Conditional Preferences in Software Stakeholders’ Judgments

Ivana Ognjanović, Dragan Gasevic, Ebrahim Bagheri, Mohsen Asadi

ABSTRACT

In reality, many of the stakeholders’ decisions about their desirable requirements can be dependent on other internal or external factors. Such dependencies entail conditionality between the requirements that have been defined, e.g., a requirement is desirable for the stakeholders only if a certain condition is met or some other requirements are excluded. In this paper, we propose a novel framework that tackles the challenge of capturing and processing software stakeholders’ conditional preferences. Our proposal extends the Stratified Analytic Hierarchical Process (S-AHP) method that we have previously introduced. S-AHP is built on top of the Analytic Hierarchical Process method, which performs a pair-wise comparison of stakeholders’ preferences. The current main framework for handling conditionality is TCP-nets, which suffers from the inability to handle hierarchical structure of comparisons and cycles in dependencies defined by the conditional requirements. Also, TCP-nets is primarily developed for qualitative preferences and its quantitative extensions cannot completely capture quantitative relative importance. We show that our framework is able to address these shortcomings of TCP-nets while preserving many of its advantages.

1. INTRODUCTION

Evidence suggests that the successful delivery of software engineering projects is directly related to the way in which stakeholders are involved in the decision making process within the various stages of the software development process [1]. It is important that the stakeholders’ judgments and opinions are taken into account and properly addressed when critical decisions need to be made. For instance, within the context of the software development process, understanding the priorities and objectives of the involved stakeholders of a target application is essential for choosing the best software features that need to be included in the product that is going to be developed [2]. This is because a software application is only successful if its stakeholders and end users are satisfied with it. Hence, software engineers need to constantly keep in touch with the stakeholders in order to understand and operationalize their interests, goals and preferences.

For these reasons, different researchers have been interested in developing tools and techniques for eliciting, formalizing and interpreting stakeholders’ priorities over the existing software requirements, features or options, e.g., the pair-wise comparison method, priority groups, and cumulative ratings [12]. These techniques generally aim at creating a relative ranking between the available options in a software system based on the interests of the stakeholders. For instance, one might conclude that given the interests of a group of stakeholders, security is more important than performance and accessibility, which should therefore receive more attention while designing the software system. This approach to representing stakeholders’ priorities is intuitive and easy to understand, but fails to capture an important trait of prioritization: conditionality.

Conditional preferences refer to cases where available options do not always have a unique priority relationship between them [13]. For instance, the stakeholders might consider security to be more important than accessibility if the application is running on the Internet otherwise if the application is deployed on a local LAN then accessibility is more critical for them than security. So, as we can see the priority relation between the available options is dependent on certain conditions, which is what we call conditional preferences. Traditional forms of dealing with preferences in the area of software and requirements engineering lack the capability for addressing these kinds of conditional preferences.

As we will see in the rest of this paper, in real world software decision making scenarios, the stakeholders do not have a clear-cut and well understood preference over all of the available options [5]. Their preferences change based on circumstances. Software engineers have encountered many situations where they have been able to persuade the stakeholders to exclude some very desirable feature (e.g., f1) from the final software with the agreement that some other features (e.g., f2 and f3) will be included in exchange. This case is essentially equivalent to saying that the stakeholders highly value the inclusion of f1 but if f2 and f3 are simultaneously present in the final application their degree of preference over f1 reduces, which is an example of conditionality. In such conditional circumstance, the priority and desirability of the combination of f2 and f3 increases to more than that of f1. These cases are prevalent when interacting with software stakeholders or even software designers/architects; however, current software prioritization methods fail to address this important issue, conditionality, which we cover in this paper.

In this paper, we formalize conditional preferences and show how they can be handled by extending our software prioritization method, S-AHP [2]. In principle, S-AHP is based on the Analytic Hierarchy Process (AHP), which is a structured way for handling and evaluating alternatives. It is amongst the simplest yet effective techniques that are widely used for decision making. The S-AHP technique takes AHP one step further by taming the number of required comparisons between the available options through the employment of a stratified approach. In this paper, we introduce how S-AHP can be further completed by the introduction of the concept of conditionality. We will show that the extensions made on S-AHP do not considerably increase the complexity of our work and that Conditional S-AHP is easy and straightforward to use for software engineers and stakeholders in that it does not impose any complex requirements on its users.

As is an important requirement for the area of prioritization, we provide formal definitions of the proposed extension and its key elements in this paper. It should be noted that the formal rigor in the presentation of the work in this paper does not impact the
usability of our approach. In reality and actual usage, the definitions provided in the paper are incorporated in our tool support for S-AHP (being transparent to the users); therefore, the users will use the approach quite simply without being concerned with any of the technical details of the approach as was practiced for the tool support for S-AHP in [2].

To extend S-AHP with conditionality an important concern needs to be considered, namely: in many cases, some of the priorities defined by the stakeholders are in the form of standard relative priorities (a is more important than b), but some others are in the form of conditional relative priorities (a is more important than b if c holds, otherwise, b is more important). Therefore, it is important for the prioritization framework to be able to simultaneously handle both kinds of priorities, i.e., standard relative priorities and also conditional relative priorities.

It is important to notice that conditional preferences are already identified as a research challenge in the area of requirements engineering. One of the most notable works on this topic is TCP-nets [13, 16]. Despite the advantages of TCP-nets, they do not allow for quantitatively comparing each pair of criteria within TCP-nets, i.e., it is possible to compare them only in the form: “if two options have the same final ranking I prefer one which has a higher value of one criterion over the other” as shown in [6]. This form of comparison might be considered to be conditional as well, but it is natural to expect requirement defined as: “In case of high price I prefer good performance but if the performance is not good enough, low price is appropriate.” This requirement will cause a cycle in TCP-net and the prioritization will not be possible with the framework of TCP-nets.

Our proposed Conditional S-AHP provides the following major benefits to the process of prioritization:

1. It provides a unified framework for capturing and handling both conditional and unconditional priorities simultaneously;
2. It allows for the prioritization of the available options by assigning quantitative priority measures;
3. It offers a theoretically sound process for prioritizing between the available options of the ranking process by working through the defined conditional requirements;
4. It is able to recognize the available inconsistencies in the defined conditional and unconditional requirements and effectively handle them through the resolution process.

In the following, we first introduce our Stratified Analytic Hierarchy Process (S-AHP) (Sect. 2). We then show how S-AHP can be extended to address the above important concerns within the context of conditional preferences (Sect. 4 and 5). Throughout the paper we employ a widely used benchmark case study (Sect. 3) from the area of software product lines and related feature modeling techniques to show how our S-AHP can be extended to address conditionality (Sect. 6), but it is clear that our proposed work in this paper is general enough to be applied to any software and requirement prioritization activity.

2. STRATIFIED ANALYTIC HIERARCHY PROCESS
The Stratified Analytic Hierarchy Process (S-AHP) is a prioritization technique that takes the preferences, business goals and high-level objectives of a given group of stakeholders into account in order to find the relative priority and importance of the available software options [2]. In other words, S-AHP helps the stakeholders find the most suitable set of features for their target application by creating a prioritization over all of the available features based on their preferences and objectives.

Essentially, S-AHP is based on the Analytic Hierarchy Process (AHP) [4], which is a pair-wise comparison method used to calculate the relative ranking of different options based on stakeholders’ judgments. In order to use AHP, stakeholders need to first determine the relative importance of each of the available criteria compared to the others. Although AHP is simple to perform, it suffers from several problems, such as quadratic number of comparisons, and inability to compare conceptually dissimilar criteria. In S-AHP [2], we have resolved these issues for feature selection in software design.

Simply stated, S-AHP takes a layered approach to the prioritization of the available options. To do this, high level objectives and goals of the stakeholders are specified and are referred to as concerns. Examples of concerns are implementation costs, development time, security, and sales. Once these concerns are identified, the options that need to be prioritized are interrelated with the concerns. For instance, one of the options to be evaluated is implementation using COTS. For the stakeholders, this option entails insignificant implementation costs, quick development time and low security. However, the other option is in-house development, which entails high implementation costs, time consuming development and high security. The formation of this interrelationship between the concerns and the available options allows S-AHP to create a prioritization over the available options by valuing options that are related to more important concerns higher. In this simple example, if security is more important for the stakeholders, in-house development will receive a higher importance and priority; likewise, if lower implementation costs are more essential, COTS-based development will be more attractive.

Formally defined, suppose that we have a triple (F, C, QT) as the input for S-AHP where F is a set of available options, each of which is annotated with concerns and a set of relevant qualifier tags. Qualifier tags are different possible enumerations for each concern. For instance, the qualifier tags for the security concern can be secure, open, and vulnerable; C is the set of defined concerns; and QT is the set of qualifier tags for the concerns. Concerns and their qualifier tags have a hierarchical structure (which is the basic characteristic of the AHP [4]). Given this triple, the following stages can be performed consecutively to produce a valid prioritization over the available options using S-AHP:

Concern prioritization step. This stage compares the set of concerns (C) using the standard AHP to determine their relative importance. Thus, first the relative importance of each concern with respect to the others is defined by the stakeholders. So, using AHP, the concerns are compared in a pair-wise way, and a relative priority is calculated for each of them. Based on these relative priorities, S-AHP can rank the concerns. The concerns with the highest ranks are then used in the option ranking step. In order to keep one of the basic characteristics of AHP [4] that a sum of priorities at each level must equal one, we choose the reciprocal division, which keeps the previous relative order of concerns.

Options ranking step. In this step, in order to find the actual priority and importance of the available options, the relative impor-

1 In our original work on S-AHP, we experimented with the use of S-AHP for configuration of feature models, which are commonly used for modeling variability in software product lines (SPLs).
tance of the qualifier tags of the most important concerns are computed by performing AHP. Afterwards, the rank of each option is determined based on the rank of the qualifier tags assigned to the concerns attached to each of the options. The goal of this stage is to assign higher ranks to the options which are related to more important concerns from the stakeholders’ viewpoint.

3. RUNNING EXAMPLE
As a running example for this paper, we will provide a case of prioritizing the features available in a software product line feature model. Feature modeling is an important conceptual tool that offers modeling support for software product lines. It provides for addressing commonality and variability both formally and graphically, describing interdependencies of the product family attributes (features) and expressing the permissible variants and configurations of a product family [2]. Given a feature model representation of a software product line, the main task of an application engineer is to understand the priorities of the target stakeholders and create a prioritization over the available features of the product line. The stakeholders’ priorities provide the rationale for why a certain feature in the software is more important than the others and needs to be selected. We will build on this important issue in software product lines to show how our conditional prioritization framework can help in selecting the right features based on the stakeholders’ opinions.

The online store is the widely-accepted benchmark application in the software product line feature modeling literature [2]. In Figure 1, the features of the online store feature model have been annotated with four concerns, namely security, customer ease, international sale and implementation cost. The concerns are qualified using high, medium and low values; therefore, the modelers are able to express their belief with regards to the features in the context of these concerns. The concerns and their qualifier tags are shown on the lower part of the leaf features and the legend explaining their interpretation is placed on the top leaf part of the Figure 1. For instance, using cash as a payment method has been shown on the lower part of the leaf features and the legend indicates that it is inexpensive from an implementation perspective. The information provided in Figure 1 is used throughout our examples in the paper.

Returning back to S-AHP from the previous section, let us first consider the first stage of S-AHP on the example from Figure 1. Suppose that the concerns from Figure 1 (Security, Customer Ease, International Sale and Implementation Cost) have their priorities defined as 0.24, 0.19, 0.16, and 0.40, respectively. Let suppose that stakeholders have chosen Customer Ease and International Sale as less important concerns. Their summed priorities are 0.19+0.16=0.35 and it might be distributed to Security and Implementation Cost reciprocal to their priorities as follows: new priority for Security: 0.24+0.24/(0.24+0.40)*0.35=0.38, and for Implementation Cost: 0.40+0.40/(0.24+0.40)*0.35=0.62. It is important to note that relative importance between Security and Implementation Cost is the same as: 0.24/0.40 ≈ 0.38/0.62. The relation ≈ is used instead of strong equality, because of the calculation with two decimals and rounding of the result.

In the second stage, options (i.e., features) are ranked based on the ranks assigned to each concern; and that is discussed in Sec. 5.

4. CONDITIONAL RELATIVE IMPORTANCE IN S-AHP
As explained in the process in Section 2, ranking is performed in two stages, which creates a stratified (layered) process. As such, S-AHP can tame the quadratic number of comparisons required in AHP and provides means for comparing conceptually dissimilar options. Here, we extend S-AHP to be able to capture and interpret conditional relative importance and priority values.

The intention of the extension of S-AHP is to allow stakeholders to define their own preferences with both relative importance and conditional relative importance statements about the qualifier tags of each concern. Stakeholders’ preferences about relative unconditional importance can be formalized with the relative importance relation, denoted as $\succ$. This relation can be used for representing relative unconditional importance between both concerns and their qualifier tags, as per the following definition.

Definition 1 (Relative importance). Relative importance between concerns (or qualifier tags) $a$ and $b$ is: $a \succ b$ iff concern (or qualifier tag) $a$ is more important than concern (or qualifier tag) $b$ with coefficient $\alpha$. Basic characteristics of this relation are:

Reflexivity: $a \succ a$

Symmetry: $a \succ b \Rightarrow b \succ a$

Transitivity does not hold, which means that if $a \succ b$, $b \succ c$, then we cannot make any conclusion about the relation between concern (or tags) $a$ and $c$.

Traditionally, 1, 3, 5, 7, and 9 are used to represent the degree of importance of different options over each other. They show equality, slight value, strong value, very strong and extreme value, respectively. Therefore, $a \succ b$ would show that concern (tag) $a$ is equally important as concern (tag) $b$ or that the stakeholder is indifferent about the priority between concerns (tags) $a$ and $b$.

Due to the two-level hierarchical structure of concerns and their qualifier tags, matrices for calculating relative rankings with S-AHP are created in two different levels: the level of concerns and the level of qualifier tags. In the first level, global ranks for concerns are calculated and in the second, qualifier tags of each concern are locally ranked. Their global ranks are calculated by multiplication of concern ranks and the local rank of each tag.

The matrices of different levels, i.e., the level of concerns and the level of qualifier tags, can be formally defined as follows:

Definition 2 (Matrix of relative importance between concerns). Let us suppose that $C = \{c_1, \ldots, c_n\}$ is a set of concerns. A matrix of relative importance between concerns according to relation

$$
\begin{array}{cccc}
1 & a & b & c \\
\alpha & \frac{1}{\alpha} & \alpha & \frac{1}{\alpha} \\
\alpha & \frac{1}{\alpha} & 1 & \alpha \\
\frac{1}{\alpha} & \alpha & \frac{1}{\alpha} & 1 \\
\end{array}
$$

Reflexivity: $1 - a$

Symmetry: $a - b \Rightarrow b - a$

Transitivity does not hold, which means that if $a - b \land b - c$, we cannot make any conclusion about the relation between concern (or tags) $a$ and $c$.

Traditionally, 1, 3, 5, 7, and 9 are used to represent the degree of importance of different options over each other. They show equality, slight value, strong value, very strong and extreme value, respectively. Therefore, $a - b$ would show that concern (tag) $a$ is equally important as concern (tag) $b$ or that the stakeholder is indifferent about the priority between concerns (tags) $a$ and $b$.
is defined as $R^\alpha_{\max}(C) = \{R[i, j] = \alpha \mid 1 \leq i, j \leq n, c_i > c_j\}$.

**Definition 3 (Matrix of relative importance for concern $c$).** Let us suppose that concern $c$ is annotated with a set of tags $\mathcal{T} = \{t_1, \ldots, t_m\}$. A matrix of relative importance for concern $c$ according to relation $\succ$ is defined as follows:

$$R^\alpha_{\mathcal{T}}(c, \mathcal{T}) = \{R[i, j] = \alpha \mid 1 \leq i, j \leq |\mathcal{T}|, t_i \succ t_j\}.$$

As the relation $\succ$ is reflexive and $\alpha$-symmetric, matrices $R^\alpha(C)$ and $R^\alpha(c, \mathcal{T})$ satisfy the following conditions:

$R^\alpha(C)[i, i] = 1$, i.e., $R^\alpha(C)[i, i] = 1$, and $R^\alpha(C)[i, j] = R^\alpha(C)[j, i] = 1$; that is, these matrices are uniquely determined with $\{R^\alpha(C)[i, j], i < j\}$. Accordingly, for cases when we have a set of three concerns $\{c_1, c_2, c_3\}$, its matrix of relative importance for this is uniquely determined with $\{a_0 > c_2, a_0 > c_1, c_1 > a_0 > c_3\}$. The unique determination of the matrix for one concern (level 2) is defined similarly.

The formal notation used for defining the relative importance is necessary because of the simplicity that it will provide for the specification of the process in Section 5. Also, Definitions 2 and 3 are sufficient for the specification of the 5-AHP algorithm: First, for the set of concerns $C = \{c_1, \ldots, c_n\}$ based on the matrix $R^\alpha(C)$ the set $\{r_1, \ldots, r_m\}$ of relative priorities is calculated. Less important concerns are filtered, the set of concerns is reduced to $m \leq n$ concerns and their recalculated priorities are computed as $\{r'_1, \ldots, r'_m\}$. Then, for each concern $c_i$ the matrix $R^\alpha(c_i, \mathcal{T}_i)$ is used for ranking its qualifier tags and their local priorities are respectively $r'_1, \ldots, r'_m$.

The final ranking for the qualifier tags is delivered as $n \cdot r'_1, \ldots, n \cdot r'_m$ for $i = 1, \ldots, m$.

Let us introduce a simple example for understanding the nature of conditional requirements for relative priorities of concerns and their qualifier tags introduced thus far.

**Example 1.** In the previously introduced example about the online store, stakeholder might define priorities between concerns:

- Unconditionally: $\text{ICost} \succ \text{I Sale}$
- Conditionally: in case of a high preference over $\text{International Sale}$, then $\text{Scc} \succ \text{CustEase}$, otherwise $\text{Sec} \succ \text{CustEase}$.

In the second level, i.e., the level of qualifier tags, the stakeholder might conditionally define the importance between the qualifier tags of concern $\text{Security}$ as follows:

In case of a high preference over $\text{International Sale}$, $\text{Security}$ should be high (i.e., $\text{Sec.High} \succ \text{Sec.Medium}$, $\text{Sec.High} \succ \text{Sec.Low}$; Sec.Medium $\succ$ Sec.Low); otherwise, delivery cost should be medium or high (i.e. $\text{Sec.High} \succ \text{Sec.Medium}$, $\text{Sec.High} \succ \text{Sec.Low}$; Sec.Medium $\succ$ Sec.Low).

It is important to mention that the relative importance among qualifier tags of one concern might not be defined with only one conditional statement, i.e., the stakeholder might define one more conditional statement about relative importance between the qualifier tags of $\text{Security}$ as follows:

In case of a low preference over $\text{Implementation Cost}$, $\text{Security}$ can be medium (i.e., $\text{Sec.High} \succ \text{Sec.Medium}$, $\text{Sec.High} \succ \text{Sec.Low}$, Sec.Medium $\succ$ Sec.Low); otherwise, $\text{Security}$ should be high (i.e., $\text{Sec.High} \succ \text{Sec.Medium}$, $\text{Sec.High} \succ \text{Sec.Low}$).

We can see that the relative importance between two concerns might depend on the qualifier tags of other concerns. Also, relative importance between qualifier tags of one concern might depend on the qualifier tags of other concerns. We consider that any other way of dependency is not to be naturally expected for relative importance. In order to address the presented conditional requirements, we introduce Definitions 4 and 5.

**Definition 4 (Conditional relative importance between concerns).** Let us suppose that $C = \{c_1, \ldots, c_n\}$ is a set of concerns $\mathcal{Q}_T = \bigcup_{c_i \in C} \mathcal{Q}_T$, and $\mathcal{Q}_T, \mathcal{Q}_T'$ is the set of their qualifier tags. Conditional relative importance between concerns according to relation $\succ$ is defined as a quadruple $(\Psi, C, R^\alpha(C), R^\alpha(C)_2)$ where:

- $\Psi$ – propositional logic formula over a set of tags $\bigcup_{c_i \in C} \mathcal{Q}_T$ as propositional variables connected with $\land$, $\lor$, and $\neg$ representing logical operators conjunction, disjunction and negation, respectively;
- $\mathcal{Q}_T, \mathcal{Q}_T'$ – set of tag values of concern $c_i$;
- $R^\alpha(C)$ – relative importance matrix for concerns if $\Psi$ is true;
- $R^\alpha(C)_2$ – relative importance matrix for concerns if $\Psi$ is false.

Whether $\Psi$ is true or not determines which matrix of relative importance would be applied. This can be represented by pseudo-logical statements in the following format: $\Psi : R^\alpha(C), R^\alpha(C)_2$.

For example, for three given concerns $\{c_1, c_2, c_3\}$, conditional relative importance can be specified as $\Psi : c_1 \succ c_2, c_1 \succ c_3, c_2 \succ c_3 \land c_1 \succ c_2, c_1 \succ c_3, c_2 \succ c_3$. That is, if $\Psi$ is satisfied, the matrix of relative importance is defined by $\{a_0 > c_2, a_0 > c_1, c_1 > a_0 > c_3\}$, otherwise by $\{c_1 > c_2, c_1 > c_3, c_2 > c_3\}$.

**Definition 5 (Conditional relative importance for concern $c_i$).** Let us suppose that $C$ is a set of concerns and $\mathcal{Q}_T$ is their appropriate set of concern qualifier tags. Conditional relative importance for concern $c \in C$ according to relation $\succ$ is defined as a 5-tuple $(\Psi, c_i, \mathcal{Q}_T, R^\alpha(c_i, \mathcal{Q}_T), R^\alpha(c_i, \mathcal{Q}_T)_2)$ where:

- $\Psi$ – propositional logic formula over set of tags $\bigcup_{c_i \in C} \mathcal{Q}_T$ as propositional variables connected with $\land$, $\lor$, and $\neg$ representing logical operators conjunction, disjunction and negation, respectively;
the previously defined conditional requirements:

c - set of tag values of concern c;

\( R^c (c, QTc) \) - relative importance matrix for concern c if the condition holds;

\( R^c (c, QTc) \) - relative importance matrix of concern c if the condition does not hold;

Whether \( \Psi \) is satisfied or not determines which matrix of relative importance will be applied. This can be defined by pseudo-logical statements in the format: \( \Psi: R^c (c, QTc)_1 \rightarrow R^c (c, QTc)_2 \).

For example, for a concern with only three tag values \( \{t_1, t_2, t_3\} \), conditional relative importance can be specified as:

\[ \Psi: t_1 \rightarrow t_2, t_1 \rightarrow t_3, t_2 \rightarrow t_3 | t_2 \rightarrow t_1, t_1 \rightarrow t_3, t_2 \rightarrow t_3. \]

This means that if condition \( \Psi \) is satisfied, the matrix of relative importance is determined with \( \{t_1 \rightarrow t_2, t_1 \rightarrow t_3, t_2 \rightarrow t_3\} \), otherwise with \( \{t_2 \rightarrow t_1, t_1 \rightarrow t_3, t_2 \rightarrow t_3\} \). Restrictions on condition \( \Psi \) for relative importance between the qualifier tags of concern c, which cannot contain qualifier tags of concern c, is a part of Definition 5. It means that qualifier tags cannot depend on themselves.

**Example 1 (Continued).** In the first level (concerns), the stakeholder defines priorities with one unconditional and one conditional requirement. In this case, the conditional and unconditional requirements should be joined together as per Def. 4 as follows:

\[
\text{ISale.High} : \text{ICost} \rightarrow \text{ISale}, \text{Sec} \rightarrow \text{CustEase} | \\
\text{ICost} \rightarrow \text{ISale}, \text{Sec} \rightarrow \text{CustEase}, \text{ICost} \rightarrow \text{Sec}
\]

This requirement defines relative importance between four concerns and according to the comments after Definition 4, it is uniquely defined with relative importance between each pair of concerns. If the stakeholders do not specify each value, by default we consider that undefined priorities represent stakeholder’s indifference and according to comments of Definition 1 it is equal to one. So, the previous requirements are transformed into:

\[
\text{ISale.High} : \\
\text{ICost} \rightarrow \text{CustEase}, \text{ICost} \rightarrow \text{Sec}, \text{ISale} \rightarrow \text{CustEase}, \text{ISale} \rightarrow \text{Sec}, \\
\text{ICost} \rightarrow \text{ISale}, \text{Sec} \rightarrow \text{CustEase}
\]

\[
\text{ICost} \rightarrow \text{CustEase}, \text{ISale} \rightarrow \text{Sec}, \\
\text{ICost} \rightarrow \text{ISale}, \text{Sec} \rightarrow \text{CustEase}, \text{ICost} \rightarrow \text{Sec}
\]

Based on the standard AHP algorithm, we can calculate priority values for concerns in both cases: when *International Sale* is high and when it is not. It means that the requirement can be transformed into sets of priorities as follows:

\[
\text{ISale.High}: \\
\text{Sec} \rightarrow 0.35, \text{CustEase} \rightarrow 0.16, \text{ISale} \rightarrow 0.18, \text{ICost} \rightarrow 0.31
\]

\[
\text{Sec} \rightarrow 0.24, \text{CustEase} \rightarrow 0.19, \text{ISale} \rightarrow 0.16, \text{ICost} \rightarrow 0.40
\]

This shows that the first stage of prioritizing concerns is completed. The next stage covers prioritization of the qualifier tags of each concern. We will show that process for concern *Security* and the previously defined conditional requirements:

\[
\text{ISale.High}: \\
\text{Sec.high} \rightarrow \text{Sec.high, Sec.high} \rightarrow \text{Sec.low, Sec.medium} \rightarrow \text{Sec.low}
\]

\[
\text{Sec.high} \rightarrow \text{Sec.medium, Sec.high} \rightarrow \text{Sec.low, Sec.medium} \rightarrow \text{Sec.low}
\]

\[
\text{ICost.low}: \\
\text{Sec.high} \rightarrow \text{Sec.medium, Sec.high} \rightarrow \text{Sec.low, Sec.medium} \rightarrow \text{Sec.low}
\]

\[
\text{Sec.high} \rightarrow \text{Sec.medium, Sec.high} \rightarrow \text{Sec.low, Sec.medium} \rightarrow \text{Sec.low}
\]

Analogous to the level of concerns, the requirements should be transformed into sets of priority values which are calculated with standard AHP:

\[
\text{ISale.High}: \\
\text{Sec.high} \rightarrow 0.63, \text{Sec.medium} \rightarrow 0.26, \text{Sec.low} \rightarrow 0.11 \\
\text{Sec.high} \rightarrow 0.43, \text{Sec.medium} \rightarrow 0.43, \text{Sec.low} \rightarrow 0.14
\]

\[
\text{ICost.low}: \\
\text{Sec.high} \rightarrow 0.29, \text{Sec.medium} \rightarrow 0.57, \text{Sec.low} \rightarrow 0.14
\]

Finally, global ranking for qualifier tags and their prioritization will be calculated by checking conditions and applying appropriate values. For example, in case of high *International Sale* and medium *Implementation Cost*, ranking for concerns is:

\[
\text{Sec} \rightarrow 0.35, \text{CustEase} \rightarrow 0.16, \text{ISale} \rightarrow 0.18, \text{ICost} \rightarrow 0.31 \quad \text{condition ISale.High is satisfied} \\
\text{and tag of concern Security:} \\
\text{Sec.high} \rightarrow 0.63, \text{Sec.medium} \rightarrow 0.26, \text{Sec.low} \rightarrow 0.11 \quad \text{(in the first requirement condition is satisfied and in the second it is not)}
\]

The final rankings are:

\[
\text{Sec} \rightarrow 0.35 \times 0.63 = 0.22, \text{Sec.medium} \rightarrow 0.35 \times 0.26 = 0.08, \\
\text{Sec.low} \rightarrow 0.35 \times 0.11 = 0.04
\]

Note that in the above example in case of low *International Sale* and low *Implementation Cost*, priorities among qualifier tags are not uniquely determined, as the first layer results in:

\[
\text{Sec} \rightarrow 0.24, \text{CustEase} \rightarrow 0.19, \text{ISale} \rightarrow 0.16, \text{ICost} \rightarrow 0.40 \quad \text{and the second in Sec.high} \rightarrow 0.29, \text{Sec.medium} \rightarrow 0.57, \text{Sec.low} \rightarrow 0.14.
\]

This means that based on the stakeholders’ requirements the unique values for priorities of qualifier tags of concern *Security* cannot be generated and it should be recognized as inconsistency in the stakeholders’ requirements. This is discussed in the next section.

### 5. EXTENSION OF S-AHP

The overall process of the extended S-AHP is similar to the standard S-AHP introduced in [2]. However, some stages have been changed and new stages have been added. We extend S-AHP to support conditional relative importance for concerns. The extension consists of the following stages:

**Define conditional relative importance.** In order to use conditional S-AHP, this stage captures the conditional relation defined on concerns as well as the corresponding relative importance of the qualifier tags.

When we have a set of n concerns \( C = \{c_1, \ldots, c_n\} \) the matrix of size \( n \times n \) should be filled for calculating priorities between concerns and filtering less important ones. As shown in Example 1, it is more natural to expect that the stakeholders partially define their requirements, i.e., they unconditionally define relative importance of subset of \( n_1 < n \) concerns, conditional on another condition for
subset of \( m < n \) and so on. All of these situations are to be expected and should be addressed. A special framework for processing stakeholder requirements is beyond the scope of this paper. Requirements about relative importance for the subset of concerns should be joined into requirements which would satisfy Definitions 2-5 (as shown in Example 1). If any of the specified conditions cannot be satisfied, it should be recognized as an inconsistency in the requirements and the stakeholders should be informed. They will have a chance to correct the inconsistency before the next step. The same situation is with requirements about relative importance between qualifier tags of each concern.

For the next step of Conditional S-AHP, we assume that the stakeholders’ requirements are specified in the form of Definitions 2-5. Given these, the following steps need to be taken:

**Determine the local priorities based on conditions.** For each requirement about relative importance of concerns \( \Psi : R^*(C_1) \mid R^*(C_2) \) we calculate their priorities in both cases, when the condition is satisfied and when it is not. Also, for each requirement \( \Psi : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) about importance between qualifier tags of concern \( c_i \) we calculate their local priorities in the both situations. All of these priorities would be later used for final ranking.

**Option ranking.** This steps aims at ranking the relative importance of the qualifier tags of the remaining concerns and uses them to rank the available options. This will provide a final ranking over the most important concerns and their most significant qualifier tags. The process of ranking options is applied in five steps for each option separately:

**Step 1.** For each conditional requirement between concerns, conditions are checked and priorities which satisfy all conditions are chosen. If unique values for priorities do not exist (i.e., different requirements define different priorities for the same concern), it should be addressed as inconsistency in the requirements.

It is important to mention that not all options are annotated with qualifier tags for each concern, so situations when a condition cannot be checked for a certain option is possible. Let annotate with \( q \) any qualifier tag which is not used to annotate an option. Then, the following rules apply:

- If a condition only contains \( q \), the conditional requirement is inconsistent and the stakeholders should be informed about it;
- If a condition contains \( q \) and other qualifier tags, we consider that logical value of the whole condition is false, i.e., the option does not satisfy the condition.

**Step 2.** The stakeholders filter out less important concerns. Their priorities should be proportionally shared with others as it is described in Section 3.

**Step 3.** For each conditional requirement between the qualifier tags of the most significant concerns, conditions are checked and local priorities which satisfy all conditions are chosen. If unique values for priorities do not exist (i.e., different requirements define different priorities for the same qualifier tag), this should be addressed as inconsistency in the requirements. If the stakeholders filtered out a concern in the previous step, it is a sign that it is of less interest in the ranking process and conditions which include its qualifier tags should be modified. We modify its conditions, so that they cannot depend on qualifier tags that are out of interest: *Conditions’ reduction of less important concerns’ qualifier tags.* If we denote by \( q \) any qualifier tag of any less important concern, each conditional requirement should be transformed iteratively using the following rules:

**Rule 1:** Conditional requirement in the forms \( \Psi \land q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) or \( \Psi \land \neg q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) transforms into \( \Psi : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \).

**Rule 2:** Conditional requirement in the forms \( \Psi \land q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) or \( \Psi \land \neg q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) transforms into \( \Psi : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \).

**Rule 3:** Conditional requirement in the forms \( q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \), \( \neg q : R^*(c_i, QT_j) \mid R^*(c_i, QT_{j+1}) \) is inconsistent and the stakeholder should be asked for its resolution.

Rules 1 and 2 should be repeated until either the condition defined in rule 3 is satisfied or a requirement, which does not contain any qualifier tags of the filtered concerns, is obtained.

**Step 4.** Based on the selected priorities of concerns and local priorities of its qualifier tags, global priorities are calculated.

**Step 5.** Final option ranking based on the priority of the qualifier tags is completed.

6. **ILLUSTRATIVE EXAMPLE**

In Section 3, we have introduced the widely used e-store feature model. Suppose that an actual e-store needs to be developed based on the stakeholders’ requirements. Conditional S-AHP can be used to select the best set of features. Throughout this illustration, we will reference to numbered elements of Example 1.

First, stakeholders define requirements about the relative importance for the level of concerns defined in (2) in Example 1 and relative importance between the qualifier tags for each concern. Let us now suppose that in addition to (3) defined in Example 1 where the stakeholders defined requirements about relative importance between tags of concern Security, they also defined:

- For international sale:
  
  Sec:Medium \( \sim ISale.High \sim ISale.Medium \sim ISale.High \sim ISale.Low \),
  
  ISale.Medium \( \sim ISale.Low \mid ISale.High \sim ISale.Medium \sim ISale.High \sim ISale.Low \sim ISale.Medium \sim ISale.High \sim ISale.Low \)

- For implementation cost and customer ease:
  
  ICost.Low \( \sim ICost.Medium \sim ICost.Low \sim ICost.High \sim ICost.Medium \sim ICost.High \sim ICost.Medium \sim ICost.High \)

- And
  
  CustEase.Low \( \sim CustEase.Medium \sim CustEase.High \sim CustEase.Medium \sim CustEase.Low \)

The stakeholders might define requirements partially and incompletely, and such requirements should be joined together and amended to be completed as shown in (4).

The next phase of Conditional S-AHP includes determination of local priorities based on each requirement. First, priorities are computed for the level of concerns (5) and then for the level of tags for each concern. In (6) local priorities for tags of concern Security are calculated and analogously local ranks for other concerns are calculated and their values are respectively:

Sec:Medium : ISale:High \( \sim 0.57 \), ISale:Med \( \sim 0.29 \), ISale:Low \( \sim 0.14 \), ISale:High \( \sim 0.43 \), ISale:Med \( \sim 0.43 \), ISale:Low \( \sim 0.14 \)

ICost:High \( \sim 0.14 \), ICost:Med \( \sim 0.29 \), ICost:Low \( \sim 0.57 \)

CustEase:High \( \sim 0.63 \), CustEase:Med \( \sim 0.26 \), CustEase:Low \( \sim 0.11 \).
These values will be used in the next phase, which includes calculation of the final rank for each option. Suppose that an application engineer is interested in choosing the best shipping method based on the average importance assigned to the features. This is completed for each available feature through the following steps:

Feature Pickup shipping is annotated with high Sec, low ISale and low ICost. In the first step, conditions in each conditional requirement for the level of concerns are checked and those whose condition for this feature is satisfied are selected. In (5), condition is not satisfied, so valid ranking for concerns Security, Customer Ease, International Sale and Implementation Cost are 0.24, 0.19, 0.16, 0.40, respectively. In the second phase, the stakeholders filter out less important concerns. Let us suppose that the stakeholders decided to keep only concerns Security and Implementation Cost due to their higher priority values. The sum of priorities of less important concerns Customer Ease and ISale should be proportionally shared with others as it is described in (1) and their new values are 0.38 and 0.62, respectively. In the next step conditions in each conditional requirement about importance for the level of tags are checked and those whose condition is satisfied are selected. As less important concerns are filtered out, conditions’ reduction of its qualifier tags should be done according by rules defined in previous section. It means that first one of requirements (6) is inconsistent based on rule 3 and stakeholders should resolve it. They will decide which priority should be chosen as appropriate one. Let us suppose that they chose Sec:High = 0.63, Sec:Med - 0.26, Sec:Low = 0.11. The second requirement about priorities of tags of the same concern in (6) should not be reduce, as its condition does not contain qualifier tags of less important concerns. Its condition is satisfied so it gives the local priorities for tags of concern Security: Sec:High = 0.29, Sec:Med - 0.57, Sec:Low = 0.14. But, these values are not the same as previously obtained by the first requirement, so this is a newly recognized inconsistency. The stakeholders should again decide which relative priority for high Sec is relevant: 0.63 (based on the first requirement) or 0.29 (based on the second requirement). Let us suppose that they choose 0.63. It still remains to check the requirement defined for the relative priorities of the tags of concern ICost and it gives local priority for low ICost of 0.57. Now, the next step for the calculation of global priorities can be performed. The global priority for high Sec is 0.38*0.63 = 0.24 and for low ICost is 0.62*0.57 = 0.35. So, the final step gives the average final priority for feature Pickup Shipping which equals to 0.29.

Similarly, feature E-delivery is annotated with high CustEase, high ISale and medium ICost and its final average priority is 0.27. The final priority for Airmail is 0.32 and for Regular mail 0.17. So, the feature with the highest rank is Airmail and it is recommended based on the stakeholders’ requirements as the most appropriate feature for inclusion in the final application.

7. DISCUSSION AND RELATED WORK

Within the software engineering research, the requirements engineering community has significantly contributed to the development of prioritization techniques [11]. In this context, one suggestion for prioritizing requirements has been the use of requirement priority groups as a way to nest similar requirements together and create an internal rank for requirements in each group. Similarly, the hierarchical AHP [3] is proposed in the requirement engineering literature to create a structure for interrelated requirements. In this method, only requirements at the same layer are compared with each other, and hence this reduces the total number of required comparisons in contrast to AHP.

Due to the space constraint for this paper, we only outline how conditional S-AHP satisfies the requirements of an effective prioritization technique based on the challenges and characteristics that have been introduced in [1]:

- In our approach, stakeholders are able to define conditional and unconditional preferences about the available concerns and its qualifier tags. Compared with the S-AHP method that does not support conditional preferences, concerns and their qualifier tags do not have static final priorities because they depend on options which might either satisfy the conditions defined by stakeholder’s requirements or not. The ranking process is performed in steps, i.e., local rankings are calculated only once independently of options after the stakeholders have defined their requirements. Global priorities are computed from local priorities for each option after checking if an option annotated with appropriate qualifier tags satisfies the conditions in conditional requirements; thus, our approach effectively handles conditional preferences.

- The activities within Conditional S-AHP are very easy to perform and are based on a simple pair-wise comparison method. It can be inexpensively implemented in a spreadsheet program such as MS Excel with additional usage of any program for checking satisfaction of conditions in conditional defined requirements.

- During the whole process, in each step, we check whether an inconsistency exists in the requirements of the stakeholders, and if so, stakeholders’ intervention might be necessary.

- We decided to have local calculation of priorities, as it disables any cycles and dependency in the processing of requirements. For further explanation let us consider previously defined requirements about relative importance of qualifier tags of concerns Security and International Sale. With (3) from Example 1, the stakeholders define Security dependent on International Sale and with (7) International Sale dependent on Security. These two requirements cannot be processed with the method introduced in TCP-nets because they make a cycle of dependency between these two concerns. On the other hand, as it can be seen in the case study in Section 5, Conditional S-AHP does not suffer from any cycle in dependency between the concerns.

- In comparison to S-AHP, the number of pair-wise comparisons is the same as in the case when the stakeholders define each conditional requirement as two separate unconditional requirements. Any additional operation includes checking conditions for each option and calculating global priorities by simply multiplying local priority of qualifier tags by its concern’s priority, which takes linear complexity for each option. As the number of options increases and is much larger than the number of concerns and its qualifier tags, it is possible to develop optimization for reusing global ranks which will reduce the overall complexity.

- Regarding group decision making ([4 [14]), AHP considers two different approaches: aggregation of individual judgments (AIJ) and aggregation of individual priorities (AIP) [15]. Conditional S-AHP is also able to include preferences from different stakeholders with different interests and views with both approaches. If there are a large number of different stakeholders, conditional S-AHP provides proper mechanisms for grouping them. The main advantage of the introduced definitions and our approach is that it will be able to even unify different conditional and unconditional stakeholder preferences into a unique representation.

It is important to mention that our work is based on the well accepted and widely adopted AHP method, which has been exten-
sively used in many important decision making domains such as forecasting, total quality management, business process re-engineering, quality function deployment, and the balanced scorecard just to name a few [14, 4]. Therefore, given the adoption of AHP in such domains, it is possible for practitioners to easily use our approach that provides them with the additional benefit of supporting conditionality. This transition to using conditional S-AHP is smooth since the practitioners are already familiar with AHP, which they have been frequently using in their work. In addition, since AHP requires too many pair-wise comparisons in some cases, the use of S-AHP that handles and alleviates this is an extra advantage for conditional S-AHP. Therefore, conditional S-AHP provides the additional benefit of reducing the required number of pair-wise comparisons as shown in [2] and also supports conditionality, while at the same time keeps the added complexity of its approach reasonably low, which makes it quite simple to use by practitioners.

Also related, the Bubble sort technique has been used to rank order the requirement statements. Bubble sort is in essence very similar to AHP with the slight difference that requirement comparisons are made to determine which requirement has a higher priority, but not to what extent. It is clear that Bubble sort suffers from similar issues to AHP, e.g., the large number of required comparisons. There have been proposals to reduce the number of required comparisons in comparison-based techniques, which are generally referred to as incomplete pair-wise comparison methods [9]. These techniques are based on some local and/or global stopping rule, which determines when further comparison will not reveal more useful information with regards to the prioritization of the options. Such techniques can be beneficial if used along with techniques such as AHP, S-AHP, and Bubble sort.

Additionally, Hierarchical Cumulative Voting (HCV) has been used to prioritize requirements where top vote-getter requirements are prioritized higher than the others [8]. One of the drawbacks of this approach is that as the number of requirements (options) increases, it becomes very hard for the stakeholders (voters) to select the best voting tactic, which would reveal their preferences about the highest priority requirements. In addition, HCV assumes that it is possible to hierarchically divide the objects of interest into different levels but does not contain any mechanism for doing so [10]. If we would like to extend HCV for both conditional and unconditional situations, the first solution would be to divide conditional and unconditional options in two different groups. Although in such a case it may happen that we have only one unconditional option (or even none) and a large number of conditional options which means that we have a new problem of properly dividing the options into groups. On the contrary, if all of the options are in one block, there is a problem of compensation which is well known for this method.

TCP-nets [7] are very closely related to our proposed work but it has limitations for quantitative approach such as prioritizing options based on hierarchical structure of criterions. There is another variant of the TCP-net, known as UCP-nets [16] that capture quantitative preferences and relative importance information using utility functions. They combine the theory of CP-nets and GAIA-nets (generalized additive decomposable utility functions). They have a limitation, as they do not make any assumption on the kind of interactions between attributes that need to be prioritized [16].

8. CONCLUSION

Conditional S-AHP is a layered prioritization framework that effectively captures both conditional and unconditional preferences. It addresses the shortcomings of methods such as TCP-nets. Our work is based on the well-known quantitative prioritization AHP algorithm. Besides being able to support conditional requirements, the framework is able to recognize the different inconsistencies in the stakeholders’ requirements and effectively handle them through their resolution. Our future work will include the formal specification of Conditional S-AHP and the proof of its correctness, completeness and soundness. Also, a framework for processing partial and incomplete requirements should be developed that would provide appropriate data structures for efficiently storing and reusing the calculated global priorities.

9. REFERENCES