SOFL - A Better Approach for Formal Specification: A Case Study Based Comparison

Muhammad Babar Yaqoob Khan, Sadaf Anwar, Nasir Mehmood Minhas

University Institute of Information Technology
PMAS Arid Agriculture University Rawalpindi, PAKISTAN

Abstract— Formal Methods are necessary for the specification, development and verification of safety critical systems. Formal Languages help us to identify errors at an early stage in the development process and can aid in reducing overall system development costs. Despite their benefits, formal methods are not widely accepted in the industry due to the need for high abstraction and a mathematical labor pool. The SOFL (Structured Object-Oriented Formal Language) methodology can be effective and efficient for safety critical systems. Some major issues with formal languages like higher abstraction levels, the need of mathematical skills, developer maturity and high costs can be overcome by using SOFL. SOFL specifically encapsulates three major layers: 1) structured methodology in the early stage of development, 2) object oriented methodology at the detailed level and 3) incorporation of formal methods. In this paper, we have tested the SOFL 3-step approach for the formal specification of a Railway Signaling System, already implemented by using Zed. Afterward we performed a comparison, using certain parameters identified from literature, to assess the appropriateness of Zed and SOFL methodology for the formal specification of a safety critical system. We found SOFL as an instinctive and perceptive for formal specification which overcomes the limitations of other formal languages.

Keywords— SOFL; Formal Specification; Railway Signaling System; Formal languages.

I. INTRODUCTION

The Structured Object-Oriented Formal Language (SOFL) is a sort of Formal Language which basically started at the University of Manchester, UK in 1989. SOFL was completed at the Hiroshima City University and it was settled down at Hosei University in 2000. SOFL is new for the Safety Critical systems. It has been used for the information systems, mostly [1][2]. SOFL has been finalized with the assortment of Vienna Development Method (VDM), Petri Nets and Data Flow Diagrams (DFD) [3][4].

The challenges of formal languages are: specifications of large-scale and complex software systems, it can be difficult to understand and to write. The second major issue is communication between client and developer because formal methods used the mathematical notations due to this, modifications become time-consuming and costly. The last issue is the tool support which does not necessarily reduce the difficulty of formal methods usage. The target of SOFL is to handle all above issues.

SOFL is based on three major steps. The first step is Informal Specification. Informal specifications are identified after taking requirements from the users in an informal way. Then do functional analysis. This is a compatible and informal pace. The second step is Semi formal Specification which is formed from the informal system specification.

The third and last step of this three step approach is Formal Specification but before that CDFD (condition data flow diagram) need to be developed and on the basis of formal design, formal specification is gathered. Later on verification, validation, implementation, review and testing has been performed [5].
SOFL is similar to the iterative model, but even SOFL helps to decide how long iteration will go [6]. In Formal methods some known formal languages like Z, VDM, OCL are accepted widely as the most suitable way which is available for developing secure and stable software systems. Formal methods are used for highly integrated software systems mostly, where safety and security is an important factor that needs to be deliberate and concentrate effectively.

For safety critical systems Authors proposed five keys which help the practitioners or developers to capture the safety properties using SOFL. The keys will guide the developer to identify appropriate safety properties and those identified properties will ensure that the system functions are free from failure [5].

Few authors have tried SOFL three step approach on safety critical systems and they received promising results. The failure of safety-critical systems can be a big reason that could result in loss of life, property damage or damage to the environment. There are many safety critical applications and systems around us, they are related to different fields like medicine nuclear engineering, transport, automotive and space flight [7]. So The system will be safety critical if it is concerned with the life of a human being and Property.

In our proposed approach, we’ll develop the formal specification of RSS (Railway signaling system) by using SOFL methodology and compare proposed work with specification implemented in Zed on following parameters:

- Abstraction Level
- Mathematical Skills
- Expensive
- Ready to Use in Industry
- Provides Client Interaction and Understanding
- Large Scale Development Support
- Tool support

Our work is based on five sections. The first section has basic information about the railway signalling system and its major features and portions which we have covered. The second section is based on all informal specifications of the railway signalling system. Third Section will be consisted on Semi Formal Specification of (RSS). A detailed CDfd will be shown in the fourth section. In Final step five, formal specifications will be presented. Verification and validation is not part of our research.

II. LITERATURE REVIEW

Software requirement specification ultimate base the whole software product because the next activities of software development rooted out from here. Formal methods concisely as well as in precise way to specify system specifications, but incorporate high abstraction, mathematical skills and cost overhead to specification. Another initiative, but formal approach that is structured object oriented formal language specifies requirements by overcoming the traditional formal methods’ limitations in comprehensible way in 3-steps. From the literature, Shaoying Liu et al presents SOFL language and methodology to develop a system. Authors suggested graphical notations to model the system rather than a mathematical one which have precise semantics and are more readable and can be entered via standard keyboard [11]. The chart below is a survey report on SOFL.

![SOFL Survey 2004](chart.png)

Table Number 01
Inspiration taken from [20]

Another author presented the application, which was developed by using SOFL and developed in the university information system. The intention was to investigate whether and how SOFL is beneficial in system development. Concluded as if appropriately applied, it saves the time, cost and facilitates with ample client developer communication and transforms requirements to design and to programs [2].

Whereas [12] demonstrate the case study of SOFL with application on a service oriented software development and precisely presented a 3-step formal approach to the construction of formal specification. The focus was on issues that how to apply 3-step approach for formal specification of service oriented system modeling. The author experienced SOFL as an effective and efficient, formal language for service-oriented type of softwares, with certain undertakings, simple formal notations, explicit diagrams and specification evolution.

Jichuan Wang et al. Shown that SOFL is good for embedded safety critical system development like insulin pump. In particular how the 3-step formal specification modeling approach utilized and helps to obtain a valid and precise specification for embedded system is experienced through this study [13].

Shaoying Liu presents an intuitive, expressive but formal method for specification. In this study, the author integrates the scenario based compositional and top down decompositional method that allow the user to early negotiate for judgment of requirements and reduces the probability to modification of specification versions for the larger purpose of...
consistency and completeness of a software system and incorporate SOFL’s CDFD to express higher level processes [14].

Yuting Chen et al. Presented a SOFL’s extension to specify and model a concurrent software system. This approach adds the strait and space to the SOFL for both the formal and graphical notations describing the coinciding software. For the specification of software concurrency regions are used, whereas channels used to describe the message passing communication in formal module [15]. A case study is presented by Yuting Chen in another study to specify a concurrent system [17].

Azma Binti Abdullah developed an approach based on SOFL for capturing the safety properties of a system and comprised of process and five keys that are used to capture the safety properties. The main concern of the study is how to get the safety properties for any safety critical systems[5]. In another study [16], author proposed an approach that helps out to identify, analyzed the causal effects of hazards for safety critical systems. Whereas in [17] SOFL based approach is presented to analyze the requirements for effective specification and demonstrate the suitability of SOFL.

In another study [6] of brain tumor system, authors developed the formal specification of a brain tumor treatment which is also a safety critical system using SOFL 3-step approach of formal specification. In our study, we have developed a formal specification for distance control module of railway signaling system using SOFL 3-step approach. Specification in z language is described below [9].

### DCMS

<table>
<thead>
<tr>
<th>current : Block</th>
<th>direction : Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>pma : Block → PMA</td>
<td></td>
</tr>
<tr>
<td>speed : Block → Speed</td>
<td></td>
</tr>
<tr>
<td>obstacle : P Block</td>
<td></td>
</tr>
<tr>
<td>redBlock : P Block</td>
<td></td>
</tr>
<tr>
<td>yellowBlock : P Block</td>
<td></td>
</tr>
<tr>
<td>greenBlock : P Block</td>
<td></td>
</tr>
<tr>
<td>connection : ConnectionStatus</td>
<td></td>
</tr>
</tbody>
</table>

(\text{redBlock, yellowBlock, greenBlock}) partition Block

\text{Behind Of (redBlock, direction) ∩ greenback} = 0

∀ b : \text{redBlock • pma (b) = Red}

∀ b : \text{yellowBlock • pma (b) = Yellow}

∀ b : \text{greenBlock • pma (b) = Green}

Inspiration taken from [9]

Which is about the distance control module (DCM). In this schema some major information which is required to control the train is mentioned clearly like connection status of the train and all signal information in the form of yellow green and red.

## III. PROPOSED WORK

The proposed work has basically five sections. Section A is about the basic functions of the RSS. Section B based on informal specifications. Section C consist of semi-formal specifications and CDFD has been defined in the section D of RSS system.

### Section A: System Working

RSS (Railway Signaling system information) is a safety critical system and with the passage of time when its requirements need to be defined, it becomes more and more complex. To deal with this complexity, formal methods are required.

![RSS figure: 02](image)

SOFL not only provide readability and traceability but it also provides an effective and efficient way for reducing complexity of systems [8]. To check the feasibility and reliability of the proposed methods, we carried out the case study to make requirements specified in the formal specification form for the railway signaling system (RSS). As RSS is a complex and large software application so the target portion of our work is CRD (Control Route Distance) system. The core function of RSS system is divided into two major parts one of them is interlocking function such as route control (ATS direct commands to route control part) and the other one is train protection function such as train distance control.

To verify the feasibility of our proposed specification, we have taken a case study of railway signaling system (RSS). The main core system is controlled route distance (CRD). The target system is divided into two core functions which are:

- Interlocking function (Route control)
- Train protection function (Distance control module)

The interlocking or route control function received commands from an automatic train stop (ATS) for route setup and point
machines. Basically interlocking function is responsible for safe interlocking regarding to route setup.

The distance control module is responsible for train movement corresponding to block information and the information about particular block is a part of permissive movement authority (PMA) message. The message carried by PMA is created at DCM and sent to automatic train protection (ATP).

In our proposed formal specification, we limit our specification to distance control module which has five major functions: Control route distance (control the distance of a route), Validation of train location (confirm the particular location of a train as train location), Command of the train movement/direction (control train direction according to block information), temporary speed limit command and block opening and block closing.

The system responds to on-site control directives like route setup which is received from ATS and controls the signal equipment according to block information and direct it to ATP for train movement.

**Section B: Informal Specification**

Informal specification Requirements are as follows on the basis of RSS. The Basic functions which are high level operations needed for the system those are as follows.

1. Control Route Distance
2. Train Location (communication occurs in this portion)
3. Train Movement/Direction
4. Temporary Speed Limit
5. Blocking
   5.1 Opening
   5.2 Closing

Control Route distance is a function which contains the following:

Data resources: data items needed to fulfil the functions

1. Route (1, 2)
2. Location (2, 5)
3. Direction (3)
4. Speed (1, 2, 4, 5)
5. Block (1, 5)

There are three main constraints on RSS functions and data resources. Which are described below:

1. Predefined routes (1)
2. 1 block for 1 train at a time
3. Reduce speed from 1km near destination (4)

**Role of ATP, ATS and PMA:**

- PMA sends message to ATP for train movement.
- DCM receives a message from ATP and confirm the train location receiving info from ATP

**Section C: Semi-formal Specification**

Semiformal specification of RSS is following:

```plaintext
module Distance_Control_Module;

type

Train_Info = composed of
  train_id: int
  initial_point: string
  train_destination: string
  train_route: string
  route_distance: float
end;

Train_Information = given;
Train_Route = set of routes;
Train = set of Trains;

process Train_Location (train_location: TL)
communication: boolean
ext rd Train_Information: Train
rd Route: RD
wr train_location: TL

pre
  if communication_status (from ATP) == True
  then
    communication connection is on board
  else

post Train Located
end process;

process Train_Direction (Train_Direction: TD)
communication: boolean
ext rd Train_Information: Train
rd Route: RD
wr train_location: TL

pre
  if communication_status (from ATP) == True
  then
    communication connection is on board
  else

post Train Located
end process;

module Blocking;
Current_Train = block (block0, block1, block2, block3)
Current_Direction = Direction
```

All Rights Reserved © 2014 IJARCSEE 159
Section D: CDFD

The last section is about CDFD (conditional data flow diagram). There are two CDFDs of RSS. Which elaborates the abstract overview of the systems working. One CDFD is about the distance control module as shown in the figure 03 and the second one is for Block opening and closing as shown in figure 04.

According to the authors the data flow diagram is promoted by natural tendency and notation for formal specification. It describes what to do. After formalizing DFD, it’s become condition data flow diagram which describes the behavior of the system at the architectural level [10].
specification. In this section an analysis is presented with regards to Zed methodology and SOFL methodology following the certain parameters.

There are a few parameters in our research like Mathematical Skills, Expensive, High Abstraction, Ready to use in established software industry, Client interaction and understanding, Large scale development and Tool Support. First is Mathematical Skills, according to [11] pure mathematical notations or skills for large and complex systems, don’t scale well and problematic for engineers to read and understand in formal methods. The comparison is shown in table 02.

<table>
<thead>
<tr>
<th>S. No</th>
<th>Parameters</th>
<th>SOFL</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Abstraction level</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>02</td>
<td>Mathematical Skills</td>
<td>Not Required</td>
<td>Required</td>
</tr>
<tr>
<td>03</td>
<td>Expensive</td>
<td>Less</td>
<td>High</td>
</tr>
<tr>
<td>04</td>
<td>Use in Established Industry</td>
<td>Ready</td>
<td>Not Ready</td>
</tr>
<tr>
<td>05</td>
<td>Provides client Interaction and Understanding</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>06</td>
<td>Large Scale Development support</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>07</td>
<td>Tool Support</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Comparison b/w SOFL and Z Formal Language Table: 02

On the other end SOFL approach uses semantically enrich graphical and textual notations as formal notations to model the high level architecture of the system [11]. If we look at the expansiveness of other formal methods, they persuade highly expensive in the early stage of software development. This is due to the training overhead of staff, efficient tool support and consultancy [18][19] but the SOFL specification can be developed by software engineer apart from mathematical labor skill. It provides formal specification with low cost and without training and Expert’s consultancy [6]. High abstraction is another parameter, Formal methods require high abstraction [11] and for requirements and design SOFL provides an intelligible as well as understandable language and method support for developing software systems also SOFL offers simple formal notation for developing software specifications [6].

When it comes to the use of formal specification methods and SOFL in the industry, then presently formal methods do not offer operational and proficient methods that will suit to establish industrial software processes. Formal methods concerns with high precision and mathematical notations that leads to necessary training to get familiar with it [4][11][19] and apart from other formal methods SOFL have the capability to survive itself in establishing industrial software processes because it facilitates the specification in a figured way to transform in design as an object based style[6][11].

Interaction between clients and developers is not supported by formal methods. Formal requirements are documented via formal notations which are not understandable by client/customer. Whereas SOFL interpret and define requirements in natural language which benefits in client satisfaction in understanding requirements that ultimately save the time and cost [5][6][11].

Large scale development Software development with formal methods suffer with high abstraction and mathematical skills that require staff training which leads to cost overhead and schedule slipping [11][18]. SOFL support the large scale software development within the cost and schedule. The engineer does not consider it cultural shock because of its natural tendency and easy transformation from one to another phase. SOFL provides a robust support to software developers to develop a large scale system and acts as mediator between formal methods and industrial applications [10].

V. CONCLUSION

Formal methods have some limitations like high abstraction, mathematical notations and maturity of development, all these limitations lead to cost overhead by means of increasing development duration, staff training and expert consultancy for the first time. Despite from other formal methods structured object oriented formal language overcome the limitations enforcing by other formal methods. It incorporates the 3-step approach to formal specification in which requirements are collected in natural language in informal specification phase, semi-formal specification phase changed the requirements into appropriate expressions provided by SOFL language and finally hierarchical system architecture is defined by using conditional data flow diagrams in formal specification phase that lead to implementation. During the study, we have implemented a case study of safety critical system using SOFL 3-step approach.

We have chosen a case study of railway signaling system that is already implemented in Zed, transformed into formal specification using SOFL 3-step approach and evaluate the both case studies using certain parameters. The case study provides us an insight to the familiarity of how to formalize the requirements using SOFL 3-step approach.

We experienced SOFL as a novel approach for formal specification of safety critical system as it provides easy to use method, forthcoming and understandable language for formal specification. It supports comprehensible communication between developer and client and reduces the limitations of Zed method as well as development cultural shock which observed in Zed.
FUTURE WORK

Our study has given some clear points regarding the success of SOFL approach towards the application. The future work concerns with the application of SOFL 3-step approach for more diversified large scale specification development and to test on multiple case studies. So, it can be taken as standard formal approach for formal specification. Also during the study, we have noticed the immense need of a tool for SOFL especially which facilitate 3-step automatic formal specification. So, the open source tool which is freely available is definitely necessary to ensure the correctness of specification and expected results.

REFERENCES