A Node-Merging based Approach for Generating iStar Models from User Stories

Chao Wu¹, Chunhui Wang*¹, Tong Li² and Ye Zhai¹

¹College of Computer Science and Technology, Inner Mongolia Normal University, Hohhot, China
²Faculty of Information Technology, Beijing University of Technology, Beijing, China

wuchao@mails.imnu.edu.cn, ciecwch@imnu.edu.cn, cieczy@imnu.edu.cn
litong@bjut.edu.cn

Abstract—User story is a widely adopted requirement notation in agile development. Generally, user stories are written by customers or users in natural language with limited form to describe user’s needs for the software system from their perspectives. However, since user stories are generally presented in a flat list, the relations derived from the user stories are difficult to capture. It reduces the understanding of the system as a whole. One solution to this problem is to build goal-oriented models that provide explicit relations among user stories. But extracting concepts and relationships from a large number of discrete user stories often take a lot of time for the agile development team. This paper proposes an iStar model generating approach based on node-merging from user stories. The method first extracts the iStar nodes from the semi-structured user stories, then uses a BERT (Bidirectional Encoder Representations from Transformers) model to measure the similarity between the nodes, and then nodes to be merged are identified and the edges between the iStar nodes are connected. Experiments are designed to illustrate the effectiveness of the proposed approach.

Index Terms—User story, iStar, Model construction

I. INTRODUCTION

Agile software development includes a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams [1]. It advocates adaptive planning, evolutionary development and delivery, and encourages rapid and flexible response to changes [2]. These features have led to agile methods to achieve remarkable success in software industry.

User story is a widely adopted requirement notation in agile development. Generally, user stories are written by customers or users in natural language with limited format that illustrates requirements from the user’s perspective. A general user story pattern relates a who, a what and possibly a why dimension, and uses keywords arranged these dimensions into one sentence (e.g., Cohn suggests a user story pattern [3]: *As a <type of user>, I want <some intention>, [so that <some reason>]*). Although a user story is short and simple descriptions, it normally consist of the following elements: *Role* (the aspect of who representation), *Goal/task* (a circumstance described by roles or specific things that must be done), and *quality* (expectations of the customer for the quality of the final product).

One challenge that is not addressed by user stories is to capture the relations of the user stories. For example, it is difficult to know the relations of decomposition and hierarchy between user stories. Such relations help the developers to better understand and structure the backlog items between user stories.

Goal-oriented modeling allows for the clear and explicit dependencies of goals to facilitate understanding of stakeholder needs, dependencies, etc. Some studies have observed that some concepts and relationships in user stories can potentially be aligned with goal models, and proposed the transformation approach from user stories to goal models [4]–[6].

Building goal models from user stories takes a lot of time and human overhead when user stories are large in scale. An automated goal-oriented model extraction approach is proposed in [7]. They propose 3 heuristic rules to identify refinement relationships between the what parts in user stories by using natural language processing (NLP) techniques. Their approach ignores the information in the why part of the user story and also ignores the relationship between the what and why parts. And this information is necessary to identify the model concepts and the hidden hierarchical relationships.

iStar model is a goal model that has been applied in many areas, e.g., healthcare, security analysis, eCommerce [8]. It focuses on the intentional (why), social (who), and strategic (how) dimensions. The concepts of iStar model correspond to the concepts of user story, and can express the composition and hierarchical relationships between user story concepts.

In [9], we proposed a preliminary framework for constructing iStar models from user stories and summarized 5 heuristic rules for extracting iStar elements from input user stories. In this paper, we focus on the refinement relationship between iStar nodes. The key to identifying the refinement relationships is a node-merging based approach. This approach pays attention to the nodes with similar description in the model, and combines them based on merging rules. The process of this approach consists of three steps: model node identification, node merging and edge identification. Model node identification extracts iStar nodes from a set of user stories by using NLP techniques. Node merging combines
the similar nodes by using a node similarity measurement approach based on BERT mode. Edge identification can find the edges between the iStar nodes by using three rules and finally obtains an initial iStar model. Several experiments are designed to evaluate our approach.

The rest of the paper is organized as follows. Section 2 outlines the relevant work. Section 3 details the process of building an initial iStar model from user stories. Section 4 performs this process with a small example. Section 5 evaluates the effectiveness of our proposed approach by using three real user story datasets from different domains. Section 6 summarizes the paper.

II. RELATED WORK

A. From user stories to goal models

There has been some research focusing on the transformation from user stories to goal models. However, since existing goal oriented modeling methods tend to be overly complex for non-technical stakeholders, some researchers propose some simplified goal models. Wautelet et al. [4] proposed a rationale diagram to build various trees of relating user story elements in a single project. Lin et al. [5] use a light weight goal net to model user stories. The goal net supports goal selection and action selection mechanisms and provides flexibility to task selection and process optimization in agile software development.

To build goal models from user stories, some studies define heuristic rules for the transition from user story templates to model elements. Jaqueira et al. [10] propose role in user story is mapping to actor in iStar model, action in user story is mapping to task in goal model, and goal is mapping to goal in goal model. Gune et al. [7] propose to automatically generate a goal model from a set of user stories by applying NLP techniques and 3 heuristics. The heuristics focuses on grouping the user stories around common objects or verbs.

Our approach also focuses on automated generation from user stories to iStar models. We propose to use node merging to extract refinement relationships between user stories. The key is to compare concepts that can be combined in the iStar model. In addition, we also focus on the identification of iStar nodes and their connected edges. Our goal is to generate a realistic model.

B. Similarity calculation of requirements

Similarity calculation is a key technique for identifying similar concepts and grouping similar user stories. At present, in the field of requirements engineering, there are mainly two types of methods: lexical-based methods and semantic-based methods [11].

Lexical-based methods regard the requirements as a sequence of words and concern with the similar of characters in word and their sequence of characters. There are many algorithms, such as Levenshtein distance [12], Jaccard [13], Hamming distance. Mihany et al. [11] proposed an automated system for measuring similarity between software requirements to identify reused components. This system uses Dice, Jaccard, and cosine similarity methods to measure the similarity between requirements in order to identify reusable components.

Semantic-based methods consider the meaning of words rather than characters of words. In order to compare the meaning of two words, corpus and some kind of language model are usually used. For example, some researchers use neural network models (e.g. Word2Vec, CNN, BERT) to generate word vector. Aran et al. [14] pre-trained BERT model to generate semantic textual representations to automatically identify software requirements from APP review. They trained seven different applications, and then an eighth application was used to evaluate the model. The results of the evaluation showed that the F1 scores were more than 40%.

In our work, we use the BERT model to generate word embedding for user stories. In order to accommodate the similarity comparison of short texts, we add the corpus of similarity pairs when pre-training the BERT model.

III. A NODE-MERGING BASED APPROACH

In this section, we introduce the proposed approach to generating iStar models from user stories, the process of this method consists of three steps (shown in Fig. 1): iStar node identification, node merging, and edge identification.

A. Node identification

In this phase, we identify iStar nodes and their types from a set of input user stories. There are three types of nodes: role type, intention type, and quality type. The detailed definitions of these three types can be found in [8], where the intention nodes includes goal nodes and task. Since it is difficult for automated methods to distinguish these two concepts, we do not make a distinction between these two concepts in this paper.

Given a set of user stories, we first extract user story components to construct iStar nodes. Each user story in the set has a fixed format. We use the following common user story pattern [3]: As a < type of user>, I want <some intention>, [So that <some reason>]. From this pattern, we use NLP techniques to extract their components: who, what and why. These components correspond to the part after the three fields “As a”, “I want to” and “so that”, respectively.

Specifically, we perform the following NLP steps: (1) Convert case: all nodes were converted in to lowercase. (2) Tokenization: nodes were split into a set of words by
removing spaces and commas. (3) Remove special characters: special characters are usually non-alphabetic and numbered characters. To remove noisy data, we remove special characters using the `re` regular expressions. (4) Remove stop words: stop words are removed like “a”, “to” etc and remove the keywords “As a”, “I want” and “So that”. And in the final step (5) Word Lemmatization: removes the affixes of words depending on part-of-speech (POS) tagging identifies the lexical properties and maps them to their root form to find the original form.

After applying the above steps to a set of user stories, we obtain the user story components, and apply the following 3 rules to extract role and intention nodes from user stories.

- Each who component maps to role type node;
- Each what component maps to intention type node;
- Each why component maps to intention type node.

In order to discover quality type nodes, we query the quality word in intention type nodes. To obtain the quality words, we collected 241 words from 4 related works [15]–[18] that contained a keyword list of non-functional requirements. The adjectives and adverbs were selected among these words and 58 words were obtained, such as easy, high, clear, friendly, early, ugly, fast, and so on.

**B. Node merging**

Node merging mainly includes three steps: representing node with BERT model, calculation of node similarity, and return of node pairs with high similarity.

We pre-train the BERT model [19] with STSBenchmark dataset of similarity scores (8628 sentence pairs) [20], and its parameters are set to that batch size is 16, epoch is 20, step is 1000, eventually saved fine-tune BERT model for generating the corresponding node vector. Equation 1 is used to measure the similarity between two vectors obtained by BERT model. The similarity result is a value between 0 and 1. For pairs with a similarity higher than 0.8 (see the section V-D for the details of threshold selection), we consider nodes that should be merged and merge them.

$$\cos(x, y) = \frac{x \cdot y}{|x| \cdot |y|}$$  \hspace{1cm} (1)

**C. Edge identification**

In this phase, we automatically identify the edges between nodes based on the identified iStar nodes and the merged nodes. There are three types of edges: means-ends, refinement and contribution. The means-ends connects two intentions nodes in a user story. The source is from what part in the user story. The target is from why part in the user story. The detailed definitions of refinement type and contribution type in this paper show in [8].

3 rules are used to identify the above three types of edges.

- **Rule 1**: Add an edge of type means-ends for two intention nodes from the same user story. The source is from what part in the user story. The target is from why part in the user story.
- **Rule 2**: Add edges of type refinement type for a merged node and its components (nodes).
- **Rule 3**: Add an edge of type contribution for a quality node and its corresponding intention node.

**IV. ILLUSTRATIVE EXAMPLE: ONLINE SHOPPING**

In this section, we use a case of Book Factory [21] to illustrate the proposed method. The Book Factory is a small online book purchasing system. In the online shopping system, users should be allowed to browse online book details, add books to shopping cart, complete online orders and query logistics information. On the other hand, for the system, it should calculate the order prices before customers pay for the orders. In addition, online payments are processed through the Ogone payment platform to improve payment security. For this group of user stories, we unified the user story template (using the key words of as a, I want to and so that), and the user story description information is shown in Table 1.

<table>
<thead>
<tr>
<th>US ID</th>
<th>Dimension</th>
<th>User Story</th>
</tr>
</thead>
<tbody>
<tr>
<td>US1</td>
<td>Role</td>
<td>As an owner I want my clients to be able to place orders online so that the customer-friendliness of our services increases.</td>
</tr>
<tr>
<td>US2</td>
<td>Feature</td>
<td>As a client I want to complete an order so that I can place it online.</td>
</tr>
<tr>
<td>US3</td>
<td>Feature</td>
<td>As a client I want to fill my online cart with products.</td>
</tr>
<tr>
<td>US4</td>
<td>Feature</td>
<td>As a client I want to pay my invoice so that I can complete an online order.</td>
</tr>
<tr>
<td>US5</td>
<td>Feature</td>
<td>As a system component I want to calculate the total amount of the order so that the invoice can be paid.</td>
</tr>
<tr>
<td>US6</td>
<td>Role</td>
<td>As a system component I want to pay my order online so that my invoice is paid.</td>
</tr>
<tr>
<td>US7</td>
<td>Feature</td>
<td>As a system component I want to process payments on the Ogone-payment platform so that the payment is secured.</td>
</tr>
</tbody>
</table>

Next, we describe the process of this case according to the method proposed in section III.

**A. Node identification from BookFactory**

After we input the user stories in Table I and execute the process of node identification, we extract the nodes with role, intention and quality type. Fig. 2 shows the results of this process. Here, nodes with intention type are graphically represented as ovals, nodes with role type are represented with circle, while qualities are represented as more curved cloud-like shapes.

**B. Merging similarity nodes**

Second, the BERT model-based approach generates node embedding for each node; then any two node embeddings are calculated using the cosine similarity algorithm to derive a similarity score; the pairs of nodes with our score greater
than a threshold are merged. Here, we list the merged nodes together for viewing ease. The result of this step is shown in Fig. 3.

C. Edge identification

On the basis of node identification and merging similarity nodes, we identify the edges between nodes according to rules, as shown in Fig. 4. Here, the contribution edges are represented graphically via a dotted line connecting the node type that is qualifies. The refinement edges are represented graphically via a solid arrow directed connecting the node type that is merged node.

V. EVALUATION WITH EXPERIMENT

We evaluate the effectiveness of our proposed approach in both quality node extraction and similarity node merging. 3 sets of user stories from different fields (game, business and education) are used to evaluate our approach. These 3 data sets are first manually labeled and then used to evaluate the performance of the proposed approach.

A. Dataset

a) GA: GA is from Ben-Gurion University of the Negev and is used for model building experiment [22]. The data set contains 21 user stories with correct syntax, which mainly describes the needs of setting up games and playing games in the development of game products.

b) WebCompany: webCompany comes from a Dutch company which makes custom web business applications [23]. The data set contains 79 user stories with correct syntax, which mainly covers the development of a web application focused on interactive story telling in 2014.

c) BADcamp: The BADcamp case study is obtained from the public user story requirements data set [24]. It is composed of 70 user stories with correct syntax. It mainly describes the needs of the educational platform in five scenarios: scheduled meeting, training course, sponsorship, blog and session.
B. Manually labeled datasets

For these 3 data sets, we conducted two sets of labeling experiments. One is for quality requirements, and the other is for finding similar pairs of user stories. The participants in these labeling experiments consist of 4 graduate students with software engineering related knowledge. Each student has one semester of software engineering class experience.

In order to ensure the availability of the labeled results, in each group of the labeling experiments, we divided the participants into two groups. The labeling process is as follows:

- Divide the data to be labeled into two parts;
- Divide the 4 students into two groups;
- Each group separately labels a part, and each student of the group labeled all the data in the part assigned to the group;
- Each group separately reviews the inconsistent data labeled by other group, and form a final labeled data set.

When labeling similar pairs, since there are a total of 170 user stories in the 3 data sets, 57630 pairs to be labeled will be generated. After preliminary research, we found that pairs with a similarity lower than 0.5 basically do not have the same semantics. So, in the labeling stage, we only let the students label pairs with a similarity greater than 0.5 and let them determine if the pairs are similar. A similarity greater than 0.5 have 443 pairs, then there were conflicting 71 pairs in the first round, then all conflicting pairs were identified in the second round, we finally obtained 49 similar pairs.

C. Metrics

We use precision, recall, and F1-score to evaluate the effectiveness of quality node identification and the merging node identification. The precision is used to evaluate the correctness of the node identification. The recall is used to measure the coverage. The F1-score is used to balance the accuracy and recall of the model. Equation 2, Equation 3, and Equation 4 are used to measure the precision, recall and F1-score, respectively. Where, TP represents true positive which is the nodes identified by automated approach, FP represents false positive which is the nodes identified by automated approach but not by labeled, and FN represents false negative which is the nodes identified by labeled but not by automated approach.

\[
\text{precision} = \frac{TP}{TP + FP} \quad (2)
\]
\[
\text{recall} = \frac{TP}{TP + FN} \quad (3)
\]
\[
F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (4)
\]

D. Experiment Result and analysis

Table II shows the precision, recall and F1 score to identify quality node and similarity pairs for each of the 3 user story sets. For quality nodes identification, the average precision, recall and F-value of our method based on quality words are 76.5%, 73.3%, and 72.1%, respectively. BADcamp data set has a higher precision (100%) than other data sets. The reason is that it only uses the “quickly” and “easily” quality words in non-function description. In other data sets, there are still some quality nodes are not found, the reason is the keyword list extracted by this paper less the related quality works and need to be expanded.

For similar node identification, we summarize the precision, recall and F-1 score of manually labeled semantically identical pairs with different thresholds (0.5-1, step size is 0.1) in the data sets, as shown in Figure 5. The result shows that we can select nodes with a similarity greater than 0.8 to merge.

![Fig. 5. Similarity result with different thresholds in three datasets](image)

VI. CONCLUSIONS AND FUTURE WORKS

Software requirements are usually expressed in natural language (such as user stories). Although the text is readable, it is difficult to provide an overall view of the most relevant entities and relationships. Especially in the case of a sharp increase in requirements, it is becoming more and more difficult to get the whole in mind. Model-driven development uses model to create a product and form a whole perspective to observe user needs. The iStar model is a widely used goal-oriented model. Its concepts can be aligned with user stories.

To construct iStar model from user stories, the nodes and edges in iStar model need to be identified from user stories. However, since the user stories provide by different stakeholders are scattered, the relationships between user stories...
are difficult to identify. This paper focuses on the refinement and contribution relationships between user stories. A node-merging based approach is proposed to identify the potential refinement relationships. A quality word list is built to find the non-function user story and to identify the contribution edges. 3 data sets are used to evaluate the effectiveness of the proposed approach.

As for our next step work, we plan to develop a prototype tool that implements our proposed framework. In addition, we will conduct extended experiments on more data sets to verify the effectiveness of the proposed approach.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (No.62162051), the Project of Beijing Municipal Education Commission (No.KM202110005025), the Natural Science of Foundation of Inner Mongolia Province (No.2021MS06024) and Inner Mongolia Normal University Graduate Students'research Innovation fund(CXJJS21158).

REFERENCES


---

**TABLE II**

**EFFECT OF USER STORY ANALYSIS OF THREE DATASETS**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Similarity of node pairs</th>
<th>Quality nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TP</td>
<td>FP</td>
</tr>
<tr>
<td>webCompany</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>BADCamp</td>
<td>7</td>
<td>4</td>
</tr>
</tbody>
</table>