Using Goal Net to Model User Stories in Agile Software Development

Jun Lin1,2
1School of Computer Engineering, Nanyang Technological University, Singapore
2College of Software, Beihang University, 37 Xueyuan Rd., Beijing, China
Jlin7@e.ntu.edu.sg; linjun@buaa.edu.cn

Han Yu1, Zhiqi Shen1, Chunyan Miao1
1School of Computer Engineering, Nanyang Technological University, 50 Nanyang Avenue, Singapore
han.yu@ntu.edu.sg; zqshen@ntu.edu.sg; ascymiao@ntu.edu.sg

Abstract—Agile methodologies use user stories to capture software requirements. This often results in team members over emphasizing their understanding of the goals, without proper incorporation of goals from other stakeholders or customers. Existing UML or other goal oriented modeling methods tend to be overly complex for non-technical stakeholders to properly express their goals and communicate them to the agile team. In this paper, we propose a lightweight Goal Net based method to model goal requirements in agile software development process to address this problem. It can be used to decompose complex processes into phased goals, and model low level user stories to high level hierarchy goal structures. Our preliminary analysis and studies in educational software engineering contexts show that it can improve agile team’s group awareness to project goals and, thus, improve team productivity and artifact quality. The proposed approach was evaluated in university level agile software engineering projects. It has achieved an improvement of over 50 percentage points in terms of the proportion of high quality user stories generated by students compared to the standard user story template used in Scrum.

Keywords—Agile Software Development; Goal Net; Software Engineering

I. INTRODUCTION

Agile Software Development (ASD) is designed to improve visibility and predictability of schedule performance, as well as overall software quality. ASD is considered as one of the major innovations in software development methodology of the last few years [1]. It 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. It advocates adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, face-to-face communication and encourages rapid and flexible response to changes [2, 3]. Those features have led agile methods to achieve a remarkable success in the software industry. Today, an increasing number of companies are adopting the ASD practice.

Instead of using requirement analysis and modeling activities in other plan-driven methodologies, agile team uses lean user story techniques to capture software requirements. However, there is more to requirements than user stories. For example, where is the business value in a user story? What are reasonable arguments for implementing a feature? Comparing to requirements, goals are a more subjective, but more important human factor in software development. Goals may be formulated at different levels of abstraction, ranging from high-level, strategic concerns (such as “enhancing end-users’ satisfactions” or “providing ubiquitous payment service” for an online shopping system) to low-level, implementation concerns (such as “speeding up the display of product search results” or “supporting payment with VISA credit card in the system”). Goals can also cover different types of concerns: functional concerns associated with software features and non-functional concerns associated with quality of service (e.g. safety, security, accuracy, performance). The multidimensional hierarchy characteristics of goals bring complexity and difficulty for modeling goals in the software development process.

Existing research works have used traditional requirement modeling methods, such as User Case method, to model goals [4, 5]. However, for ASD, UML is too complex to apply and implement, especially when non-technical stakeholders such as customers are involved [6]. In this paper, we propose a lightweight Goal Net based method to model structured goal requirements according to existing user stories produced in the agile process. The proposed approach was applied in university level agile software engineering education. It was shown to result in significant improvement in the number and quality of user stories generated by students compared to current approach.

II. RELATED WORKS

A. Goal-Oriented Requirement Engineering (GORE)

Requirement Engineering (RE) is a sub-field of Software Engineering. It is a vital phase in the overall development process, because the success of a software product depends on how well it fulfills the goals of the stakeholders. A number of goal oriented methods have been proposed to improve this process. The related research branch can also be referred to as Goal-Oriented Requirement Engineering (GORE) [7].

During the past two decades, GORE researchers proposed a set of approaches using the concept of goal to specify and explore the different objectives of systems, organizations, and users. Examples of such approaches include KAOS [8], 1* [9], GBRAM [10], Framework NFR [11], GQM [12], AGORA [13], Goal! Strategy MAP [14],
Goal-scenario coupling: CREWS-L'Ecritoire [15], and Goal Net [16] etc. These approaches explore the goals of the stakeholders and the actions required to achieve their objectives. Most of them attempt to link requirements to customers' goals, as they believe that using goal-oriented requirement model has the following advantages:

- Object models and requirements can be derived systematically from goals;
- Goals provide the rationale and humanity for requirements;
- A goal graph provides vertical traceability from high-level strategic concerns to low-level technical details; it allows evolving versions of the system under consideration to be integrated as alternatives into one single framework;
- Goal model can provide the right abstraction level at which stakeholders can be involved for important decisions;
- The goal hierarchy model provides a comprehensible structure for the requirements document;
- Alternative goal refinements and task assignments allow alternative system proposals to be explored;
- Goal formalization allows refinements to be proved correct and complete.

B. The Goal Net Modeling Method and Tools

Most of current GORE methods are ad-hoc or overly complex to be applied in ASD. However, as a practical and lightweight tool, Goal Net modeling theory is the most suitable for our purpose. The Goal Net theory was proposed in [16, 17]. It was designed to model and design goal-oriented agent systems. According to [17], a Goal Net model consists of four basic concepts: states, transitions, arcs and branches. There are two types of states in a Goal Net: composite state and atomic state. An atomic state represents a single state which cannot be split. A composite state represents a goal that may be split into sub-states. States are interconnected by transitions. A transition primarily shows the relationship between the states it joins, specifying the tasks to be performed in order to transit between goals. There are four kinds of relationships between two states. They are 1) sequence, 2) concurrency, 3) choice, and 4) synchronization [17].

Goal Net also serves as a practical methodology for engineering agent oriented software systems [16]. The methodology covers the entire life cycle of the agent oriented system development process from requirement analysis, architecture design, and detailed design to implementation [18]. Based on this methodology, the Goal Net Designer, which is an integrated tool and Development Environment (IDE) for modeling agent behavior based on Goal Net model, was proposed in [19]. The Goal Net Designer provides a way for users to simplify the various stages of agent design. It also can be used by the Multi-Agent Development Environment (MADE) automatically to create intelligent agents. [19]

The Goal Net model provides a rich set of relationships and selection mechanisms to form a dynamic and highly autonomous agent problem-solving framework. It supports goal selection and action selection mechanisms [20, 21]. Goal selection is used in the “choice” transition relationship and is affected by factors such as goal achievement and cost. Action selection, on the other hand, provides sequential, rule-based or probabilistic inference execution for the tasks specified in a transition.

Based on the Goal Net methodology, a hierarchical Goal Net model for a typical agile software development process, such as Agile United Process (AUP) or Scrum process, was proposed in [22].

As a modeling method, Goal Net is a novel way to present the overview of system goals. Its goal selection and action selection mechanism can also provide flexibility to task selection and process optimization in ASD.

III. ANALYSIS

A. Current Roles in the AUP Process

Generally in an agile team, there are several roles, which are named differently in different ASD methods. Roles are not positions, any given person can assume one or more roles, and can switch roles over time. Any given role may be performed by zero or more people at any given point in a project.

Figure 1 shows the general structure of an agile team. The core agile team includes developers who are led by the team lead, working closely with a product owner to build software in an iterative and incremental process. Sometimes an architecture owner is also involved. The supporting roles include technical experts, domain experts and independent testers.

![Fig. 1. The organizational structure of a typical agile team](image)

Other stakeholders include anyone who is a direct user, indirect user, manager of users, senior manager, operations staff member, the gold owner who funds the project, support IT staff member, auditors, program manager, developers working on other systems that integrate or interact with the one under development, or maintenance professionals potentially affected by the development and/or deployment of a software project.

By analyzing current roles in ASD, we can further consider their different goals during the development process.

B. Current Requirement Management in the AUP Process

Agile team uses user story to depict requirements. A user story may include functions, features, enhancements, bugs, and refactors. A user story is a short, simple description told
from the perspective of the person who desires the new capability, usually a user of the system. User stories typically follow a simple template:

As a `<role>`, I want to `<goal/desire>` [so that `<benefit>`].

Elements in the angled brackets are to be specified and elements in the square brackets are optional. [23]

Product Owners are primarily responsible for user stories. But others can also contribute to them. In practice, many users write user stories. The first requirement may come from an end user. Others, such as the product owner, architect, scrum master, or business analyst etc., can update them.

User stories are often written in a non-technical manner from the perspective of an end user. This user story will be further defined. After fine tuning the stories to an extent that it can be put to review to the agile team, the entire agile team will work on these stories to understand it. Any technical constraints or limitations need to be noted down and presented to the customer. Finally, the user stories will be stored in the product backlog, and divided into small tasks for ASD team members to implement. The product backlog is a prioritized list of functionalities that will be developed into a software product or service.

One of the benefits for agile user stories is that they can be written at varying levels of detail. We can write user stories that cover large number of functionalities. These large user stories are generally known as “epics”. Here is an example epic for an online B2C marketplace services:

As a customer, I want to pay via mobile phones so that I can buy goods on mobile phones quickly.

As an epic is generally too large for an agile team to complete in one iteration, it needs to be split into multiple smaller user stories first. The epic above can be split into many smaller user stories, including for example:

As a VIP customer, I want to pay cash on delivery so that I can buy goods on mobile phones without paying immediately.

As a common customer, I want to be able to pay by credit card so that I can buy goods on mobile phones quickly.

Table I-III show examples of user stories at different levels of abstraction.

**TABLE I. AN EXAMPLE OF USER STORY LIST**

<table>
<thead>
<tr>
<th>ID</th>
<th>As a/an</th>
<th>I want to...</th>
<th>so that...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>visitor</td>
<td>Easily search goods on mobile phones</td>
<td>I can find my favorite goods with no digital divide</td>
</tr>
<tr>
<td>2</td>
<td>visitor</td>
<td>Easily sort the search results</td>
<td>I can find my favorite goods according quickly</td>
</tr>
</tbody>
</table>

**TABLE II. DETAILED USER STORY LIST EXTENDED FROM TABLE I**

<table>
<thead>
<tr>
<th>ID</th>
<th>As a/an</th>
<th>I want to...</th>
<th>so that...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>visitor</td>
<td>Easily search goods on mobile phones</td>
<td>I can find my favorite goods with no digital divide</td>
</tr>
<tr>
<td>1.1</td>
<td>visitor</td>
<td>Search goods on mobile phones by voice input</td>
<td>I don't need to type</td>
</tr>
<tr>
<td>1.2</td>
<td>visitor</td>
<td>Search goods on mobile phones by clicking a category</td>
<td>I can find my favorite goods according to category what I'm choosing</td>
</tr>
<tr>
<td>2</td>
<td>visitor</td>
<td>Easily sort the search results</td>
<td>I can find my favorite goods quickly</td>
</tr>
<tr>
<td>2.1</td>
<td>visitor</td>
<td>Sort the result according to price</td>
<td>I can find my favorite goods at bargain prices</td>
</tr>
<tr>
<td>2.2</td>
<td>visitor</td>
<td>Sort the result according to location</td>
<td>I can find my favorite goods near me</td>
</tr>
<tr>
<td>3</td>
<td>customer</td>
<td>Quickly pay via mobile phones</td>
<td>I can buy goods on mobile phones quickly</td>
</tr>
<tr>
<td>3.1</td>
<td>VIP customer</td>
<td>Choose to pay cash on delivery</td>
<td>I can buy goods on mobile without paying first</td>
</tr>
<tr>
<td>3.2</td>
<td>common customer</td>
<td>Choose to pay by credit card</td>
<td>I can buy goods and pay on mobile quickly</td>
</tr>
</tbody>
</table>

**TABLE III. DETAILED USER STORY LIST WITH TASKS**

<table>
<thead>
<tr>
<th>ID</th>
<th>As a/an</th>
<th>I want to...</th>
<th>so that...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>visitor</td>
<td>Easily search goods on mobile phones</td>
<td>I can find my favorite goods with no digital divide</td>
</tr>
<tr>
<td>1.1</td>
<td>visitor</td>
<td>Search goods on mobile by voice input</td>
<td>I don't need to type</td>
</tr>
<tr>
<td></td>
<td>Task 1.1.1</td>
<td>Investigate voice input solutions for mobile phones</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task 1.1.2</td>
<td>Design a new User Interface (UI)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task 1.1.3</td>
<td>Choose one solution and implement it on mobile phones</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Task 1.1.4</td>
<td>Integration and unit test</td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td>visitor</td>
<td>Search goods on mobile phones by clicking a category</td>
<td>I can find my favorite goods according to category what I'm choosing</td>
</tr>
</tbody>
</table>
A. Problems

From the above analysis, we can see that the ASD user story method has provided a way to present customers’ goals via a *I want to* clause in the template. To some extent the hierarchy of user stories also reflects the hierarchy of goals. However, in practice many agile teams, especially novice teams, often ignore this hierarchical structure between goals when the project enters the detailed iterative development process. We have conducted an eight week group-based agile software engineering project as part of an undergraduate course work in Beihang University from 01/04/2013 to 31/05/2013. We collected 142 user stories and they were split into 726 tasks by 20 teams with an average team size of 6 persons. By analyzing the user stories data, we discovered that about 4/5 of them ignored the hierarchical relationships, as most of them just described the developers’ goals without considering users/other stakeholders’ goals.

In addition, current approaches are based on nature language expressions, which can be ambiguous. This is especially true when requirements are not so clear for customers themselves. This situation occurs more often in the beginning of the software projects. If there is a well-defined modeling theory to support agile teams to model hierarchical goal structures, it will better facilitate the teams to understand the requirements. A graphical goal model can be more intuitive and effective for product owners to inspect and understand stakeholders’ real goals, too.

B. Proposed Light Weight Method

Based on Goal Net theory, the proposed method consists of the following three steps: 1) Defining Goals: to define stakeholders’ high level goals by interviews; 2) Identifying Goals: to identify different goals hidden in user stories; and 3) Modeling Goals: to model goal structure according to Goal Net. The improvement will not incur extra effort for the PO or other ASD team members.

C. Example

**Assumption:** an agile team is developing a mobile shopping app for iPhones. Before a new iteration/sprint, some elderly end users felt the working system was not easy to use. Therefore, they provided the product owner with new feedbacks. Based on these feedbacks, the PO has created some user stories, part of backlog is been listed in Table I, II and III. Then, the PO follows the proposed 3-step method to model the hierarchical goal structure.

**Input:** user story lists in Table I, II and III (partially)

**Output:** a Goal Net model

**Modeling Steps:**

*Step 1. Defining Goals*

After interviewing the elderly users, the PO knows that they actually want to enhance the user experience of the working system. The PO then clusters the user stories into four high-level goals: 1) improved user interface, 2) clearer navigation system, 3) friendly help system, and 4) simplified work flow. Therefore, the PO firstly designs an initial high-level Goal Net model shown in Figure 2:

![Initial Goal Net model without detailed goals/requirements](image)

**Step 2. Identifying Goals**

Based on the initial high level Goal Net model, the PO needs to assign high level goals to each user story according to the *I want to* clause.

**Step 3. Modeling Goals**

After the above two steps, the PO is able to build the full Goal Net diagram which is as shown in Figure 3. In this case, the full Goal Net model has four levels. In level 2, two goals of “*Easily search goods on mobile phones*” and “*Easily sort the search results*” are linked to their parent goal “*Improved user interface*” in level 1. The goal of “*Quickly pay on mobile*” is linked to its parent “*Simplified work flow*” in level 1. The goals depicted in the “*I want to*” clause of sub-user stories are linked to their respective parent goals in level 2.

Necessary sequence transitions, concurrency transitions, and synchronization transitions are included in this Goal Net model. For example, the transition between two sub-goals in level 2, “*Easily search goods on mobile phones*” and “*Easily sort the search results*”, are sequence transitions. This is because only after executing “*getting search result activities*” can the implementations related to the goal of “*Easily sort the search results*” start. Three pairs of “*Implementing activities*” under level 2 are concurrency transitions, as after achieving two sub-goals, the process needs to be synchronized to reach their parent goal, so that these two “*Implementing activities*” can be executed by two or more developers concurrently.
The transition activity in the Goal Net represents a list of tasks for implementing the goals (user stories). For example, the “Implementing activity” for the goal “Search goods on mobile phones by voice input” can be depicted by the following GET card [17].

**Goal 1.1: Search goods on mobile phones by voice input**

<table>
<thead>
<tr>
<th>Environment Variables</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Man Power (location, role): David (LA, UI Designer), Michael (BJ, Developer), Grace (SG, Developer) 2. Device: iPhone5, iPhone 4s, HTC One 3. Third-part mobile voice input solutions - Speech Input API for Android (Google) - Xunfei Voice recognition API (USTC Xunfei)</td>
<td>1.1.1 Investigate voice input solutions for mobile phones 1.1.2 Design new UI and implement it on mobile phones 1.1.3 Choose one solution according to price 1.1.4 Integrate into the working system</td>
</tr>
</tbody>
</table>

The final Goal Net model can be used in subsequent iterations/sprints and can be updated with more user stories. It can help an entire ASD team review the user stories during iteration/sprint planning meetings and verify the working software retrospectively.

**V. EVALUATION**

Since 2011, the College of Software at Beihang University has introduced Scrum into practical software engineering courses to train development and management skills for undergraduate students. These students were divided into teams with an average team size of 6 persons to carry out an 8-10 week group-based software development project. There are typically around 20 teams during each semester. All teams possess similar skill levels and backgrounds and adopt the Scrum process during the project.

At the end of each semester, the quality of the user stories created by the teams were graded by the course instructors with a score representing how well they reflect stakeholders’ goals. Table IV shows the comparison result between class 2011, 2012 and 2013.

**TABLE IV. COMPARISON OF TEACHING RESULTS**

<table>
<thead>
<tr>
<th>Class</th>
<th>2011</th>
<th>2012</th>
<th>2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>Used Goal Net to model user stories</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Total number of student teams</td>
<td>26</td>
<td>17</td>
<td>23</td>
</tr>
<tr>
<td>Total number of user stories</td>
<td>189</td>
<td>118</td>
<td>135</td>
</tr>
<tr>
<td>The average number of user stories per team</td>
<td>7.3</td>
<td>6.94</td>
<td>5.87</td>
</tr>
<tr>
<td>Standard deviation of the number of user stories per team</td>
<td>3.3</td>
<td>3.37</td>
<td>1.62</td>
</tr>
<tr>
<td>The proportion of High Quality User Stories reflecting stakeholders’ goals</td>
<td>21%</td>
<td>19%</td>
<td>74%</td>
</tr>
</tbody>
</table>

From Table IV, we can see that as we introduce proposed method and Goal Net diagram into the class 2013, the standard deviation of number of user stories per team decreased significantly and the proportion of high quality user stories increased significantly. Overall all teams understood more clearly the stakeholders’ goals and recorded them as user stories, but for class 2011 and 2012, they put too much emphasis on non-core goals. The proposed approach achieved an improvement of over 50 percentage points in terms of the proportion of high quality user stories generated by students compared to the standard user story template used in Scrum.

**VI. CONCLUSIONS**

In this paper, we proposed a novel goal-oriented method to model user stories in Agile software development process.
By modeling user stories via Goal Net diagram, we provide a clear overview and a new perspective to allow ASD team members to see the requirements. Through studies in an educational context, we have evaluated our proposed approach against current industry practice. The results showed that the number and quality of the user stories are significantly improved with the use of the proposed approach.

VII. ACKNOWLEDGMENTS
This research is supported in part by Interactive and Digital Media Programme Office, National Research Foundation hosted at Media Development Authority of Singapore (Grant No.: MDA/IDM/2012/8/8-2 VOL 01).

REFERENCES