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

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

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

Group Recommender Systems, Group Decision Making, Group-based Configuration, Knowledge-based Configuration.

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 Dotmocracy[1], Doodle[2]. 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* ∪ *CKB* ∪ *I* represents specific requirements of users, a set of constraints, and a set of products from a catalog. In this context, *REQ* = $\bigcup REQ_i$ is the union of user requirements REQ_i, *CKB* represents a configuration knowledge base.

[1] www.dotmocracy.org
[2] www.doodle.com

Definition 2: ***Group-based Configuration***. A group-based configuration (solution) for a group-based configuration task is a complete set of assignments *CONF* = $\bigcup a_i : v_i = v_{ai}$ to the variables $v_i \epsilon V$ such that *CONF* ∪ *REQ* ∪ *CKB* ∪ *I* is consistent.

Example 1: ***Group-based Configuration Task.*** We examplify 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*** = {*hotel#stars, food, dest, att, cost, trans*}.
- ***D*** = {
dom(hotel#stars) = {2, 3, 4, 5},
dom(food) = {*Asian, European, American, Australian*},
dom(dest) = {*Europe, Asia, America, Australia*},
dom(att) = {*monument, museum, palace, beach, mountain, river*},
dom(cost) = [100..2000],
dom(trans) = {*plane, train, bus*}}.
- ***CKB*** = {
c_1: *dest* = *Asia* ⇒ food = Asian,
c_2: (*trans = plane*) ∧ (*hotel#stars* ≥ *3*) ⇒ *cost* ≥ *900*,
c_3: *dest* = *Asia* ⇒ *trans* = *plane*,
c_4: *cost* ≤ *500* ⇒ *(dest = Europe)* ∧ *(trans = bus)*,
c_5: *dest* = *America* ⇒ *cost* ≥ *1200*}.
- ***REQ*** = {
REQ_1 = {*trans = plane*, *1000* ≤ *cost* ≤ *1500*, *hotel#stars = 3*, *att = beach*},
REQ_2 = {*food = Asian*, *500* ≤ *cost* ≤ *1000*, *att = beach*},
REQ_3 = {*dest = Asia*, *500* ≤ *cost* ≤ *1000*, *hotel#stars = 3*}}.
- ***I*** = {
I_1 = {*(trans = plane) AND (food = Asian) AND (dest = Asia) AND (att = beach) AND (cost = 1000) AND (hotel#stars = 3)*}OR
I_2 = {*(trans = bus) AND (food = European) AND (dest = Europe) AND (att = mountain) AND (cost = 450) AND (hotel = 2)*}OR
I_3 = {*(trans = plane) AND (food = American) AND (dest = America) AND (att = monument) AND (cost = 1200) AND (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., *CONF* = {*trans = plane*, *food = Asian*, *dest = Asia*, *att = beach*, *cost = 1000*, *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 "*attraction*" 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_3 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-

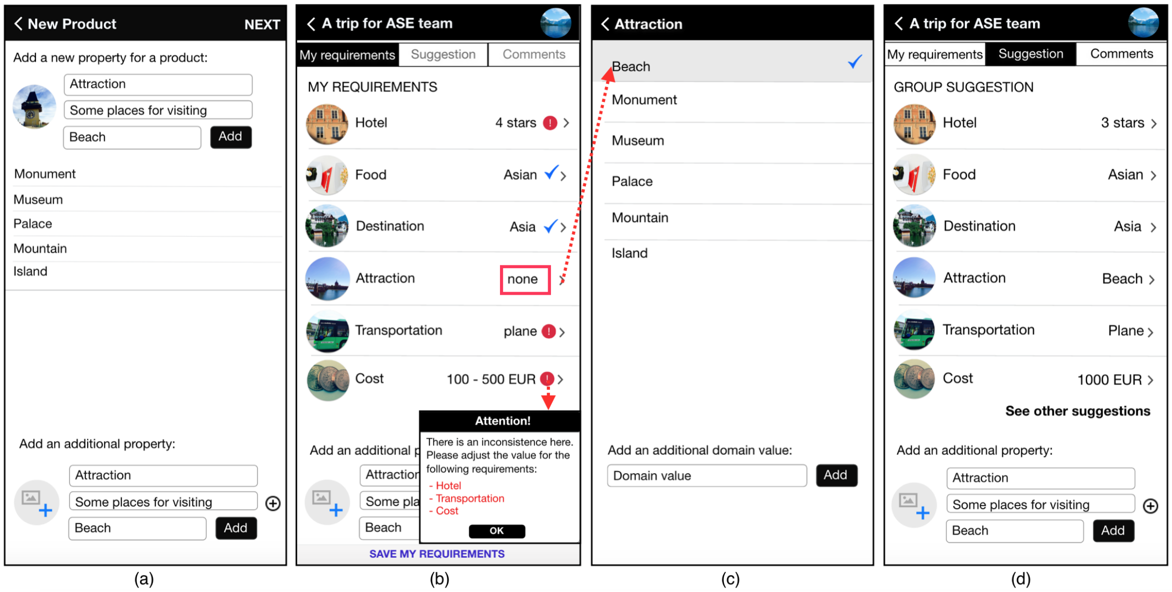

Figure 1: A sketch of CHOICLA's extended user interfaces for configurable products. Users can add attributes for a product and corresponding domain definitions. During the configuration process, user can get notifications from the system with regard to inconsistent requirements among group members. After finishing the requirement configuration process, users are provided a suggestion generated by the system for the whole group and the corresponding explanation for each attribute by clicking the arrow next to each attribute.

quirements as follows: REQ_u = {r_{u1}: *trans = plane*, r_{u2}: *food = Asian*, r_{u3}: *dest = Asia*, r_{u4}: *100* $\leq$ *cost* $\leq$ *500*, r_{u5}: *att = Beach*, r_{u6}: *hotel#stars = 4*}. We can observe that there are some inconsistences here. Particularly, the combination of requirements (r_{u1}, r_{u4}, *and* r_{u6}) creates a conflict set (inconsistency) with constraint c_2. Another conflict set is (r_{u1}, r_{u3}, *and* 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 $\bigcup REQ_i$ such that a solution can be found for the remaining constraint.

Definition 3: ***A conflict set*** $CS \subseteq \bigcup REQ_i$ is a minimal set of requirements such that *inconsistent(CS)*. *CS* is *minimal* if there does not exist a conflict set *CS'* with *CS'* is a conflict set and $CS' \subset CS$.

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

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

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:
REQ_1 = {r_{11}: *trans = plane*, r_{12}: *1000* $\leq$ *cost* $\leq$ *1500*, r_{13}: *hotel#stars = 4*}
REQ_2 = {r_{21}: *food = Asian*, r_{22}: *dest = Asia*, r_{23}: *trans = train*}
REQ_3 = {r_{31}: *dest = Australia*, r_{32}: *hotel#stars* = 4, r_{33}: *att = 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 diagnoses 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

Table 1: An example about the impact of different diagnoses on current requirements of users. For example, if the minimal diagnosis set (r_{11}, r_{22}) is chosen then $user_1$ has to adjust one of all his/her requirements.

Users	(r_{11}, r_{22})	(r_{23}, r_{22})	(r_{23}, r_{31})
$user_1$	1	0	0
$user_2$	1	2	1
$user_3$	0	0	1
Least Misery	**1**	**2**	**1**

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

[1] A. Falkner, A. Felfernig, and A. Haag. Recommendation technologies for configurable products. *AI Magazine, AAAI*, 32(3):99–108, 2011.

[2] A. Felfernig, K. Isak, K. Szabo, and P. Zachar. The vita financial services sales support environment. In *AAAI*, pages 1692–1699. AAAI Press, 2007.

[3] A. Felfernig, M. Schubert, G. Friedrich, M. Mandl, M. Mairitsch, and E. Teppan. Plausible repairs for inconsistent requirements. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 791–796, Pasadena, California, USA, 2009.

[4] A. Felfernig, M. Stettinger, G. Ninaus, S. Reiterer, M. Jeran, A. Falkner, A. Tiihoren, J. Tiihonen, and G. Leitner. Towards open configuration. In *Workshop on Configuration*, pages 89–94. Novi Sad, 2014.

[5] A. Felferning, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger, and F. Reinfrank. *Advances in User Modeling*, chapter: Group Decision Support for Requirements Negotiation, pages 105–116. Springer Berlin Heidelberg, 2012.

[6] G. Fleischanderl, G. E. Friedrich, A. Haselböck, H. Schreiner, and M. Stumptner. Configuring large systems using generative constraint satisfaction. *IEEE Intelligent Systems*, 13(4):59–68, July 1998.

[7] A. Haag. Sales configuration in business processes. *IEEE Intelligent Systems*, 13(4):78–85, July 1998.

[8] A. Jameson. More than the sum of its members: Challenges for group recommender systems. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 48–54, New York, NY, USA, 2004. ACM.

[9] J. Masthoff. Group recommender systems: Combining individual models. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 677–702. Springer, 2011.

[10] G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, and W. Schanil. INTELLIREQ: intelligent techniques for software requirements engineering. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, pages 1161–1166, 2014.

[11] M. Richardson and P. Domingos. Building large knowledge bases by mass collaboration. In *Proceedings of the 2Nd International Conference on Knowledge Capture*, K-CAP '03, pages 129–137, New York, NY, USA, 2003. ACM.

[12] M. Stettinger. Choicla: Towards domain-independent decision support for groups of users. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, pages 425–428, New York, NY, USA, 2014. ACM.

[13] M. Stettinger, A. Felfernig, G. Leitner, and S. Reiterer. Counteracting anchoring effects in group decision making. In F. Ricci, K. Bontcheva, O. Conlan, and S. Lawless, editors, *UMAP*, volume 9146 of *Lecture Notes in Computer Science*, pages 118–130. Springer, 2015.

[14] M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, and M. Jeran. Counteracting serial position effects in the choicla group decision support environment. In *Proceedings of the 20th International Conference on Intelligent User Interfaces. IUI 2015 (ACM - San Francisco;)*, pages 148–157, 2015.

[15] M. Stumptner. An overview of knowledge-based configuration. *AI Commun.*, 10(2):111–125, Apr. 1997.

[16] E. Tsang. Foundations of constraint satisfaction, 1993.