An extension of CHOICLA User Interfaces for Configurable Products

Thi Ngoc Trang Tran
Institute for Software Technology
Inffeldgasse 16b/II
A-8010 Graz, Austria
ttrang@ist.tugraz.at

Martin Stettinger
Institute for Software Technology
Inffeldgasse 16b/II
A-8010 Graz, Austria
mstettinger@ist.tugraz.at

Müslüm Atas
Institute for Software Technology
Inffeldgasse 16b/II
A-8010 Graz, Austria
muatas@ist.tugraz.at

Alexander Felfernig
Institute for Software Technology
Inffeldgasse 16b/II
A-8010 Graz, Austria
afelfern@ist.tugraz.at

ABSTRACT
Configuration technologies are usually employed in closed settings where only single users (or small group of users) configure the corresponding products. However, in reality, there are many situations in which products need to be configured jointly by groups with many users. CHOICLA is a group decision making environment supporting decision scenarios in a domain-independent fashion. The current version of CHOICLA solely offers decisions for groups on non-configurable alternatives. In this paper, we propose the extension of CHOICLA's user interfaces supporting group-based configuration processes for complex products.

CCS Concepts
- Information systems → Decision support systems; Recommender systems.
- Human-centered computing → Interaction design process and methods; Software and its engineering → Designing software;

Keywords

1. INTRODUCTION
Configuration is a basic form of design activity where the target product is composed from a set of predefined parts in a way which is consistent with a given set of constraints [15]. It is applied in many domains, such as financial services [2], software requirement engineering [10], furniture [7], and telecommunication [6]. Most configuration technologies are performed in closed settings where knowledge bases and corresponding configurations are developed by a single (or a small group of) knowledge engineers [4]. Implementing configuration applications in closed settings reveals hindrances in terms of missing scalability in knowledge engineering [11] and sub-optimal decisions [5] when a single user's decision does not reflect the group preferences in an optimal fashion. Group-based configuration stated in [4] is a potential approach to overcome these drawbacks. This technique is considered as a part of open configuration which allows groups of users to be engaged in configuration processes and jointly configure complicated products and services in a consensual fashion.

CHOICLA is a decision support environment which supports groups of users to make decisions in various scenarios in a domain-independent fashion [12]. Domain-independent support of decision tasks is a key feature of CHOICLA compared to other group decision making support tools, such as Dotmocracy1, Doodle2. The current version of CHOICLA optimizes the quality of group decisions by integrating explanations, group recommendations, and counteracting negative influences of psychological biases on decision outcomes [13], [14]. In this paper, on the basis of maintaining group decision making functions in the current version of CHOICLA, we propose some user-interface extensions which assist groups of users in configuring complex products.

Examples in the tourism domain are presented throughout the paper as an illustration for the extension of CHOICLA user interfaces.

The remainder of this paper is organized as follows. In Section 2, we provide a formal definition of group-based configuration task and group-based configuration. In Section 3, we give a short description of the current version of CHOICLA and how it could be extended for group-based configuration processes. Finally, we conclude the paper with Section 4.

2. GROUP-BASED CONFIGURATION
As a basis for the following discussions, formal definitions of a group-based configuration task and a group-based configuration are presented based on a Constraint Satisfaction Problem (CSP) [16].

Definition 1: Group-based Configuration Task. A group-based configuration task can be defined as a CSP \((V, D, C)\) where \(V\) is a set of variables, \(D\) represents the corresponding domain definitions, and \(C = REQ \cup CKB \cup I\) represents specific requirements of users, a set of constraints, and a set of products from a catalog. In this context, \(REQ = \bigcup_i REQ_i\) is the union of user requirements \(REQ_i\), \(CKB\) represents a configuration knowledge base.

1www.dotmocracy.org
2www.doodle.com
**Definition 2**: Group-based Configuration. A group-based configuration (solution) for a group-based configuration task is a complete set of assignments \( \text{CONF} = \bigcup_{i} a_i : v_i = v_{a_i} \) to the variables \( v_i \in V \) such that \( \text{CONF} \cup \text{REQ} \cup \text{CKB} \cup I \) is consistent.

**Example 1**: Group-based Configuration Task. We exemplify of a group-based configuration task in the tourism domain with components defined as follows where dest denotes a destination, att denotes a tourist attraction, and trans denotes a means of transportation:
- \( V = \{ \text{hotel#stars}, \text{food}, \text{dest}, \text{att}, \text{cost}, \text{trans} \} \).
- \( D \) = {
  \text{dom(hotel#stars)} = \{2, 3, 4, 5\},
  \text{dom(food)} = \{
    \text{Asian}, \text{European}, \text{American}, \text{Australian}\},
  \text{dom(dest)} = \{
    \text{Europe}, \text{Asia}, \text{America}, \text{Australia}\},
  \text{dom(att)} = \{
    \text{monument}, \text{museum}, \text{palace}, \text{beach}, \text{mountain}, \text{river}\},
  \text{dom(cost)} = \{100..2000\},
  \text{dom(trans)} = \{\text{plane}, \text{train}, \text{bus}\} \}.
- \( \text{CKB} \) = {
  c_1: \text{dest} = \text{Asia} \Rightarrow \text{food} = \text{Asian},
  c_2: (\text{trans} = \text{plane}) \land (\text{hotel#stars} \geq 3) \Rightarrow \text{cost} \geq 900,
  c_3: \text{dest} = \text{Asia} \Rightarrow \text{trans} = \text{plane},
  c_4: \text{cost} \leq 500 \Rightarrow (\text{dest} = \text{Europe}) \land (\text{trans} = \text{bus}),
  c_5: \text{dest} = \text{America} \Rightarrow \text{cost} \geq 1200) \}.
- \( \text{REQ} \) = {
  \text{REQ}_1 = (\text{trans} = \text{plane}, 1000 \leq \text{cost} \leq 1500, \text{hotel#stars} = 3, \text{att} = \text{beach}),
  \text{REQ}_2 = (\text{food} = \text{Asian}, 500 \leq \text{cost} \leq 1000, \text{att} = \text{beach}),
  \text{REQ}_3 = (\text{dest} = \text{Asia}, 500 \leq \text{cost} \leq 1000, \text{hotel#stars} = 3). \}
- \( I \) = {
  I_1 = \{(\text{trans} = \text{plane}) \land (\text{food} = \text{Asian}) \land (\text{dest} = \text{Asia}) \land (\text{att} = \text{beach}) \land (\text{cost} = 1000) \land (\text{hotel#stars} = 3) \}\lor
  I_2 = \{(\text{trans} = \text{bus}) \land (\text{food} = \text{European}) \land (\text{dest} = \text{Europe}) \land (\text{att} = \text{mountain}) \land (\text{cost} = 450) \land (\text{hotel} = 2) \}\lor
  I_3 = \{(\text{trans} = \text{plane}) \land (\text{food} = \text{American}) \land (\text{dest} = \text{America}) \land (\text{att} = \text{monument}) \land (\text{cost} = 1200) \land (\text{hotel#stars} = 3) \}\}.

**Example 2**: Group-based Configuration. On the basis of the example about group-based configuration task, a corresponding solution is created by a configuration solver, i.e., \( \text{CONF} = \{\text{trans} = \text{plane}, \text{food} = \text{Asian}, \text{dest} = \text{Asia}, \text{att} = \text{beach}, \text{cost} = 1000, \text{hotel#stars} = 3\} \).

3. CHOICLA AND EXTENDED VERSION FOR CONFIGURABLE PRODUCTS

The main objective of CHOICLA [12] is to provide a group decision support environment assisting various types of group decision scenarios in different application domains. In this section, we will discuss in detail main functions of CHOICLA's current version and propose some corresponding extensions on user interfaces facilitating group-based configuration processes for complex products.

**Configuration requirement management.** The current version of CHOICLA is capable of supporting the management of decision tasks in a flexible fashion. In the initial phase, a set of alternatives (at least one alternative) is entered by the creator of decision task and every participant is allowed to add more alternatives during an ongoing decision process. In addition, more participants can be also added during the decision process. However, the management of decision alternatives in the current version of CHOICLA is only for non-configurable products. Therefore, this version solely provides user interfaces to add basic information for alternatives (i.e., name, description, and image). Regarding product configuration scenarios, in the extended version of CHOICLA, each decision task corresponds to a configurable product which includes many properties. Each property is defined by name, description, and property domain. CHOICLA user interfaces need to be extended such that users not only can add a new property but also define manually different values for a property domain. For example, as depicted in Figure 1a, a CHOICLA's extended user interface in the tourism domain allows users to add a new property named "attribution" and define corresponding domain values (e.g., monument, museum, palace, mountain, beach, etc). In addition, with the same user interface, users can also add more properties and corresponding domain values.

**User requirement configuration.** In CHOICLA's current version, users are allowed to specify their preferences for a certain alternative by simply specifying a rating value. However, when configuring user requirements for complex products with many properties inside, users have to choose suitable values for each property. Figure 1b illustrates a proposed user interface which helps a user to configure a set of requirements on different properties of a tourist package, including hotel, food, destination, attraction, transportation, and cost. In the initial phase, for the purpose of counteracting the decision biases, "none" value is set for every property and users can replace this value with another value. Besides, with each property which is being configured, users are allowed to add new domain values (see Figure 1c). Furthermore, in order to minimize the user effort in configuration sessions, we exploit constraint-based reasoning [1] which takes into account the context of current user by interpreting a set of constraints. For instance, according to constraints \( c_1 \) and \( c_4 \) presented in Example 1, if a current user chooses "Asia" for dest then the values for trans and food will be set automatically to "plane" and "Asian".

**Preference visibility.** In group decision making processes, the visibility of preferences of other group members can bring different effects on the quality of decision outcomes [5], [8]. For instance, seeing individual preferences of all participants is very necessary in the scenario where all managers want to make a time agreement for a business meeting. However, in another group scenarios, preference visibility will deteriorate the decision quality [14]. For the purpose of counteracting the negative effects of decision biases, the current version of CHOICLA only allows to disclose the group preferences for a certain alternative after a current user completes his/her rating. This mechanism will be also maintained in the extended version of CHOICLA in which users' preferences are represented by a set of requirements. In Figure 1b, the "My requirements" tab allows a specific user to configure all his/her requirements. The "Suggestion" tab showing suggestions for the whole group is only activated after the user finishes a requirement configuration process and presses "SAVE MY REQUIREMENTS".

**Recommending explanations.** In the context of configuring products, there exists a high probability of situations where users specify requirements inconsistent with a set of constraints in the knowledge base (see Example 3). In order to support users to detect inconsistencies, minimal explanations (diagnoses) [3] are applied. Minimal explanations are minimal sets of requirements which have to be repaired or omitted such that a solution can be identified.

**Example 3**: Assuming that a user \( u \) has defined a set of re-
requirements as follows: \( \text{REQ}_u = \{ r_{u1}: \text{trans} = \text{plane}, r_{u2}: \text{food} = \text{Asian}, r_{u3}: \text{dest} = \text{Asia}, r_{u4}: 100 \leq \text{cost} \leq 500, r_{u5}: \text{att} = \text{Beach}, r_{u6}: \text{hotel#stars} = 4 \} \). We can observe that there are some inconsistencies here. Particularly, the combination of requirements \( \{ r_{u1}, r_{u4}, r_{u6} \} \) creates a conflict set (inconsistency) with constraint \( c_2 \). Another conflict set is \( \{ r_{u1}, r_{u3}, r_{u4} \} \) which is inconsistent with constraint \( c_4 \).

The inconsistency between requirements will be resolved by calculating minimal diagnoses from minimal conflict sets. A minimal diagnosis (see Definition 5) represents a minimal set of requirements that have to be deleted from \( \bigcup \text{REQ}_i \) such that a solution can be found for the remaining constraint.

Definition 3: A conflict set \( CS \subseteq \bigcup \text{REQ}_i \) is a minimal set of requirements such that \( \text{inconsistent}(CS) \), \( CS \) is minimal if there does not exist a conflict set \( CS' \) with \( CS' \) is a conflict set and \( CS' \subseteq CS \).

Definition 4: Group-based Configuration Diagnosis Task. A group-based configuration diagnosis task is defined by a group-based configuration task \( (V, D, C = \text{REQ} \cup \text{CKB} \cup I) \) where \( \text{REQ} \cup \text{CKB} \cup I \) is inconsistent.

Definition 5: Group-based Configuration Diagnosis. A diagnosis for a given group-based configuration task \( (V, D, C = \text{REQ} \cup \text{CKB} \cup I) \) is a set \( \Delta \) such that \( \text{CKB} \cup \text{REQ} \cup I - \Delta \) is consistent. \( \Delta \) is minimal if \( \forall \Delta' \subseteq \Delta \).

In the extended version of CHOICLA, as mentioned above, we propose a user interface in form of a tab, called "My requirements", in which each user can configure a set of requirements for a product. If there is an inconsistency between different requirements, a red circle will be shown next to that requirement. The user can click this circle to see the detail of the explanation.

Each explanation proposes one/some diagnosis set(s) that should be adjusted by users in order to remove the inconsistencies (see Figure 1b).

In the context of group-based configuration, if there exists an inconsistency between different requirements of group members, then the content of the explanation will be a list of diagnosis sets which should be modified by group members. This list is created by performing four steps: First, specifying a candidate set of minimal diagnoses. Second, considering the impact of the different diagnoses on the current requirements of users. Third, applying different group decision heuristics [9] (e.g., Least Misery, Average, Most Pleasure, Group Distance, Ensemble, etc) to figure out diagnoses acceptable for the whole group. Finally, by using a "less-is-better", the list of diagnoses is ranked for the purpose of choosing which diagnosis set will be recommended first to users. An illustration about the creation of a group explanation is presented in Example 4.

Example 4: Let us assume that three users \( u_1, u_2, \) and \( u_3 \) have specified the following requirements:

\( \text{REQ}_1 = \{ r_{11}: \text{trans} = \text{plane}, r_{12}: 1000 \leq \text{cost} \leq 1500, r_{13}: \text{hotel#stars} = 4 \} \)

\( \text{REQ}_2 = \{ r_{21}: \text{food} = \text{Asian}, r_{22}: \text{dest} = \text{Asia}, r_{23}: \text{trans} = \text{train} \} \)

\( \text{REQ}_3 = \{ r_{31}: \text{dest} = \text{Australia}, r_{32}: \text{hotel#stars} = 4, r_{33}: \text{att} = \text{museum} \} \)

Three minimal conflict sets determined here are \( \{ r_{11}, r_{23} \} \), \( \{ r_{22}, r_{31} \} \) and \( \{ r_{22}, r_{23} \} \). A list of corresponding minimal diagnoses extracted from the minimal conflict sets is: \( \{ (r_{11}, r_{22}), (r_{23}, r_{22}) \) and \( (r_{23}, r_{31}) \) \). In Table 1, we point out the influence of different strategies on the current requirements of users. Besides, in this example, we use the Least Misery strategy and show how this strategy effects on the selection of diagnoses in the group scenario. Least Misery strategy prefers alternatives minimizing the misery of
individual users in the group. This table depicts obviously that, by using "less-is-better" ranking criterion, a list of diagnoses shown in the explanation will be in the following order: 1-(r_{11}, r_{22}), 2-(r_{23}, r_{31}) and 3-(r_{23}, r_{22}).

| Group-based recommendation: Similar to the current version, the extended version of CHOICLA provides users a so-called "Suggestion" tab so that all group members can see a final suggestion for the whole group (see Figure 1d). For making a group recommendation, we use the following preference aggregation strategy [9]: First, a set of requirements of all members are aggregated into group requirements. After that, these group requirements will be matched with a set of products from a catalog in order to find out the appropriate solutions. The list of recommended products shown for a group will be ranked according to some criteria. For example, "more-is-better" [1] (e.g., the higher the energy is, the better the performance of a car is), less-is-better [1] (e.g., the lower price of a tourist package is better). In addition, the explanation for each group suggestion will be maintained for the purpose of increasing the trust of recommendations and letting users understand how a decision can be created. For example, the value of "cost" property for the whole group is 1000 EUR because this value configured by user_1, user_2, and user_3 are respectively [1000..1500], [500..1000], and [1000..1500].

| 4. CONCLUSION |

In this paper, we proposed some extended user interfaces from the current version of CHOICLA in order to support groups of users to configure complex products jointly in a domain-independent fashion. In this extended version, CHOICLA provides an environment where users are able to configure all requirements for a complicated product, add new properties as well as add more participants during an ongoing decision process. In addition, users are also provided a conflict-detection mechanism which helps to detect and repair inconsistent requirements. Furthermore, to increase the users' trust for recommendations, user interfaces supporting the insight of users' individual requirements are offered as the way to explain how a suggestion can be created for the whole group. Within the scope of future work, CHOICLA's user interfaces should be improved in order to support the more flexible configuration mechanisms, for instance, user can configure cost <= 400 EUR or cost >= 400 EUR instead of cost = 400 EUR; users can configure different values for a specific attribute, e.g., att = "mountain" AND att = "beach". In addition, in order to minimize the user effort, the content of explanations will be improved by pointing out not only requirements that have to be adapted by users but also some suggestions related to repair actions.

5. REFERENCES