An Approach of Crossover Service Goal Convergence and Conflicts Resolution

Yu Peng, Bing Li, Jian Wang, Zhengli Liu
School of computer science
Wuhan University, Wuhan, China
e-mail: {yupengdd, bingli, jianwang, zhengli_liu}@whu.edu.cn

***Abstract*—With the increasingly complex requirements of users in modern society, it is difficult to meet the complex goals of users by services from a single domain, and thus crossover service convergence, which aims to provide deep service integration and innovation across boundaries of industries, organizations, and value chains becomes an urgent need. Goal convergence is the first step to guide the process of crossover service convergence. However, there is currently no systematic theory and method to address this problem. In this paper, we propose a method to support goal convergence of crossover services. Firstly, two goal models from different domains are converged according to new user requirements. We then detect inconsistencies of goal models and resolve potential conflicts. We have developed a goal model convergence tool to help requirements analysts perform goal convergence in a visualized way. A case study is presented to show the feasibility of the proposed method.**

Keywords-goal convergence; crossover service; conflict resolution

I. INTRODUCTION

As an emerging service mode, crossover service has recently attracted wide attention from industry and academia. As a new economic phenomenon in the modern service industry, crossover service business is no longer limited to a single domain but expands to multiple domains. For example, OTA (Online Travel Agency) providers not only provide online hotel booking services but also integrate payment and insurance services. Crossover service usually involves mutual penetration and convergence of multiple industries, organizations, and value chains. Therefore, in the process of crossover service convergence, multiple boundaries, such as organization, business, value chain, and knowledge, are usually crossed [1, 2]. User requirements on crossover services are characterized by multiple actors, diverse goals, and variable processes. It is difficult to capture the potential needs or real intentions of users accurately. The reason for these problems lies in that each industry has its unique user requirements, and the requirements of different domains need to be deeply converged.

In a crossover service scenario, an service is initially designed for a single domain. Since user requirements are constantly evolving, some emerging user requirements or goals may fail to be achieved by the initial service. The *to-be-converged goal* refers to such a new emerging goal that cannot be satisfied by current domain goal models due to requirements evolution. Although *to-be-converged goals* cannot be satisfied in the current domain, there are potential solutions in other domains that can be leveraged. We call the domain which generates *to-be-converged goals* as a *subject domain*, and the domain which has potential solutions to satisfy the *to-be-converged goals* as an *object domain*. Therefore, this requires that the subject domain goal model must be deeply converged with an object domain goal model to meet evolving user requirements. In the process of crossover service requirements convergence, an important aspect is the convergence of goals in different domains, that is, the goal model in the subject domain and the goal model in an object domain must be deeply converged to satisfy the *to-be-converged goals* in the subject domain. Because the service requirements of each domain have their unique characteristics, and constraints in different business areas are also different, conflicts will inevitably arise during the convergence process. Therefore, an effective convergence method is urgently needed to guide goal convergence for crossover services.

In this work, we propose an actor-based goal convergence method that supports the deep convergence of goal models in different domains and automatically detects and resolves goal conflicts. This method can automatically match an object domain goal model according to the *to-be-converged goals* in a subject domain goal model, and automatically converge two goal models to satisfy the *to-be-converged goals*, thereby forming a unified goal model.

The contribution of this paper can be summarized as follows:

(1) We propose a crossover service goal convergence method to support the deep convergence of goal models in different domains, which can automatically discover the complementarities of the two goal models from different domains and converge them.

(2) We develope a prototype system to support the convergence of goal models from different domains in a visual way. The prototype can automatically check the grammar mistakes and identify conflicts in the goal models, which could help to improve the quality of goal models and reduce the burden of modelers.

(3) A conflict detection method is proposed to identify conflicts between goals, and it can effectively detect non-functional goal conflicts, operational goal conflicts, resource conflicts and other conflicts.

The rest of the paper is organized as follows. Section II introduces related research works and analyzes its deficiencies. Section III illustrates a motivation example. Section IV introduces the method of goal convergence and conflicts resolution. Section V shows our prototype system and a case study. Finally, Section VI summarizes the conclusions and introduces future work.

2642-939X/20/$31.00 ©2020 IEEE
DOI 10.1109/SERVICES48979.2020.00053

II. RELATED WORK

We introduce some related work on crossover service and goal conflict detection.

A. *Crossover Service*

Wu et al. proposed the concept of crossover services and the technical framework for realizing crossover service [1]. Yin et al. illustrated five challenges of the service convergence process: pattern convergence, ecosystem convergence, environment convergence, quality convergence, and value convergence using Alibaba's crossover service case [2]. However, they only discussed the framework theory of crossover services from a macro perspective and did not involve specific convergence methods.

Xi et al. proposed a scenario-based requirements model to model requirements of crossover healthcare service. Their proposed modeling method focuses on scenarios, processes and rules, that is, given a scenario, which processes are involved and which rules need to be satisfied [3]. Guo et al. proposed an interactive crossover service fusion approach based on microservice architecture to enable smooth and rapid integration of domain services [4]. Liu et al. analyzed the inner and external formation mechanism of crossover service and constructed a knowledge graph extracted from crossover news service for crossover event analysis [5]. They did not consider the convergence of crossover services from the goal level.

B. *Goal Conflict Detection*

Inconsistency management has been the focus of many recent works, and most of them are informal or semi-formal [6, 7]. Conflicts between non-functional requirements have also been studied [8-11]. Alrajeh et al. described a tool-supported technique for generating a set of obstacle conditions guaranteed to be complete and consistent with respect to the known domain properties. The approach relies on a novel combination of model checking and learning technologies. One important limitation of these approaches is that they are ineffective in situations that arise when multiple goals are conflicting [12]. Murukannaiah proposed a novel approach to capture inconsistencies of stakeholder goals and resolve goal conflicts. They used a structured analytic technique to elicit stakeholder goals, and employed a rational argumentation for determining goal conflicts to resolve them [13]. Degiovanni et al. proposed a new automatic method to calculate the boundary conditions of conflicting goals represented by Linear-Time Temporal Logic (LTL), using a tableaux-based LTL satisfiability process in [14]. They further evaluated the possibility of conflict [15], and presented a novel approach to automatically identify boundary conditions, using evolutionary computation [16]. But their methods are to detect conflicts in a single domain, without involving the detection of goal conflicts in crossover services.

Although existing research supports requirements modeling of crossover services, requirements convergence of different domains is not considered, nor the conflicts that may occur in the convergence process, such as boundary conditions conflict and inconsistency. Therefore, we need a systematic approach to support the convergence of goal models from different domains.

III. MOTIVATION EXAMPLE

Fliggy, as a famous OTA, is positioned as a leisure and holiday brand for young consumers and provides online hotel booking services. However, users sometimes will unsubscribe or change the hotel due to the uncertainty of the itinerary. To minimize the loss of users when unsubscribing or changing the hotel, *Fliggy* converges with *Ant Insurance* to launch the service of "No reason cancellation insurance," which can be used with hotel reservation services. On average, only 5% of the order cost is required to pay for the insurance, which means that they can enjoy 50% - 90% of the guarantee or prepaid room fee when they are unable to check in due to objective conditions or subjective reasons. By providing *no reason cancellation insurance*, it is estimated that the calls for cancellation of the hotel can be reduced by about 15%, and the operation efficiency of the hotel can be improved.

During the convergence process between *Fliggy* and *Ant Insurance*, some problems may occur. How to capture potential needs or real intentions of users to achieve deep convergence of the two domains? In the traditional OTA scenario, *Fliggy* only provides a single hotel booking online service. Now, we need to converge OTA, payment, insurance, hotel, and other services to meet multiple requirements. Inconsistencies and conflicts will inevitably arise during the convergence process due to different goals, business processes, and constraints of *Fliggy* and *Ant Insurance*. How to converge the user goals of different domains deeply and support the detection and resolution of conflicts has become an urgent problem.

IV. METHOD

Driven by the above motivation, we propose a crossover service goal convergence method, as illustrated in Fig. 1. Part A is goal model preparation. Part B presents how to converge goal models from various domains. Part C describes how to detect the inconsistencies of the converged goal model and resolve conflicts. Note that some steps are guided by domain experts.

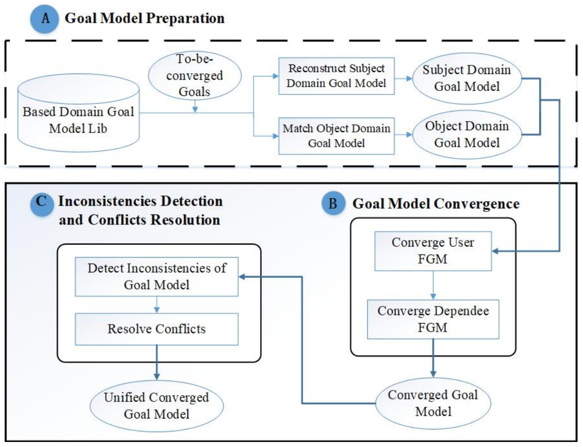

A Goal Model Preparation
Based Domain Goal Model Lib
To-be-converged Goals
Reconstruct Subject Domain Goal Model
Match Object Domain Goal Model
Subject Domain Goal Model
Object Domain Goal Model
C Inconsistencies Detection and Conflicts Resolution
B Goal Model Convergence
Converge User FGM
Converge Dependee FGM
Detect Inconsistencies of Goal Model
Resolve Conflicts
Unified Converged Goal Model
Converged Goal Model

Figure 1. Crossover service goal convergence method

A. Goal Model Preparation

To have a specific understanding of the goal model in our method, we give the following definitions.

Definition 1. *A goal model is a four-tuple <act, g, rel, res> where act denotes an actor, g is a goal, rel is the relationship, and res is a resource.*

Definition 2. *A fragmentation goal model is a fragment of goal model, which may contain one or more elements among actors, goals, relationship, and resources.*

The goal model contains many actors. Actors' intentionality is made explicit through the actor boundary, which is a graphical container for their goals together with their interrelationships. Goals in a goal model can be functional goals, non-functional goals, and operational goals. Relationships include interrelationships (AND, OR, Dependency, Exclusion, Help, and Hurt) between goals and social relationships (Dependency) between actors. An actor depends on another actor through resources, and the actor that provides resources is called the *dependee* [17].

To get goal models to be converged from a domain goal model library, we perform the following steps.

1) Reconstruct Subject Domain Goal Model: We reconstruct the subject domain goal model (SDGM) according to the *to-be-converged goal*. First, we analyze the SDGM to comprehend the actors, the goals of actors, and the dependencies between actors. The *to-be-converged goal* is essentially derived from the needs of users, so we add the *to-be-converged goal* in an appropriate place of the user fragmentation goal model (FGM). When the user actor has a *to-be-converged goal*, its dependee will also propose a goal accordingly, which is also a *to-be-converged goal*, to meet the user's *to-be-converged goal*. Then, we will add the *to-be-converged goal* of the dependee in the appropriate place in the appropriate user's dependee fragmentation goal model. Finally, we get the reconstructed SDGM.

2) Match Similar Domain Goal Model List: After reconstructing SDMG, we need to use the *to-be-converged goal* of user actor to match a high similarity goal model, which may have solutions to satisfy the *to-be-converged goals* in the based domain goal model library.

The matching steps are defined as follows:

a) Preprocess the *to-be-converged goal* of user actor and goals of goal models in based domain goal model library, which includes tokenization, word stemming, stop word removal, and word frequency statistics.

b) For each domain goal model *d*, calculate the importance of each word *w* and rank all words in descending order according to the word importance weight *Weight$_w$* to get the *Ranked Domain Keyword List* (*RDKL$_d$*). *Weight$_w$* is computed as follows.

$$\text{Weight}_w = \frac{\dfrac{N_{d,w}}{\max\limits_{wi\in d} N_{d,w_i}} \bullet \dfrac{\sum_{i=1}^{N_{d,w}} G_i}{N_{d,w}}}{\log\left(\dfrac{D}{D_w+1}\right)} \tag{1}$$

where $N_{d,w}$ represents the frequency of w in d, and $\max\limits_{wi\in d} N_{d,w_i}$ represents the maximum frequency of all words in d. D is the total number of domain goal model in Corpus, and D_w is the number of the domain goal model containing w. $\sum_{i=1}^{N_{d,w}} G_i$ represents the sum of the weight of w in the goal model. We think the importance of a word is related to its position in the goal model. Therefore, we give different weights G to the same word in different positions, and the higher level the goal is, the greater its weight is.

c) Calculate the cosine similarity between the *to-be-converged goal* of user actor g_u and $RDKL_d$ for each domain goal model. We generate the word frequency vectors of g_u and $RDKL_d$, denoted by $X[x_1, x_2, ..., x_n]$ and $Y[y_1, y_2, ..., y_n]$. The similarity value $Sim(g_u, RDKL_d)$ is computed as follows.

$$Sim(g_u, RDKL_d) = \frac{\sum_{i=1}^{n}(x_i \bullet y_i)}{\sqrt{\sum_{i=1}^{n} x_i^2} \bullet \sqrt{\sum_{i=1}^{n} y_i^2}} \tag{2}$$

The higher $Sim(g_u, RDKL_d)$ is, the more relevant g_u is to the domain goal model. We rank all goal models in descending order, according to $Sim(g_u, RDKL_d)$ to get the *Ranked Similar Domain Goal Model List* (*RSDL*).

3) Match Object Domain Goal Model: To get the object domain goal model (ODGM), we perform the following steps:

a) Search the *similar goal* of the first goal model in *RSDL*. A *similar goal* is a goal that is similar to the *to-be-converged goal* of the user actor, by calculating the cosine similarity of the goal names.

b) If a *similar goal* is refined by its subgoals, there may be solutions that can satisfy the *to-be-converged goal* of the user actor. And the goal model of the *similar goal* is viewed as ODGM. The FGM of similar goal solutions is shown in Fig. 2.

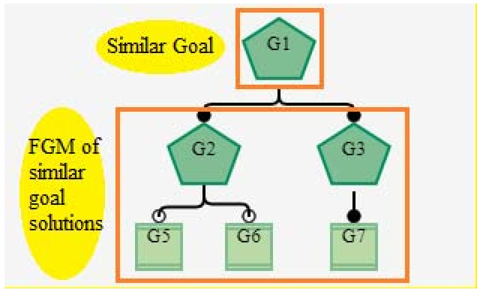


Figure 2. A sample graph of the FGM of similar goal solutions

c) If the goal model has no solution or the solution can not completely satisfy the *to-be-converged goal* of user actor, we repeat the above operations for the next goal model in *RSDL* until we get ODGM.

B. *Goal Model Convergence*

From Part A of Fig. 1, we get SDGM and ODGM, and then we need to converge these two goal models.

1) Converge user FGM: We converge the FGMs of user actors in two domain goal models, like Fig. 3.

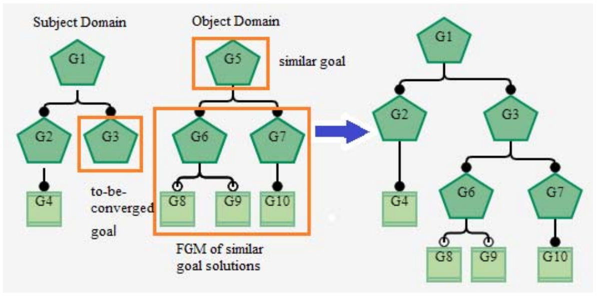


Figure 3. Converge user FGM

2) Converge dependee FGM: Usually, the realization of user goals depends on dependee actors of the user (named as "dependee" for simplicity). We need to converge dependee actor goals after converging user actor goals.

a) In the previous step, we got the FGM of similar goal solutions that can satisfy the *to-be-converged* goal. And the goals in FGM are satisfied by dependees. Therefore, we find the dependee set of FGM according to the dependencies in ODFM, and add the dependee set to SDGM.

b) In step "*Reconstruct Subject Domain Goal Model*", we already know the dependee of SDGM (Dep_s) and its *to-be-converged goal* (to distinguish it from the above, we call it "*sdg*"). As FGM of similar goal solutions can satisfy the *to-be-converged goal*, there may exist a dependee that can satisfy *sdg* in the dependee set of this FGM. Then, we match the dependee (Dep_o), which can satisfy *sdg* in a dependee set of FGM according to *sdg*, similar to the previous steps.

c) We converge the two dependees, and redefine the dependencies among users, Dep_s and Dep_o as that a user depends on Dep_s, while Dep_s depends on Dep_o.

d) Based on context and domain experts' suggestions, we modify the goal model and add new crossover service goals to satisfy *sdg*.

C. *Inconsistencies Detection and Conflicts Resolution*

Different domains may have different understandings of the same goal or resource. Therefore, we need to preprocess the goal model and use unified words to specify the same goals or resources to avoid ambiguity. Then we detect inconsistencies of the goal model and resolve conflicts. The inconsistency of a goal model can be divided into: non-functional goal inconsistency, functional goal inconsistency, and operational goal inconsistency.

1) Non-functional goal inconsistency: The definition of non-functional goal inconsistency is that the non-functional goals of the same goal are inconsistent in different domains. For example, a goal *g* requires high security in domain *A*, while fast realization in domain *B*, and conflicts may occur during convergence.

When there is a non-functional goal inconsistency, the TF-IDF method is used to calculate the weight of non-functional goals in different domains with the help of domain experts. The higher the weight is, the more important the non-functional goal is for the domain. And we choose the non-functional goal with higher weight.

2) Functional goal inconsistency: The definition of functional goal inconsistency is that the constraints of a functional goal conflict before and after convergence.

For inconsistencies caused by different constraints, we will refine functional goals with new constraints after convergence.

3) Operational goal inconsistency: The definition of operational goal inconsistency is that errors may occur when operational goals are executed repeatedly.

For the inconsistency caused by operation repeatability, we merge the repetitive operational goals.

V. Case Study

Based on the proposed method, we have developed a goal model convergence system (GMCS), which can construct domain goal models. The modeling interface is shown in Fig. 4, where the left side shows the modeling elements, and the right side is the modeling area. And we use different shapes to represent different elements and relationships. Fig. 5 shows examples of elements and relationships.

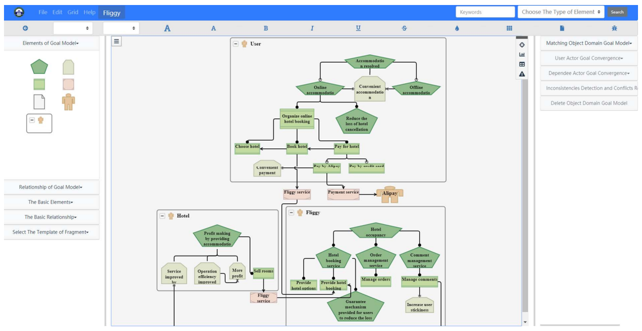

Fliggy
Elements of Goal Model
Relationship of Goal Model
The Basic Elements
The Basic Relationship
Select The Template of Fragment
User
Hotel
Fliggy
Alipay
Matching Object Domain Goal Model
User Actor Goal Convergence
Dependee Actor Goal Convergence
Inconsistencies Detection and Conflicts R
Delete Object Domain Goal Model

Figure 4. GMCS

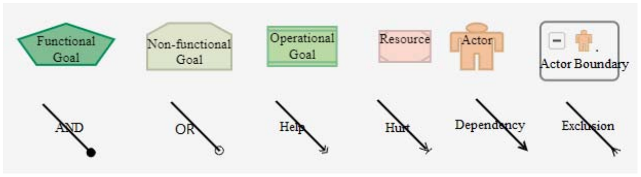


Figure 5. Example of elements and relationships

For crossover service goal convergence, goal models of two domains are imported into GMCS as input; then, GMCS converges two goal models; finally, the inconsistencies of the goal model are detected, and the conflicts are solved by users with the help of GMCS. Finally, we will obtain the converged goal model, which will be saved as an XML file or a JSON file.

To illustrate the effectiveness of the goal convergence and conflicts resolution method, we attempt to apply the proposed approach to the crossover service case of hotel and insurance convergence mentioned in Section III.

A. Goal Model Preparation

The hotel domain generates the *to-be-converged goals* “Reduce loss of hotel cancellation,” and it is regarded as the subject domain. By analyzing the hotel goal model, we found that there have four actors—user, online booking platform (*Fliggy*), hotel providers, and payment tool. The user can book hotels through the online booking platform, and pay through a payment tool. And users want to reduce the loss caused by canceling or modifying hotel reservations due to the uncertainty of the itinerary. The platform also intends to minimize user loss. So the reconstructed hotel domain goal model is shown in Fig. 6, and the *to-be-converged goals* of User and *Fliggy* are marked as *G1* and *G2*, respectively.

We use *G1* to match goal models in our domain goal model library. And we found there had solutions to satisfy *G1* in the insurance domain, which was regarded as *an object domain*. The insurance domain goal model, as shown in Fig. 7, has four actors—user, online insurance service providers (*Ant Insurance*), an insurance company, and a payment tool. Users purchase insurance through online insurance service providers for reducing loss and pay through the payment tool. The *similar goal* and the FGM of similar goal solutions are marked as *G3* and *FGMsg.*

B. Goal Model Convergence

1) Converge user FGM: Following Fig. 2, we turn *FGMsg* into the solution of *G1*.

2) Converge dependee FGM: According to the dependency in a goal model of insurance domain, the dependee set of *FGMsg* includes *Ant Insurance*, an Insurance company, and *Alipay.* Then, we add the dependee set to the hotel domain goal model. We found actor *Ant Insurance* can satisfy *G2.* Then we redefine the dependencies among User, *Fliggy,* and *Ant Insurance* as that a user depends on *Fliggy*, and *Fliggy* depends on *Ant insurance*. Based on context and domain knowledge, we generate the operational goal “Provide no reason cancellation insurance service” to satisfy *G2*.

C. Inconsistencies Detection and Conflicts Resolution

Firstly, we unify the names of goals and resources. Then, we detect the inconsistencies of the goal model and resolve conflicts.

After the convergence of the hotel domain and insurance domain, *Fliggy* launched the no reason cancellation insurance, which can improve the operation efficiency of the

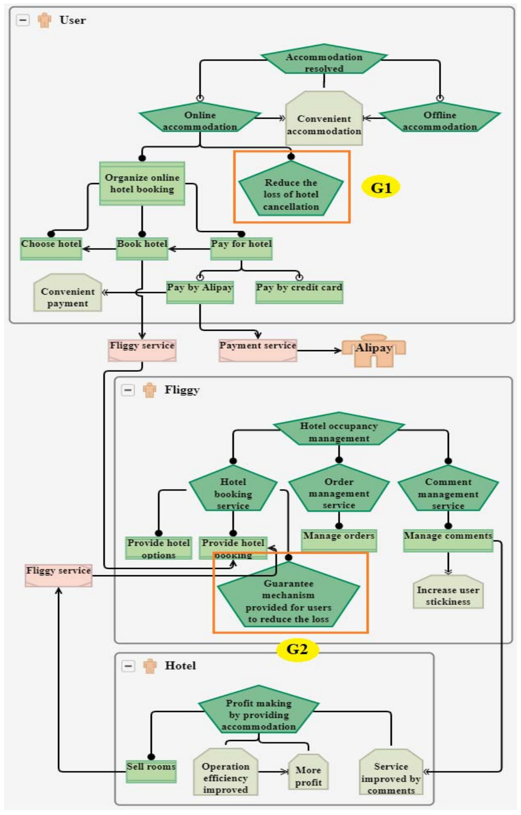
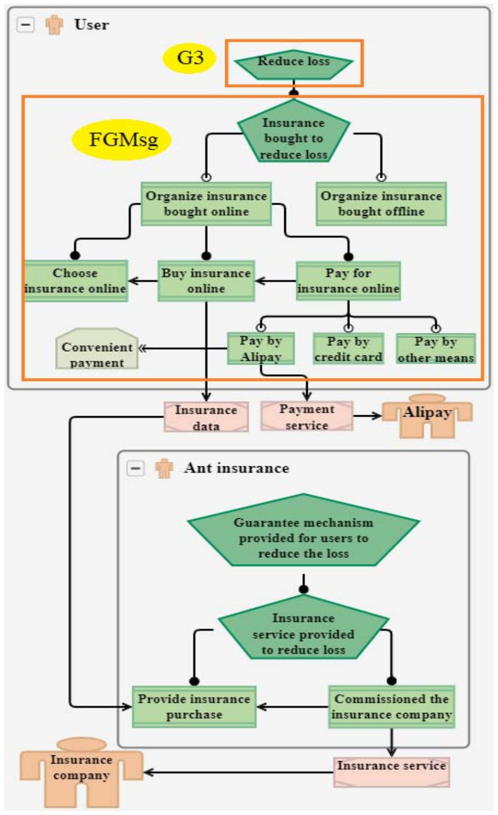


Figure 6. A goal model of the hotel domain


User
G3
Reduce loss
FGMsg
Insurance bought to reduce loss
Organize insurance bought online
Organize insurance bought offline
Choose insurance online
Buy insurance online
Pay for insurance online
Convenient payment
Pay by Alipay
Pay by credit card
Pay by other means
Insurance data
Payment service
Alipay
Ant insurance
Guarantee mechanism provided for users to reduce the loss
Insurance service provided to reduce loss
Provide insurance purchase
Commissioned the insurance company
Insurance company
Insurance service
Figure 7. A goal model of the insurance domain

hotel but also hurt its profits. The hotel has two non-functional goals “More profits” and “Operation efficiency,” which can not be satisfied at the same time. Combined with domain knowledge and TF-IDF, it is concluded that efficiency is more important. Before convergence, the goal “Insurance bought” can be refined by goal “Buy insurance offline”. But after convergence, the goal “Insurance bought” has a constraint that insurance can only be bought online and goal “Buy insurance online” is removed. Also, we found that users paid twice after convergence, and there is a duplicate payment operation, so we only keep one payment operation.

After the convergence and conflict resolution, we can obtain the final converged goal model, as shown in Fig. 8.

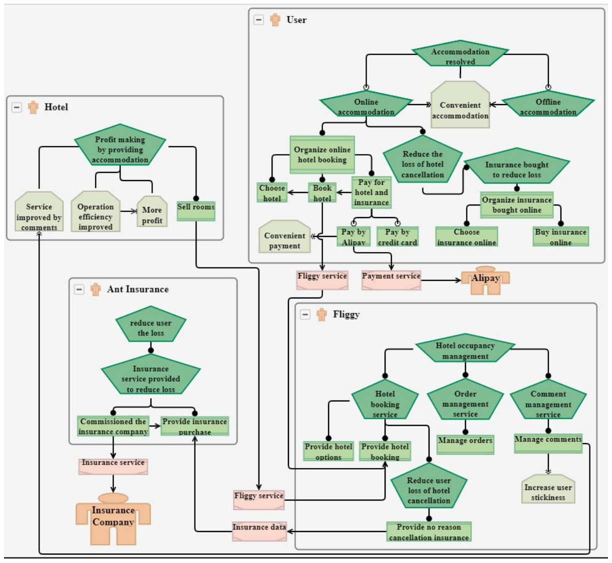


Figure 8. Unified Converged goal model

VI. Conclusion

In this paper, we propose a crossover service goal convergence method to support the deep convergence of goal models in different domains. Based on the proposed method, we developed a prototype system that supports the convergence of goal models from different domains in a visual way. We also propose an inconsistency detection method to identify inconsistencies between goals, which can effectively detect non-functional goal inconsistencies, functional goal inconsistencies, and operational goal inconsistencies. In addition, we provide a conflict resolution method to address conflicts during the convergence process.

For future works, we will further improve the conflict detection and resolution mechanism. Furthermore, we will also improve the GCMS system to support automatic goals models generation based on context.

Acknowledgment

This work is supported by the National Key R&D Program of China (No. 2017YFB1400602) and the National Natural Science Foundation of China (No. 61832014).

References

[1] Z. Wu, J. Yin, S. Deng, J. Wu, Y. Li, and L. Chen, “Modern Service Industry and Crossover Services: Development and Trends in China,” IEEE Transactions on Services Computing, vol. 9, no. 5, pp. 664-671, 2016.

[2] J. Yin *et al.*, “Crossover Service: Deep Convergence for Pattern, Ecosystem, Environment, Quality and Value,” 2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS), 2018, pp. 1250-1257.

[3] M. Xi, *et al.*, “A Scenario-Based Requirement Model for Crossover Healthcare Service,” 2019 IEEE World Congress on Services (SERVICES), vol. 2642-939X, pp. 252-259, 2019.

[4] S. Guo, C. Xu, S. Chen, X. Xue, Z. Feng, and S. Chen, “Crossover Service Fusion Approach Based on Microservice Architecture,” 2019 IEEE International Conference on Web Services (ICWS), 2019, pp. 237-241.

[5] M. Liu, Z. Wang, and Z. Tu, “Crossover Service Phenomenon Analysis Based on Event Evolutionary Graph,” 2018 International Conference on Service-Oriented Computing (ICSOC) Workshops, 2018, pp. 407-412.

[6] S. J. I. Herzig and C. J. J. Paredis, “A Conceptual Basis for Inconsistency Management in Model-based Systems Engineering,” Procedia CIRP, vol. 21, pp. 52-57, 2014.

[7] M. Kamalrudin, J. Hosking, and J. Grundy, “Improving requirements quality using essential use case interaction patterns,” 2011 33rd International Conference on Software Engineering (ICSE), 2011, pp. 531-540.

[8] I. J. Jureta, A. Borgida, N. A. Ernst, and J. Mylopoulos, “Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences, and Inconsistency Handling,” 2010 18th IEEE International Requirements Engineering Conference, 2010, pp. 115-124.

[9] C. Liu, “Ontology-Based Conflict Analysis Method in Non-functional Requirements,” 2010 IEEE/ACIS 9th International Conference on Computer and Information Science, 2010, pp. 491-496.

[10] C. Liu, “CDNFRE: Conflict detector in non-functional requirement evolution based on ontologies,” Computer Standards & Interfaces, vol. 47, pp. 62-76, 2016.

[11] R. M. Carvalho, “Dealing with Conflicts Between Non-functional Requirements of UbiComp and IoT Applications,” 2017 IEEE 25th International Requirements Engineering Conference (RE), 2017, pp. 544-549.

[12] D. Alrajeh, J. Kramer, A. v. Lamsweerde, A. Russo, and S. Uchitel, “Generating obstacle conditions for requirements completeness,” 2012 34th International Conference on Software Engineering (ICSE), 2012, pp. 705-715.

[13] P. K. Murukannaiah, A. K. Kalia, P. R. Telangy, and M. P. Singh, “Resolving goal conflicts via argumentation-based analysis of competing hypotheses,” 2015 IEEE 23rd International Requirements Engineering Conference (RE), 2015, pp. 156-165.

[14] R. Degiovanni, N. Ricci, D. Alrajeh, P. Castro, and N. Aguirre, “Goal-conflict detection based on temporal satisfiability checking,” 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE), 2016, pp. 507-518.

[15] R. Degiovanni, P. Castro, M. Arroyo, M. Ruiz, N. Aguirre, and M. Frias, “Goal-conflict likelihood assessment based on model counting,” 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE), 2018, pp. 1125-1135.

[16] R. Degiovanni, F. Molina, G. Regis, and N. Aguirre, “A genetic algorithm for goal-conflict identification,” 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE), 2018, pp. 520-531.

[17] F. Dalpiaz, X. Franch, and J. Horkoff, “istar 2.0 language guide,” arXiv preprint arXiv:1605.07767, 2016.