that generates goals and links from specification model elements. In Section 4, we discuss a process for medical device software development in which a standard document is transformed into a goal model by using the method proposed in this paper. The completion of each step is confirmed through practical examples. In Chapter 5, we analyze failure cases in the results of goal model extraction for IEC 62304 and apply the proposed method to similar standards called ISO 14971 and ISO/IEC 60601-1 to determine its general applicability. This study aims to assist medical device software developers by providing more intuitive and efficient access to the contents of complicated standards by extracting goal models from such standards.

## 2. Related Work

The study of software engineering R&D (i.e., testing, defect management, and configuration management) is a very active field. There have been several studies on compliance with the requirements of medical device software standards [4,5]. However, research analyzing individual standard specifications has been relatively sparse. Therefore, many manufacturers are experiencing difficulty with standard-based licensing procedures. Several recent studies have attempted to assess the quality of standard documents and generate more usable information from such documents [6-9]. The authors of [6,7] identified only the requirements that must be documented among the standard specifications in IEC 62304, which is a standard for medical device software, and ISO/IEC 12207, which is a standard for general software, and provided these requirements to developers in a condensed format called the R&D documentation guidelines. However, because they focused only on documentation and did not consider the flow of processes, their results are limited in terms of assisting R&D to proceed smoothly through various processes. In [8,9], the authors proposed the software process improvement and capability determination (SPICE), and medical device SPICE (MEevSPICE) to evaluate the R&D capabilities of general software and medical device software, respectively. They evaluated the capabilities of R&D processes by comparing software content to specification content based on two standards, namely ISO/IEC 12207 and IEC 62304. However, because these methods only compare standard requirements to software content after software development has been completed, they cannot be used to assist developers in conforming to standards during the software development process.

To simplify specification-based documents, such as software requirement specifications (SRS), various researchers have attempted to extract specifications as goal models or other similar models [10–13]. The authors of these studies modified specifications by using natural language processing and text mining techniques. Thousands of specifications can be automatically extracted as goals by using models trained on big data. However, because such models are trained on general-purpose specifications, they cannot be used to extract the specific expressions of medical software standards as goal models. Additionally, in case of the SRS used by such models, thousands of specification data are available for training. These models cannot be implemented using small numbers of training data. Some other studies have introduced the concept of rules, rather than training methods, to extract the specific information mentioned in the specifications of a specific field as goals [14–16]. This approach has the advantage of representing domain-specific information as goals based on extraction rules. However, the documents used in these studies had very small numbers of requirements and simple structures compared to IEC 62304. Therefore, previous methods are insufficient for extracting the large and complex contents of IEC 62304 as goals and links. To overcome these limitations, we propose a novel method based on a few previous papers that have proposed methods for extracting standard specifications as goals and links based on rules for constructing helpful models [17–19]. These previous methods have limitations in that they cannot reflect the characteristics of medical software based on general software documents. Therefore, we define a new extraction rule that reflects the characteristics of IEC 62304 and construct a goal model based on extracted goals and links.

## 3. Goal Extraction from IEC 62304

This chapter defines the rules for extracting standard specifications as goals and links. First, to provide a basic understanding of standard syntax and its usage in specifications, Section 3.1 discusses the standard configuration and specification types. Section 3.2 defines the concept of specification models. Various models must be defined because previously classified specification types contain different syntax. A model decomposes specification sentences into grammatical elements, identifies the role of each grammatical element, and determines the different meanings of particular words. Sections 3.3 and 3.4 define a rule for extracting goals and links for each specification type.

### 3.1 Configuration of IEC 62304

IEC 62304 is a document containing specifications for the lifecycle process of medical software. Specifications consist of hierarchies of “Process-Activity-Task-Requirement.” There is also a specification type for notes. All specific instructions that a developer must follow are described as requirements. If any additional descriptions are necessary, they are provided in the form of note specifications. Processes, activities, and tasks are given only brief titles in the form of noun phrases to identify which flow they belong to within the standard. For example, there are "software development planning" and "risk management planning" process specifications. **Table 1** lists each type of standard specification in the form of a hierarchy, the number of specifications in each layer, and specification examples from each layer in the actual document. Processes and specifications are divided into five categories: development, risk management, configuration management, problem resolution management, and maintenance. Hundreds of activities, task, requirement, and note specifications belong to each process specification.

**Table 1.** Configurations and Numbers of Expressions in IEC 62304

| Type of Specification | # of Items | Example |
|---|---|---|
| Process | 5 | (See Fig. 1(a) G1) |
| Activity | 26 | (See Fig. 1(a) G2) |
| Task | 124 | (See Fig. 1(a) G3) |
| Requirement | 230 | (See Fig. 1(a) G4–G7) |
| Note | 47 | (See Fig. 1(a) G9) |

### 3.2 Specification Model

Standard specifications are classified into the five types described above. Each of the types have corresponding syntax expressions. These expressions are defined by the specification model and can be represented using the schematic in **Fig. 2**. Such expressions are used as reference units for the goal model extraction rule. When examining the actual contents of the standard document shown in **Fig. 1(a)**, one can see that process, activity, and task types, such as specifications G1–G3, are represented by relatively short noun phrases. These are simple elements with only indexes and noun phrases as model elements, as shown in **Fig. 2**. However, requirement specifications, such as G4–G7, have the form of full sentences in either passive or active voice. The specification model is also divided into different forms and the syntax is divided into grammatical elements to define the goal extraction rule. This process, which has been referred to as "functional grammar" in recent rule-based natural language processing research, systematically analyzes the meaning and role of each component in a sentence [20]. A description of the functional grammar elements for each specification type is provided below.

**Fig. 2** presents a diagram of the specification model. Specifications are divided into five types. Each type has a different detail model because the syntax for each type is different. The lowest-order items are syntactic elements from sentences or phrases that are combined into grammatical elements. In the note specification, the number of expressions is too large, so notes are simply expressed as phrases.

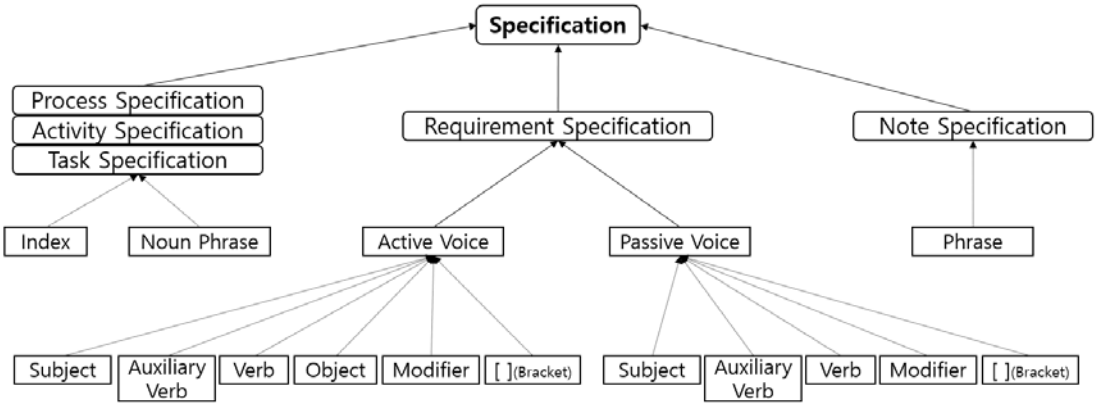


**Fig. 2.** Diagram of the specification model

① **Process, Activity, and Task Specifications**

- **Index**: These three types of specifications are written in the form of "(Index). (Title)." G1–G3 in **Fig. 1(a)** are examples of these specification types. Indexes are divided into numbering levels according to each specification type. A process is denoted as “X.,” an activity is denoted as “X.X.,” and a task is denoted as “X.X.X.” This format allows the hierarchical relationships between specifications to be understood. For example, all activity specifications in the “5.X” group are included in "5. Software Development Process." Index contents are extracted as links between goals when extracting a goal model.
- **Noun Phrase**: This element is the descriptive part of each process, activity, and task specification. Specific guidelines for R&D are provided in requirement specifications. Processes, activities, and tasks simply summarize the activities that should be performed in the form of noun phrases, which helps developers to identify the main flow of R&D. These specification types are not defined as extraction rules for the goal model because their descriptions have simple forms and short corresponding noun phrases. The method for extracting hierarchical and inclusive relationships between specifications based on index levels is discussed in Section 3.3.

② **Requirement Specification (Active Voice)**

- **Subject**: This identifies the subject who must perform the activity proposed by the requirement. The device manufacturer must determine who will handle various requirements. For example, some requirements specify the subject generically as the manufacturer, whereas others require the subject to be limited to the software developer. In the case of **Fig. 1(a)**, G4 instructs the manufacturer to write a software development plan. In this case, the subject is the manufacturer.
- **Auxiliary verb**: In contrast to their usage in general sentences, these elements have special meanings in IEC 62304. These elements indicate whether a requirement is mandatory for licensing or simply a recommendation that may be performed by a user optionally. They may also suggest activities are simply possible for the user. “Shall” implies a mandatory requirement, “should” implies a recommended requirement, and “may” implies a possible activity.
- **Verb**: This refers to the action that the subject should perform and is interpreted as a set with the object to be acted on. However, it is handled separately when extracted as part of a goal. Based on the nature of the standard specification, verbs also have special meanings. This is because a verb tells the user if a record of a requirement is documented. This is crucial information for licensing the standard to a developer and is dependent on a few verbs, such as “document,” “establish,” “define,” and “reference.” Recognizing these verbs is an important task for goal extraction.
- **Object**: This is the object of the verb and is interpreted as a set with the verb. When extracted with a goal, the object is treated as a separate element from the verb. Objects have no special meaning within the standard, but there are certain special expressions. For example, for G6 and G7 in **Fig. 1(a)**, the object is divided into several instances instead of being extracted as a single object. For many of these objects, the corresponding sentences are long. The goal model presented in this study does not recognize a), b), or c) within G7 as complete goals for the sake of efficiency. Instead, it presents them as three separate sub-requirement goals.
- **Modifier**: This is an additional element within the basic sentence syntax. It provides information that can be referred to from within the main contents of the requirement. A goal model only aims to summarize the core information of a standard and provide

it to developers. Therefore, most modifiers for requirements are deleted in the goal extraction process. However, there are cases where it is necessary to recognize a modifier as important information based on the expression of a standard specification, such as the prepositional phrase “in accordance with A.” This phrase implies that requirement A must precede the implementation of the target requirement. Therefore, in the goal model, the goal generated from the target requirement and the goal generated from requirement A are linked to assist the developer.

- **[ ] (Bracket):** The content of the final requirement belonging to a task is always denoted as "[Class ~]". This refers to the safety level of the medical device software. This parameter exists only in the field of medical device software based on the characteristics of the standard. In the medical device software standard, the safety level of a product is categorized as A, B, or C. The tasks to be accomplished differ according to each class. In other words, the developer must determine if they have complied with each task based on this bracketed information.

③ **Requirement Specification (Passive Voice)**

- **Subject**: An example of a requirement in passive form is a sentence such as “the report shall be documented” within the content of the standard itself. Most passive requirements stipulate that any documents or schedules related to R&D should be prepared and planned as shown in the previous example. The expression of the manufacturer or developer, which is the subject of the activity itself, is omitted. The subject is a word that represents the outcome of the requirement activity.
- **Auxiliary verb**: This is the same as an auxiliary verb in the active mode. In the passive mode, it indicates whether the subject must be generated necessarily or can be generated selectively at the user's discretion. “Shall” indicates a mandatory requirement, “should” indicates a recommended requirement, and “may” represents a possible requirement.
- **Verb:** The verb indicates the activity through which the output of the requirement in the subject should be generated. Similar to the verb usage in active requirements, it is important to document these requirements for standard users. Therefore, expressions such as “document,” “establish,” “define,” and “reference” are used as important information in the goal extraction process.
- **Modifier**: This is the same as the modifier used in the active mode. The additional information necessary for fulfilling a requirement is described in the modifier. In particular, the expression “in accordance with A” is used in the standard to indicate an association in which one requirement refers to another requirement. Such associations are used as important information for generating links representing reference relationships between goals in the goal model.
- **[ ] (Bracket):** This is the same as the active mode. Developers must comply only with the tasks that correspond to the safety level of the software.

④ **Note Specification**

- This is a specification for providing additional information to standard users as a sub-element of a requirement or task specification. The sentences in a note specification are expressed in the form of models containing many numbers. However, such notes are mainly used to present examples and best practices. Because the goal model aims to extract the core contents of requirements, the contents of note specifications that are not important are deleted in the extraction process. Therefore, we do not need to define the various syntaxes of such specifications as models. However, there are cases in which

there is special meaning in the nature of the standard, such as when a developer is instructed to refer to another part of the standard for compliance with a requirement. Such cases are denoted by "See ~," as shown in G9 in **Fig. 1(a)**. A relationship between two requirements is denoted by a modifier as a reference relationship. In contrast, a note indicates a broader scope of reference, such as references between requirements and activities, requirements and tasks, or requirements and other standards. This information is used as important link information between goals in the goal model.

### 3.3 Goal Extraction Rule

The specifications in the standard are divided into five types, as shown in **Fig. 3** in Section 3.1. Specifications are expressed differently for each type. All specifications in the standard are extracted as goals, which are defined separately for each type based on the corresponding syntax structure. **Fig. 3** summarizes the relationships between the standard specifications and goal model. These relationships determine how the detailed elements of specifications are transformed when an arbitrary specification is extracted as a goal or link. The elements of the standard excerpt shown on the left are the same as those defined in Section 3.2.

The goal model on the right is largely divided into goal and link concepts. Goals contain only the core information from the specifications described by the sentences in the standard and provide these concepts in itemized form. Goals are divided into six categories and contain sub-requirements, which are not defined in the standard specification, but are added to subdivide requirement specifications. Links express the relationships between goals and represent the hierarchies of specifications, reference relationships, etc. The detailed rules for each type of goal and link are defined below.

① **Process, Activity, and Task Specifications**

- Process, activity, and task specifications contain the index and noun phrase elements shown in **Fig. 3**. Noun phrase information is extracted into goal description items. These types of specifications represent important information because they provide the contents of activities as titles in the form of noun phrases. Therefore, they are extracted as descriptive items without any modification. An index is not simply a reference number but is used as important link information because it represents the hierarchical relationships (inclusion relationships) between process, activity, and task. The process of extracting and linking is described in detail in Section 3.4.

② **Requirement Specification**

- Regarding the standard requirement type, requirements are decomposed into various syntax elements, similar to the requirement model discussed in Section 3.2. Each element is extracted as a detailed element of the corresponding goal according to its role and meaning. **Fig. 4** presents the rule for extracting goals and links for active requirements. In the case of auxiliary verbs, the words "mandatory," "recommend," and "available" in the "acceptance" section of the "requirement goal" indicate compliance with the associated requirements. Goals itemize and display information such that developers can intuitively determine compliance. Additionally, a verb defines the corresponding goal's action in its basic usage, but is also used to indicate whether or not a goal is documented according to the use of a specific word. If words such as "document," "establish," or "record" are used, then the documentation item of a requirement goal should be marked as "O." Additionally, objects are not only used to define actions, but also to define sub-goals according to the usage of specific expressions. In the case of modifiers, if the expression "in conforming with" is used,

it is extracted as link information because it represents a reference relationship between requirements. Because all other expressions are simply supporting content for requirements, they are deleted without treating them as important information.

③ **Note Specification**

- This is a specification for providing additional information to the user. Therefore, content other than specific expressions is excluded. References to requirements and other sections are expressed in the form of "See ~." Because these expressions represent important link information between goals in the goal model, only the corresponding expressions are recognized and extracted as links.

**Fig. 3** presents the association between the standard and goal model in the context of model elements. In (a), the elements of the standard are the same as those analyzed in Section 3.2. Parts (b) and (c) present elements of the goal model. "Descriptions" contain all information that does not require any modification. "Classes" represent the safety levels of medical device software. A "performer" is the subject of a requirement. An "action" represents both the execution and result of a requirement. "Acceptance" represents the required level of compliance. "Documentation" implies the need to record the fulfillment of a requirement. Links are divided into three categories, which are described in Section 3.4.

**Fig. 4** presents the rule for extracting goals from requirement specifications. "X" in the syntactic element is a variable representing a word used in the actual specification that is transformed into a goal by the rule.

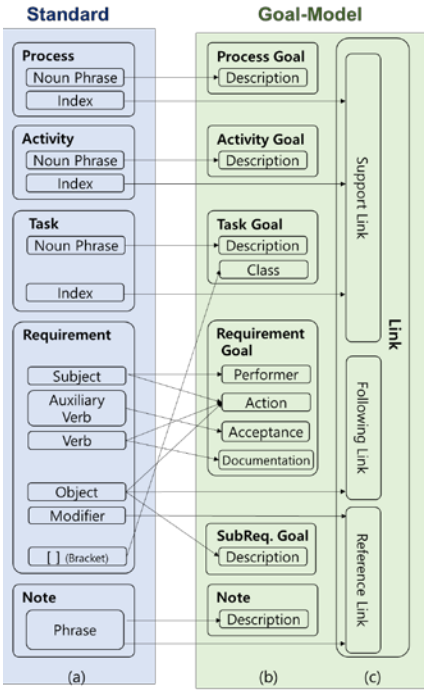


**Fig. 3.** Association between the standard and goal model

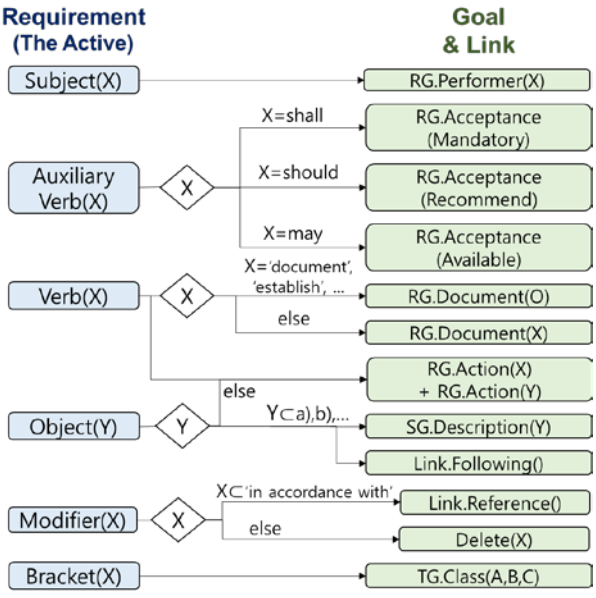


**Fig. 4.** Rule for extracting goals from requirement

### 3.4 Link Extraction Rule

While goals present only the core information from the standard to users, links intuitively present the reference associations between specifications to developers. In other words, they refer to the reference relationships between goals. Link information is also important information that should not be ignored. If the goal extraction rule in Section 3.3 represents the conversion of standard specifications into goals, then the link extraction rule in this section represents the generation of linking relationships between goals extracted from specifications that share reference relationships. The link extraction rule is summarized in **Figs. 3(a–c)**. There are three different types of links. Support links represent the hierarchical relationships between goals in the context of processes. Based on the index information in the corresponding specifications, such links express hierarchical relationships numerically and intuitively. Following links represent the relationships between requirements and sub-requirements. Reference links are used to represent broad reference relationships between specifications within a standard document or between specifications and other resources.

**Fig. 5** presents an example in which the rule above is actually applied. Developers can more effectively understand the relationships between standard specifications based on extracted links compared to the original text.

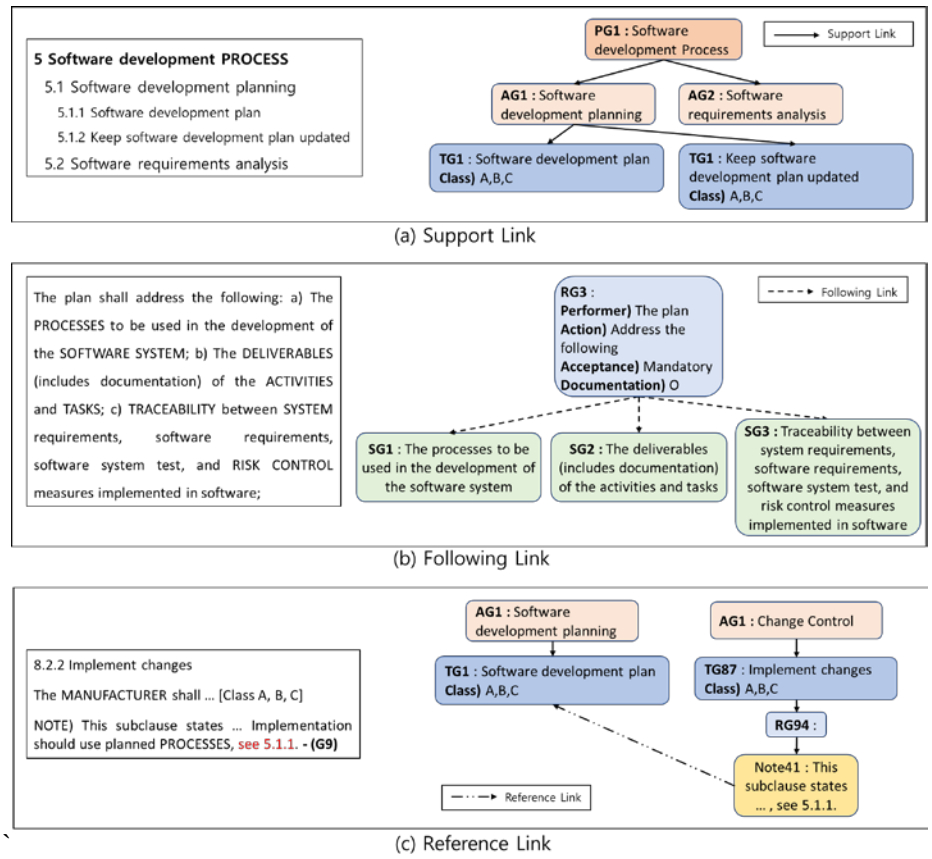


**Fig. 5.** Examples of link extraction for each type of link

## 4. Constructing a Goal-Model

This chapter defines the process through which a developer can create a goal model using the goal and link extraction rules. Each major step is illustrated using real documentation. The goal model generation process proposed in this study is illustrated in **Fig. 6**. When a user inputs a standard document, such as that shown in **Fig. 1(a)**, the text is processed by the goal and link extraction rules defined in Chapter 3. As a result, the goal model shown in **Fig. 1(b)** can be obtained. This goal model allows the user to understand the contents of the standard intuitively and follow the standard systematically. The goal model creation process is divided into four main stages. Details and examples are provided below.

**Fig. 6** illustrates the process for constructing a goal model. The framework proposed in this paper begins with standard input by the user and proceeds through the goal model generation process.

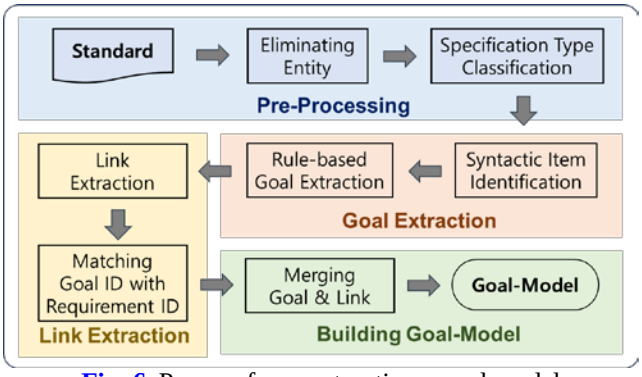


**Fig. 6.** Process for constructing a goal model

### 4.1 Preprocessing

Most standard documents, including IEC 62304, are provided in .pdf format. When a user inputs a document in .pdf format, preprocessing must be performed prior to goal extraction. First, the system removes all visual objects, such as pictures, tables, and blank spaces, and extracts only the remaining text. The system then separates the text into individual specification units and assigns each unit an ID. Finally, it recognizes the syntax of all specifications and classifies the specifications into different five types.

### 4.2 Goal Extraction

Classified specifications are separated into syntactic elements by a parser. Syntactic items are then extracted using the extraction rules defined in Chapter 3. Syntactic items exist in the structure of the specification model described in Section 3.2. They are extracted with the values corresponding to goal items according to the extraction rules. An example of an active requirement specification being extracted as a goal by the parser is provided below.

***Example 1-1)*** Requirement G4 (**Fig. 1(a)**) is split as "*Subject* (the manufacturer) + *AuxiliaryVerb* (shall) + *Verb* (establish) + *Object* (a software development plan (or plans)) + *Modifier* (for conducting the…developed)"
***Example 1-2)*** The goals from G4 (**Fig. 1(a)**) are extracted as "*DG.Performer* (manufacturer) + *DG.Accept* (Mandatory) + *DG.Action* ("establish" + "a software development plan (or plans)") + *DG.Document* (O) + *Delete* (for conducting…)"

### 4.3 Link Extraction

In addition to being extracted as goals, standard content is also extracted as links between goals. In addition to support links, which represent the vertical hierarchy between goals in the general goal model, it is important to extract following links and reference links from requirements based on the nature of the standard. These links represent important information specified by the standard. However, the link information extracted at this time is not given in the format of specification IDs and goal IDs, such as "See 5.1" or "~ in accordance with Clause 5," for internal processing. Therefore, link information and specification IDs must be matched. Because specifications are transformed into goals, the corresponding goal IDs must be matched. Goals and links can be merged following synchronization processes inside the proposed framework. An example of extracting a note specification as a link using the parser is provided below.

***Example 2-1)*** Note G9 (**Fig. 1(a)**) is identified as "*Sentence* (This subclause as … processes) + *Expression* (see 5.1.1.)"
***Example 2-2)*** The linkage in G9 (**Fig. 1(a)**) is extracted as "*Link.Reference* (To "TG1" from "Note41")"

### 4.4 Building the Goal Model

After the text is extracted from a standard document, specifications are classified, and IDs and types are assigned. Each specification is then extracted as a goal or link according to the extraction rules. Synchronization between the goal IDs and link IDs is then performed based on the specification IDs. Finally, one can connect goals and links to finalize the goal model. In the example in **Fig. 1**, the user enters the standard document in (a) and obtains the goal model in (b) as an output. Rules based on the construction of medical standards allow the important information from standards to be expressed effectively and intuitively. The resulting goal model can be visualized using various methods. In this study, we used Protégé, which is an ontology language visualization tool.

**Fig. 7** presents a visualization of a goal model. The boxes represent goals and the blue arrows represent support links. In Protégé, the user can expand only the desired part of the goal model by pressing the "+" button on a goal.

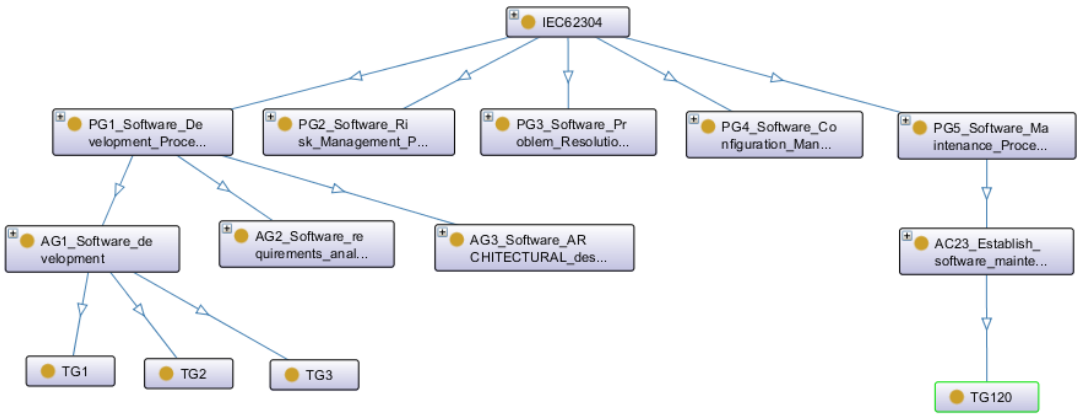


**Fig. 7.** Example of goal model visualization

## 5. Discussion

In this chapter, we analyze the accuracy of the proposed goal model extraction method by comparing it to a conventional goal model generation method. We also analyze failure cases to improve the proposed method. When a software manufacturer decides to go through the process of product development using a goal model, in most cases, a small number of project managers, quality managers, or test managers will manually define SRS content as a goal model. This paper proposed a method to overcome the limitations of existing handcrafted models when a medical device software manufacturer wishes to follow a product R&D process in conformity with standards by implementing a goal model. A developer is not required to define a goal model directly by analyzing a standard. Instead, the proposed method automatically generates a goal model based on extraction rules. To verify the accuracy of the proposed method, we consider a goal model that was manually extracted by a standard expert as an existing method and compare it to the goal model generated by the proposed method. Additionally, while this study considered only the IEC 62304 standard, which is a medical device software standard, other standards in the same field have very similar structures and expressions. Therefore, to determine the feasibility of applying the proposed method to other standards, we analyzed the results of extraction from similar standards. The IEC 60601-1 standard is a standard for medical device safety. We considered Section 14 of this standard, which describes software safety. ISO 14971 is a standard for medical device software risk management that is often used in conjunction with the IEC 62304 standard. **Table 2** lists the results of comparing the rule-based goal model extracted by the proposed method to the manually generated goal model for three medical device standards. According to the results, in the case of the IEC 62304 standard, the goal model extraction method proposed in this paper achieved an accuracy of 97%, which is relatively good. This accuracy was achieved by analyzing the expressions in all specifications of the standard and defining appropriate extraction rules. A typical failure case is provided below in Example 3. Regarding expressions that represent reference associations, the references refer to external standards, rather than IEC 62304 internals.

***Example 3)*** "The manufacturer shall document potential causes of the software item contributing to a hazardous situation in the risk management file (see ISO 14971)."

However, the accuracy for the IEC 60601-1 standard is 71.4% and that for the ISO 14971 standard is 81.9%. Most specifications were successfully transformed, but a meaningful number of specifications were not. The reason for this poor performance is that these two standards contain some different expressions compared to IEC 62304. For example, the two other standards do not use the expression "[Class A, B]" to indicate safety ratings. To overcome this limitation, it is necessary to define more comprehensive goal extraction rules. If one were to define such rules by analyzing the specifications of all standards in the medical device software field, it would result in a very complete model for all standards used by medical device software manufacturers.

**Table 2.** Accuracy of goal model generation

| Standard | Total Number of Goals | Number of Matches | Accuracy (%) |
|---|---|---|---|
| IEC 62304 | 484 | 470 | 97.1% |
| IEC 60601-1-14 | 112 | 80 | 71.4% |
| ISO 14971 | 242 | 198 | 81.9% |

## 6. Conclusion

This paper proposed a method to help medical device software manufacturers comply with IEC 62034, which is used for licensing, and conduct R&D. Medical device software developers have traditionally worked with standard experts to manually generate a document with over 80 pages of explanatory text to define a goal model. This paper proposed a method to automate the construction of a goal model by using rule-based extraction. Because a standard specification contains domain-specific information, there are reference associations between items that are presented in the form of unformatted sentences. A rule that can cover all of these expressions must be defined. To define such a rule, the standard configuration and all expression usages were analyzed. The standard specifications were then divided into five types for systematic rule definition and an extraction model was defined based on syntactic analysis. Goals and links were extracted according to the extraction rules based on syntactic item units. Different rules were defined individually according to the specification type. To validate the method proposed in this paper as a development framework, each step of the process in which a developer inputs a standard document and the resulting goal model were checked for the case of IEC 62304. We also verified the accuracy of the goal model generated by the proposed method by comparing it to a manually generated goal model and discussed potential improvements based on failure cases. The proposed method can be applied not only to the IEC 62304 standard, but also to other software and medical device standards. However, these additional standards must be considered during the rule definition process. The goal model extracted by the proposed method presents all the information contained in IEC 62304. In addition to understanding standard content based on the goal model, it is possible to understand auxiliary information, such as the safety level of a medical device. The goal model also presents relevant information that references additional requirements in an intuitive format.

## References

[1] Imagawa, Kuniki, Yoshiaki Mizukami, and Seiko Miyazaki, “Regulatory convergence of medical devices: a case study using ISO and IEC standards,” *Expert review of medical devices*, Vol 15, No.7, pp.497-504, 2018. Article (CrossRef Link)

[2] IEC, *IEC 62304 - Medical Device Software Life-Cycle Processes, IEC*, Geneva, 2015.

[3] GSN Working Group, *GSN COMMUNITY STANDARD, GSN*, UK, 2011.

[4] Stirbu, Vlad, and Tommi Mikkonen, “Towards Agile Yet Regulatory-Compliant Development of Medical Software,” in *Proc. of 2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). IEEE*, 2018. Article (CrossRef Link)

[5] Laukkarinen, Teemu, Kati Kuusinen, and Tommi Mikkonen, “Regulated software meets DevOps,” *Information and Software Technology*, Vol 97, pp 176-178, May, 2018 Article (CrossRef Link)

[6] DongYeop Kim, Ye-Seul Park, Byungjeong Lee, Jung-Won Lee, “Methods of Extracting and Providing R&D Documentation Guideline for Licensing Medical Device Software,” in *Proc. of KSII The 13th Asia Pacific International Conference on Information Science and Technology(APIC-IST)*, pp.1-3, Jung 24-27, 2018.

[7] DongYeop Kim, Byungjeong Lee, Jung-Won Lee, “Methods for Providing and Evaluating Software R&D Documentation Guideline based on Extracting Requirements of ISO/IEC 12207,” *KSII 2018*, Vol 19, No.1, pp 1-2, April 28, 2018

[8] ISO and IEC, *ISO/IEC 15504-4:2004, Information technology — Process assessment — Part 4: Guidance on use for process improvement and process capability determination*, ISO and IEC, Geneva, 2004.

[9] Lepmets, M., Clarke, P., McCaery, F., Finnegan, A., & Dorling, A., “Development of MDevSPICE the medical device software process assessment framework,” *Journal of Software: Evolution and Process*, 27(8), 565-572, 2015. Article (CrossRef Link)
[10] Cleland-Huang, Jane, et al., "Automated classification of non-functional requirements," *Requirements Engineering*, Vol 12. No.2, pp. 103-120, 2007. Article (CrossRef Link)
[11] Weber-Jahnke, Jens H., and Adeniyi Onabajo, "Mining and analysing security goal models in health information systems," in *Proc. of the 2009 ICSE Workshop on Software Engineering in Health Care. IEEE Computer Society*, 2009. Article (CrossRef Link)
[12] Ghosh, Shalini, et al., "Automatically extracting requirements specifications from natural language," *CoRR, abs/1403.3142*, 2014.
[13] Fayoumi, Amjad, Evangelia Kavakli, and Pericles Loucopoulos, "Towards a unified meta-model for goal oriented modelling," in *Proc. of the 12th European, Mediterranean & Middle Eastern Conference on Information Systems (EMCIS)*, 2015.
[14] Acher, Mathieu, et al., "On extracting feature models from product descriptions," in *Proc of the Sixth International Workshop on Variability Modeling of Software -Intensive Systems. ACM*, pp. 45-54, 2012. Article (CrossRef Link)
[15] Rauf, Rehan, Michal Antkiewicz, and Krzysztof Czarnecki, "Logical structure extraction from software requirements documents," in *Proc. of IEEE 19th International Requirements Engineering Conference*, 2011. Article (CrossRef Link)
[16] Arora, Chetan, et al., "Extracting domain models from natural-language requirements: approach and industrial evaluation," in *Proc. of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems. ACM*, pp. 250-260, 2016. Article (CrossRef Link)
[17] Shimada, Hironori, Hiroyuki Nakagawa, and Tatsuhiro Tsuchiya, "Constructing a Goal Model from Requirements Descriptions Based on Extraction Rules," in *Proc. of Asia Pacific Requirements Engineering Conference. Springer*, pp. 175-188, 2017. Article (CrossRef Link)
[18] Lee, Jonathan, Nien-Lin Xue, and Jong-Yih Kuo, "Structuring requirement specifications with goals," *Information and Software Technology*, Vol 43. No.2, pp. 121-135, 2001. Article (CrossRef Link)
[19] Nguyen, Tuong Huan, John Grundy, and Mohamed Almorsy, "Rule-based extraction of goal-use case models from text," in *Proc. of the 2015 10th Joint Meeting on Foundations of Software Engineering. ACM*, pp. 591-601, 2015. Article (CrossRef Link)
[20] Halliday, Michael Alexander Kirkwood, Christian Matthiessen, and Michael Halliday, *An introduction to functional grammar*, Routledge, 2014.

**DongYeop Kim,** He received the B.S. degrees in Department of Electrical and Computer Engineering from Ajou University in 2018. He is in a Master's course from Department of Electrical and Computer Engineering from Ajou University. His research areas include medical device software.

**Byungjeong Lee**, He received the B.S., M.S., and Ph.D. degrees in Computer Science from Seoul National University in 1990, 1998, and 2002, respectively. He was a researcher of Hyundai Electronics, Corp. from 1990 to 1998. Currently, he is a professor of the Department of Computer Science and Engineering at the University of Seoul, Korea. His research areas include software engineering and web science.

**Jung-Won Lee,** is a professor of the Department of Electrical and Computer Engineering at Ajou University, Korea. She received her PhD. Degree in Computer Science and Engineering from Ewha Womans University, Korea, in 2003. She was a researcher of LG Electronics and did an internship in the IBM Almaden Research Center, USA. Her areas of research include context-aware, embedded software and software engineering.