ABSTRACT
Software reuse on a large scale has been a goal of the software industry since the late sixties. Based on this idea, we did a survey with 38 companies to investigate the reuse practices in the Brazilian industry. The key finding of the survey was that requirement reuse is important, but it is not done very often in practice; the main reason for that is the lack of tool support. On the other hand, use case modeling is a popular and widely used technique for capturing and describing the functional requirements of a software system. The reuse of use case specifications within a specific area of knowledge, rather than building them from scratch, is a strategy that not only can increase the quality of the new specifications but also can help their construction. In this paper, a web-based tool that provides support for the reuse of use cases specification is presented to address this problem. The tool, called 4REuse, is evaluated through a social networking service case study.

KEY WORDS
Requirements Engineering, Use Cases, Requirements Reuse, Requirements Visualization, Traceability.

1. Introduction
Software development has changed considerably in the last decades to meet the needs of the competitive market, which requires software to be developed under stronger constraints of cost and time. The traditional way to develop software – from scratch – no longer meets the demands, and software reuse is considered a potential approach to address these challenges [1].

Software reuse on a large scale has been a goal of the software industry since the late sixties. Several approaches have been proposed in the literature. [6]. These approaches can be broadly categorized into three main types: frameworks, product lines and patterns.

Software product line engineering aims at developing a family of systems [9]; it guides the development of products from existing artefacts. Patterns propose a standardized documentation of a recurring solution, already well tested, providing an explicit specification of interactions between classes and objects [10], among others. Most of these approaches emphasis is on code reuse, although coding represents only about 20% of the total cost of software development [2].

The number of approaches proposed for requirements reuse has increased more recently. Examples of requirements reuse approaches are described in [11,12,13,15]. In [11], a method for creating generic requirements fragments aiming their reuse is presented. To do this, authors integrate complementary reuse techniques that deal with the analysis and documentation of a fragment. In [12] a use case modelling approach is described, and a set of rules to specify use cases are proposed, in order to reduce ambiguity and to facilitate automated analysis. In [13], an approach to support the specification of use cases that is based on the creation, use and maintenance of a catalogue is described.

Two complementary reuse approaches can be considered: building for reuse and building with reuse [18]. The first one is related to the identification of reusable units of knowledge, their representation in a concisely and abstract way, and their storage in a knowledge base, preferably indexed and classified. The second approach corresponds to seeking reusable knowledge, followed by its adaptation to the required situation, that is, the combination of the found knowledge with the new project.

In this paper a web-based tool called 4Reuse, available at [14], is proposed in order to provide support for building with reuse and for reuse, based on use case fragments [15]. The tool aims at solving the problem: “How to support the specification of use cases based on the reuse concept?” The tool helps building new applications, considering the use and adaptation of existing reusable fragments available through a catalogue specified using the tool. For details see Figure 1, developing with reuse (upper flows) and for reuse (bottom flow).

Figure 1. Specifying use cases in 4ReUse based on a Catalogue.

Observe that a visualization support was integrated in the tool; this allows a better visualization, understanding and
traceability of information in the catalogue, and also in projects being defined (existing artifacts and their relationships). The motivation was that: if a reusable artifact is easy to understand, developers are more likely to reuse it.

A Social Network project is used to illustrate the proposed approach and tool. This project was developed by a group of graduate students as module’s coursework. They considered the concept of social network as social structure, composed of people or organizations connected by one or more types of relationships that share common values or goals [8]. Therefore, the proposed social networking service focuses on facilitating the building of social networks or social relations among people who, for example, share interests, activities, backgrounds, or real-life connections.

The remaining of the paper is organized as follows. Section II discusses the state of the practice of software reuse in the IT Brazilian industry, using the results of a conducted survey. Section III provides an overview of techniques applied in the proposed approach. Section IV introduces the approach for building for reuse, and section V details the approach in terms of how to build with reuse. Finally, section VII presents the paper’s contributions and discusses future work.

2. A Software Reuse Survey in the Industry

Software reuse can help to reduce development time and cost, improve software quality, and also improve maintainability of applications. Despite the potential benefits from an effective software reuse practice, reusability is not always easy and may even sometimes be impossible to apply. Many people prefer to build from scratch, rather than needing to adapt existing artifacts constructed by others. There are also several issues when one considers good practices for reuse, such as: choosing what should be reused, identifying better ways to document reusable abstractions, defining ways to find what you want, and facilitating the adaptation of existing selected reusable fragments in new applications.

Aiming at better understanding the practice of reuse in software development companies, a local survey was conducted through a questionnaire applied into 38 companies (including 51 different people) in the period from January to March 2012 [5]. The goal was gathering information about software reuse, addressing professionals, most having between 5 to 10 years of experience in software development. See Figure 2 for details.

Among those who answered the questionnaire, we have the following professional positions: system and testing engineers, analyst (system and business networks), designers, IT managers and software developers.

This research aims at verifying, among other things: the individual's experience in software reuse, the reuse rate achieved during software development; challenges; the main results when considering the reuse of software; whether it is worth using or not; how the team works in documenting and reporting of requirements; and also the average number of requirements generated per system.

From the analysis of the questionnaires, it was observed that most workers reuse between 25%- 50% of their artefacts, and that this occurs mainly in the coding step, see Figure 3. The survey also demonstrates that professionals make up the documentation and reporting of requirements through one of the 4 ways, presented in Figure 4, and that the average number of requirements in each project is between 100 and 500.
Use Case specifications are usually written in a natural language, allowing ambiguous specifications; also reuse is not made in a systematic way, making the activity boring and very repetitive, as it was observed in the survey made and in literature.

3. Background Overview

This section presents an overview of the background required to understand the proposed approach, which applies features diagrams together with use case fragments for having a more representative expression of the involved functionality and variability. Also, the integration of interactive visualization support, for the involved data, enhances the problem of the understanding and the traceability between use cases, features and the design.

3.1 Features Diagrams

Product lines use the concept of features, system properties or relevant functionalities, to capture common characteristics and variability of software products in the same family. Features are organized into one or more diagrams, usually in the form of a tree, where is pointed the mandatory, optional, and grouped features [16]. Requirements for product families are expressed in terms of commonality and variability [3], this distinction allows early identification of the software architecture and opportunities for software reuse. Despite feature diagrams have not been so explored to specify requirements, they allow a simple and objective representation of the involved variability.

3.2 Use Case Fragments

The Use Case Fragments, proposed in [15], address reuse using decomposition of goals. These fragments are the sub-goals of a use case that can be reused in other use cases, in different applications. A Use Case Fragment is defined as a common structure for reuse, consisting of interactions between the system and its actors, including alternative flows. A fragment is detailed through a model (template) that has points of customization to be replaced by terms of the domain used. Fragments are focused on achieving a sub-generic goal. The main objective is achieved at the end of an implementation, when the whole use cases were developed [13].

A Use Case Fragment is defined by several elements:

- Name: identifies the fragment and represent its purpose;
- Sub-objective: explains the purpose of using the fragment, so you can compare your sub goal with the specific goal of the fragment, and decide whether or not to use it;
- Basic Flow: a sequence of steps which details the interactions, present in the fragment, required to achieve its sub-goal. Each step has a template that can also have customization points. During the fragment customization these points will be replaced by business objects or system’s actors. Steps can be classified as optional or not;
- Alternative Flows: steps which have a reference to a step of the basic flow, and also with an explanation of when the alternative flows should be implemented as an alternative sequence of steps. They have customization points; and also can be optional
- Data Structures: used to describe the properties of business objects; it can also has customization points and can be optional;
- Rules: describe the system’s business rules; they have customization and explanation point; they can be optional.

Use Case Fragments are more flexible then simple use cases since they include configuration points. An example of a reusable use case fragment for the Social Network project is presented in Table I, and further instantiated in section V, through the 4REuse tool.

3.3 Visualization Tools

The traditional way to display artifacts is a hierarchal tree; however, this does not scale well for large amounts of data. An alternative is to use a radial layout as used by Sunburst [3], a space-filling visualization tool. In [3], the authors applied Sunburst in two empirical studies to evaluate its effectiveness and utility. The results showed that the participants strongly preferred Sunburst, considering its ability to convey structure and hierarchy; it was also considered an approach to improve representation of traceability links [4]. In Sunburst, items in a hierarchy are laid out radially, see Figure 5. With the top of the hierarchy at the center, and the deeper leaves farther away from the center. The angle swept out by an item and its color corresponds to some attribute of the data. For instance, in a visualization of software artifacts of a project, the color may correspond to the artifact type.

Figure 5 shows the relationship between artefacts of our case study. At the center is the Social Network project, displayed in a very light gray color. The next deeper level of shows use cases of the project, such as Authenticate, Register, and Modify (displayed in green). Each use case can be related to actors (displayed in blue), rules (displayed in light green) and structures (displayed in dark blue) and other use cases; these is showed in the last deeper level of the figure.
Figure 5: Social Network Project using Sunburst view

Figure 6 presents the Social Network study, focusing on the properties of the Register use case, which include description, creation date, project source, number of times that it was reused, more used tags, other related use cases and the last actualization date. Some information about how to use the tool and the colour labels of the artefacts are displayed on the top left side of the Sunburst screen, as shown in Figure 7.

4. Approach for Building for Reuse

The approach for “building fragments for being reused”, that is, for being consumed by the developers through a catalogue, was presented in the bottom flow of Figure 1. The main proposed steps, for preparing a fragment to be registered in the catalogue are:

1. Specify the fragment features;
2. Specify the fragment functionalities, and
3. Use existing fragments and adapt them, if it is needed.

Next each of this is detailed and illustrated.

4.1 Specify Fragment Features

The concept of features is used to express the variability of a fragment that will be further available for reuse.

Figure 8, shows a feature diagram for the Social Network system. The diagram includes mandatory features, such as: members of social network (Members), registered history for a member (History), allowed operations such as invitation or communication (Operations). Some optional features are: relationships between members (Relationship), member information (Profile), and so on.

Figure 7: General Information about Sunburst View.

Figure 8: Features Diagram for a Social Network.
4.2 Specify Functionalities using the Fragment Template

The specification of a use case fragment for reuse is illustrated in Table I. It presents the use case—Registering an Object—an fragment that can be reused in specific use cases of the Social Network project, such as Register a Member.

| Table I
<table>
<thead>
<tr>
<th>Use Case Fragment For Registering an Object</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fragment Name:</strong> Register &lt;object&gt;</td>
</tr>
<tr>
<td><strong>Goal:</strong> Register an &lt;object&gt; for a &lt;project&gt;</td>
</tr>
<tr>
<td><strong>Pre-condition:</strong> The &lt;actor&gt; should not be registered in &lt;system&gt;</td>
</tr>
<tr>
<td><strong>Post-condition:</strong> The &lt;actor&gt; ready to use &lt;system&gt;</td>
</tr>
<tr>
<td><strong>Actors:</strong></td>
</tr>
<tr>
<td><strong>Dependencies with Use Cases</strong></td>
</tr>
<tr>
<td><strong>Basic Flow:</strong></td>
</tr>
<tr>
<td>1. The &lt;actor&gt; chooses the option &lt;object&gt; Register.</td>
</tr>
<tr>
<td>2. The &lt;system&gt; displays the screen &lt;object&gt; Register.</td>
</tr>
<tr>
<td>3. &lt;actor&gt; informs the data as &lt;structure object&gt; and click the button Register.</td>
</tr>
<tr>
<td>4. &lt;system&gt; certifies the &lt;object&gt; data which are validated against the &lt;rule object&gt;.</td>
</tr>
<tr>
<td>5. &lt;actor&gt; confirm the &lt;object&gt; registration.</td>
</tr>
<tr>
<td>6. &lt;system&gt; presents the message: &lt;object&gt; registered with success.</td>
</tr>
<tr>
<td>7. The &lt;system&gt; redirects the client to the main page of &lt;project&gt;</td>
</tr>
<tr>
<td><strong>Alternative Flow:</strong></td>
</tr>
<tr>
<td>a) Invalid &lt;object&gt; data</td>
</tr>
<tr>
<td>In step 2 of the Basic Flow, the &lt;system&gt; identifies that the &lt;object&gt; data violate an &lt;object rule&gt;.</td>
</tr>
<tr>
<td>1. The system presents a message: “The data from the &lt;object&gt; are not valid”.</td>
</tr>
<tr>
<td>2. The event flow returns to step 1 of the Basic Flow.</td>
</tr>
<tr>
<td>b) Clear the form fields</td>
</tr>
<tr>
<td>In steps 2,3,4 and 5 of the Basic Flow, The &lt;actor&gt; can clear all the form fields for this, just click the button named &quot;Clear Fields&quot;.</td>
</tr>
<tr>
<td>c) Cancel the registered &lt;object&gt;</td>
</tr>
<tr>
<td>In step 4 of the Base Flow, &lt;actor&gt; decide to cancel the &lt;object&gt; register.</td>
</tr>
<tr>
<td>1. System shows the message: “Register of the &lt;object&gt; was cancelled”.</td>
</tr>
<tr>
<td>2. The event flow returns to step 1 of the Basic Flow.</td>
</tr>
<tr>
<td><strong>Details of the Data Structure:</strong></td>
</tr>
<tr>
<td>&lt;Structure object&gt;: definition of the &lt;object&gt; data structure.</td>
</tr>
<tr>
<td><strong>Rule Details:</strong></td>
</tr>
<tr>
<td>&lt;Rule object&gt;: definition of the validation rule that must be applied to validate the &lt;object&gt;</td>
</tr>
</tbody>
</table>

Observe that all variants of this use case will be included and subsequently excluded (in its reuse) if not necessary. It is also interesting to note that generic use cases may be composed of other use case fragments already cataloged [15].

4.3 Use Existing Fragments and Adapt Them

If a fragment can reuse other existing fragments, search in the catalogue the appropriate ones. Each selected fragment must be customized to suit the current situation of the project. Personalization consists in the substitution of the customization points for business terms. For example, for adapting the fragment "Register <object>" , shown in Table I., the tags <actor> and <object> are replaced, respectively, by the correspondent actor and object that are part of the system being specified.

5 Approach for Building with Reuse

The 4REuse tool considers that each new project to be developed is a container for all related artifacts. This includes use cases, actors, rules and structure created during the development of the correspondent system, see Figure 1.

When the user creates a new project, he/she is asked to provide the project name, the project description, and also the collaborators’ e-mail (see Figure 9). The user is also requested to configure the structure of the use case template to be used in the project (this generally depends upon companies’ standards). After, the collaborators are notified, by email, that they were included as members of the team.

![Figure 9. Creating a new Project.](image)

The template of the Use Case specification includes, Figure 10, the basic sections of conventional use case, and also some specific ones: basic structure and business rules; also, it can include tags anywhere in the template, representing the configurable parts, such as actors and objects.
Figure 10. Use Case Specification Template.

To help dealing with tags, the 4REuse makes available, at the top right side of the screen, the list of the base tags which be incrementally updated as new tags appear, see "Tags for Reuse" in Figure 10. When a user wants to use an available tag, he/she can drag it into the main or alternative flow. The symbol allows the definition of new steps for main and alternative flows; the arrows allow moving steps position in the flow (up and down). The symbol allows the exclusion of the correspondent step.

5.1 Specifying Use Cases

The use case fragment’ specification in Table I, is presented in Figure 11 using 4REuse. It can be observed that all tags used in the document were mapped and updated in the sidebar called “Tags for Reuse.” These tags allow the user to reuse its text, once they are declared; that is, is no longer necessary to rewrite them, simply dragging the tag for the text area in which the user wishes to reuse it. Declared tags can be mapped (classified) as: Data Structure, Business Rules, Use Cases and Actors. Once declared, if they appear several times, the user can simply reuse them in any use case from the same project.

Note that in the Social Network project, if a tag <rule object> is declared within some use case, the tool will fill it in the area of Business Rules tags, using the description reported in the last statement.

Another important feature is the tool’s search engine: when the user is specifying a use case step, it is possible to make searches base on it as a search string. For example, when the user in step 2 of the main flow types the word <system>, the engine makes the query in its base, and presents five suggestions of basic flow steps that start with that same word. This helps providing agility, flexibility and an easy way for user’s specification, since it seeks reusable terms, already specified in the same or in other use cases.

5.2 Catalogue Visualization

The visualization of a project through 4REuse, integrated with Sunburst, provides an overview of the project in a compact form; the colors allow a quick perception of the elements (artifacts) that compose it. With the possibility of compression and expansion of a node, the user has the vision to highlight part of the project and the information presented on the right side according to the type of node. This helps the analyzes of possible reuse of a use case or part of it, for example.

In the initially view, on the right side, project information is displayed such as: description, creation date, use cases reused from other projects, use cases reused, most reused use cases, tags most used and date of last change. If the user passes the mouse by a particular node, its description is displayed. Also, by clicking on a node, for example one node representing a use case, on the right side of the screen, information of the use case is shown, including: its description, creation date, if it was reused from another
project, number of reuses, most used tags, dependencies with other use cases, date of last modification, and a link for the use case specification. With this link it is possible, not only to view the selected use case specification, but it is also possible to copy it to another project or to a specific area, for validating the use case in terms of the text correction, its completeness and its level of reusability.

The information displayed on the right side, related to the tags, includes: description, date of creation, in how many projects it was reused, in how many use cases it was reused, and date of last change.

5.3 Instantiation of a Use Case

With the Use Case specified through 4Reuse, one can perform a query by: Visualization, Design, Features, Project Description and Use Cases, Tags, among other attributes. If we consider that the user has selected the Use Case option, the system will display a screen with all the tags available in that document, so the user can make the change as required, see Figure 12.

5.3.1 Instantiation of a Use Case

This functionality allows that a single use case can be instantiated in different ways when it is going to be printed, simply changing the tags in the document. Once the use case is specified, the user can save it in his project (Figure 13) and use the fragment (Figure 12) as a guide for creating new use cases.

6 Tool Evaluation

The tool was used and evaluated by undergraduate and graduate students. Currently, the tool has a total of 144 use cases specification catalogued, covering different applications domains, such as social network services and games. Some of these specifications have been written for reuse and others with reuse. Two studies, detailed next, were carried out; the goal was to evaluate 4REuse tool and to demonstrate the usefulness of the approach. In both studies, there were previously: a specific training with the students on requirements specification with use cases, and also a training with 4REuse tool.

The first study was held between 14th to May 24th, 2012, with a group of 21 undergraduate students. They were divided into four groups, and each group developed one project based on one of the topics: game, social network, or reservation. The developed projects were: a RPG game, a social networking for music sharing (called the FirstFM), a social network service that groups other social networks, and a hotel which includes a reservation system. In this project almost all the use case specification were written from scratch. The students were only capable of reusing fragments inside their own project.

The second study was held from May 28th to July 5th, 2012 and was conducted with 6 graduate students which have previous experience in specifying requirements. Their objective was to develop fragments for reuse, through the 4REuse tool, considering generic/common requirements of a Social Network. This group of students built together the features’ diagram for a social network, presented in Figure 8; it consists of basic elements that characterize a social network (most representing reusable functionality which could be further used in similar projects). The other built artifacts, for example the use case fragments built by them and registered in 4REuse’s catalogue, can be seen in Figure 5 through a radially view.

Using the project and features available through the 4REuse, the graduate students search for projects related to social networks; they found the project FirstFM created by undergraduate students (described as a social network of music sharing). As the existing projects in 4REuse can be viewed in a radially form with Sunburst, using this option the graduate students conducted a preliminary visual analysis on the FirstFM project, see Figure 14. The information for each node of the graph is shown on the right side of Sunburst radial graph.

The graduate students evaluated the FirstFM project in terms of style, completeness and reusability. Then they selected some use case fragments to specify again in order to improve their reuse.
Using the radial visualization in 4REuse, the graduate students identified that: from the 16 use cases present in FirstFM, 10 use cases could be reused from the Social Network project, and also that the reuse have a great potential to promote writing better specifications (more standardized and complete) and also to include more representative set of features.

Observe that using the Sunburst it is possible to compare the two projects: FirstFM, presented in Figure 14, and Social Network presented in Figure 5. In the FirstFM project, undergraduate students tried to solve the problem by themselves, so the result was a greater number of use cases with little use of tags; in the Social Network, there are a less number of use cases, but with a larger number of tags, thereby increasing the customization points of the use cases; so these use cases could be reused by projects in the field of social networks, or even in projects with different scopes.

**Applied questioner:** In both studies the students answer a questionnaire to evaluate the usability of the 4Reuse tool. The questionnaire was composed of 26 closed questions and 2 open questions that allowed the student to express in a more complete and free way their opinion. Some of the results are presented in Figure 15 and 18. The questions were grouped into five types of questions, ones that evaluate how the interface looks and how they feel using it, others that evaluate functionalities, ease of use, productivity, and learnability. More details about the questionnaire are available in [17].

**Results:** The found values are based on questionnaires answered. In terms of reuse 52% of students were able to specify use cases based on existing ones, see figure 15; however, some students have reported the difficulty to select the proper use case fragments and say that their instantiation was not so easy. Also, many students choose to write a new use case from scratch, not making reuse. In terms of productivity, see Figure 16, 77.78% of the students agreed that there was an increase in productivity, as the tool allows to specify use cases for reuse, and also gives support for the use of tags that can be reused in any part of the project.

![Figure 14. FirstFM Project using Sunburst view](image)

![Figure 15. Could you reuse the existing uses cases in the tool?](image)

7 Conclusion and Future Works

This paper considers the problems faced in specifying use cases with non-systematic approaches. These problems include ambiguity, redundancy, inconsistencies, conflicts with the domain terminology, and the inclusion of implementation details (written in natural language). All these points make the use case specifications difficult to be maintained and understood, contributing to the production of low quality requirements for information systems.

Motivated by the literature and a survey made in local companies, an approach based on concepts from [11,12,13] was proposed to reduce the involved efforts. The focus was on writing use case specification, based on previous pre-defined fragments, aiming at exploring reusable specifications. A web tool, the 4REuse, was developed to support the approach. The tool provides an interface to specify use cases, based on a defined catalogue of use case fragments; it also helps user, through tags, to build use case specification with less effort and with more reuse. As a result of the conducted studies to evaluate the tool, we conclude that 4REuse:

- Assists users in understanding requirements. However, the fragments do not replace or guarantee the user required intellectual ability to draw up the requirements of a system for reuse and with reuse;
- Gives support for improving the quality for use case specifications;
- Assists in increasing of productivity;
- Allows visualization in radial form, which improves the view of the whole project, like: the links between use case with its actors, use cases, business rules and data structures, providing information on each link;
- Can help specifying and reusing use cases (this was observed in the results obtained in the tool evaluation).

As future work, further improvements can be stated:
Evolve the tool so to a collaborative network (such as social networks) for requirements engineering students and allowing companies also to share information;

Set up a controlled experiment to compare the time spent specifying use cases using 4REuse and other tools.

Invite TI companies to use/evaluate the tool;

Import features and use cases from specific tools;

Give support to data exportation into xml / pdf / word;

Evaluate with agile methodologies;

Integrate with the TarGet tool [7], allowing the generation of tests from use cases specified for reuse;

References
[14] 4REuse - For Reuse of Engineering Requirements http://www.4reuse.info