International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:8, 2015

Formal Specification of Web Services Applications for Digital Reference Services of Library Information System

Zainab M. Musa, Nordin M. A. Rahman, Julaily A. Jusoh

***Abstract*—**Digital reference service is when a traditional library reference service is provided electronically. In most cases users do not get full satisfaction from using digital reference service due to variety of reasons. This paper discusses the formal specification of web services applications for digital reference services (WSDRS). WSDRS is an informal model that claims to reduce the problems of digital reference services in libraries. It uses web services technology to provide efficient digital way of satisfying users' need in the reference section of libraries. Informal model is in natural language which is inconsistent and ambiguous that may cause difficulties to the developers of the system. In order to solve this problem we decided to convert the informal specifications into formal specifications. This is supposed to reduce the overall development time and cost. We use Z language to develop the formal model and verify it with Z/EVES theorem prover tool.

***Keywords*—**Formal, specifications, web services, digital reference services.

I. Introduction

SOFTWARE engineering is the process of designing, developing and delivering software systems that meet a client's requirements in an efficient and cost effective manner. To achieve this end, software engineers focus not only on the writing of a program, but on every aspect of the process, from clear initial communication with the client, through effective project management, to economical post-delivery maintenance practices [1]. The software process includes all the activities involved in software development. The high-level activities of software specification, development, validation and evolution are part of all software process.

Software specification is the process of establishing what services are required to do and the constraints on the system's operation and development [7]. The software requirements specification (SRS) is the official statement of what the system developers should implement. It should include both the user requirements for a system and a detailed specification of the system requirements. Because it is expressed in a natural language, SRS may be inconsistent and full of ambiguous statements that give different meanings to individual developers and deficiencies in software requirements are the leading cause of failure in software projects. The solution to this problem is through the conversion of these natural language specifications into formal specifications [7], [3].

Zainab Musa Magaji is with the Universiti Sultan Zainal Abidin, Malaysia (e-mail: zeemusa5@gmail.com).

Formal specifications are expressed in a mathematical notation whose vocabulary, syntax and semantics are formally defined [1]. It is the expression of the collection of properties that a system must satisfy, using a formal language and in some level of abstraction. It provide a means of proving that a specification is realizable, proving that a system has been implemented correctly with respect to its specification, and proving properties of a system without necessarily running it to determine its behaviour [2].

Formal specification describes what a system must do without saying how it is to be done. This is what makes it useful in the process of developing a computer system, because it allows questions about what the system does to be answered confidently, without the need to disentangle or sort out the information from the detailed program code. It can serve as a single, reliable reference point for the persons who investigate the user's needs, the ones who implement programs to satisfy those needs, the persons who test the results, and those who write instruction manuals for the system [4], [5]. One of the main benefits of formal specification is its ability to uncover problems and ambiguities in the system requirements [7]. Because it is independent of the program code, a formal specification of a system can be completed early in its development. It can be a valuable means of promoting a common understanding among all those concerned with the system despite the fact that it might be changed at design phase or when the needs of the user evolved [3], [8].

Verification of formal specification is essential to prove correctness and to support transformation into correct code. It is usually done manually, but nowadays, it can be done with formal method support tools such as Z/EVES [7].

In [10], we proposed a model of web services applications for digital reference services (WSDRS). We also provided the informal specifications of the model. In this paper we show the conversion of the informal specifications of the said model into formal Z specifications. Our goal is to provide a reliable model that when adapted by all libraries on earth, would lead to a networked synchronous digital reference service across the world. We also show how we verify the Z specifications using Z/VES theorem prover.

II. WSDRS Model

A web service is a software application that can be accessed remotely using XML-based languages. It allows communication between two programs irrespective of operating system, database, location, hardware or

programming language compatibility. The only thing is that each application must follow a set of standardized protocols for sharing and accessing data [5].

Digital reference service is a network that placed expertise or skill, human intermediation and resources at the disposal of users in an online environment, employs automated tools and allows human experts to concentrate or focus on difficult questions. It is viewed that digital reference service occurs when a question is received and responded electronically [6]. This model is about how web services applications could be used in proving and eradicating the complications that surrounds the provision, reliability and efficiency of digital reference services in the library information system. It is fully described in our previous paper titled "Digital Reference Services using Web Services: Overview, Design and Specifications" [10]. Fig. 1 below depicts these models.

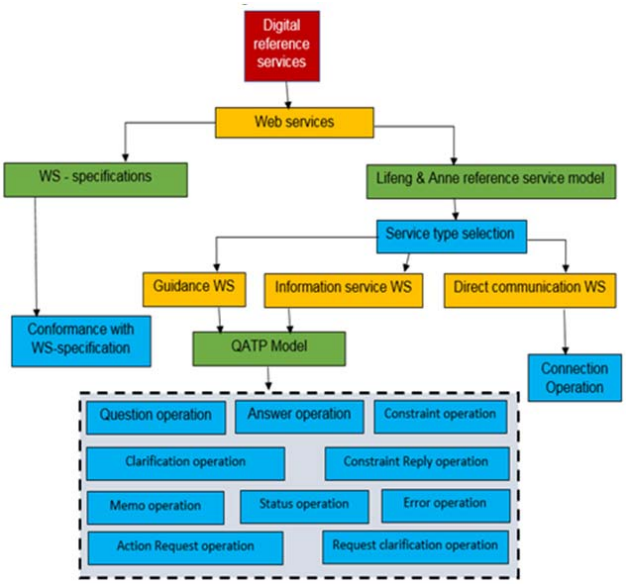

Fig. 1 The WSDRS model

The web services must conform to the WS-specifications to maintain interoperability. The web services are divided into three based on their functions. The end user (patron) is expected to select the type of service he/she wishes to get. The selection of the user will determine the web service that will be invoked. If the guidance web service or the information service web service is invoked, then the operations adapted from the QATP model will be carried out. The operations will be between end user and the web service, or web service and web services. Otherwise if direct communication service is selected by the user, then the connection operation will be carried out whereby the user will be connected to a librarian for direct communication. Direct communication will be in the form of chat, or voice call, or video call, it all depends on the user's interest.

The digital reference service as proposed in this model will provide a synchronous service to patrons. The assumption is that all possible questions that would be asked by the patrons are stored in the system database together with their answers. When a question is submitted by a user, then the web service application will search the data base for answer. If the answer exist, then it will be displayed to the user, otherwise, the web service will communicate with other web service applications of other digital reference services and send the question to them. When the answer is found, then the web service that finds the answer will send it to the web service that sends the question, which will then display it to the user and update its system database with the answer. Direct communication is expected to be used as a last option to the user when he/she is not satisfied with the answer provided by the system.

The model also handles the issue of misunderstanding of the user's question by the web service applications. Misunderstanding in this context refers to a situation whereby by the user sends a question to web services and the web service finds out that it did not have the answer to the question in its database, so it then forwards it to other web services of other digital reference services that provide similar service to the patrons; and just like the initial web service, all the other web services do not have the answer to the question in their databases. The model handles this situation by making the web services to request for a clarification message from the initial web service which will in turn request it from the user. The user will provide a clarification message by changing the original form of the question.

The issue of constraints is also fingered in this model in order to ensure full satisfaction of the user. Both the web service and the user can send a constraint message to one another and a corresponding constraint reply message. The constraint message can be attached to the question message, and it can also be sent after the question message in case the user is the sender. But when the sender is a web service, then it can be attached to the answer message.

The user is also given the ability to control the action of the web services after a transaction was initiated. This can be done by sending an action request message to the web service. Through this the user can suspend the transaction until he/she sent another request for resumption, or suspend the transaction until a specified time, or resume processing of a suspended transaction, or re-set the transaction's activity timer, or close the transaction.

III. Z Specifications for WSDRS Model

To formally specify the WSDRS system, there is the need free types and basic types. Below is the declaration of the data types used in the Z specifications:

i. [PERSON] set of all possible uniquely identified persons
ii. [WEBSERVICES] set of all possible uniquely executable web services
iii. [MESSAGE] set of all possible messages
iv. RESPONSE::= True | False | success | failure | questionInvalid
v. set of all possible responses that a web service can give

vi. MESSAGE_STATUS::= questionClear | questionNotClear | questionEmpty | accepted | notAccepted
vii. set of all possible type of status that a message can get
viii. COMM_MEDIUM::= Call | textChat
ix. Set of all the available forms of communication via the internet
x. SYSTEM_WEBSERVICES ::= infor_service_WS | guidance_WS | direct_comm_WS
xi. Set of all the available digital reference service webservices for a LIS
xii. WS_SPEC::= messaging_specs | security_specs | reliability_specs | metadata_specs | transaction_specs | management_specs | business_process_specs | xml_specs | resources_specs
xiii. Set of all the available webservices specifications
xiv. ACTION::= invoke_infor_service_ws | invoke_guidance_ws | invoke_direct_comm_ws | invoke_other_ws | connect_video_and_audio_call_services | connect_text_chat_services | suspend_process_until_another_time | resume_process | reset_activity_time | close_transaction | request_for_status_report
xv. Set of all the actions that a web service can perform

Note: Δ Symbol means there is change in state and the Ξ Symbol means there is no change in state.

The first thing to describe is the state schema for WSDRS. Like most schemas, it consists of a central dividing line in which some variables are declared above the line and the invariant properties are given below the line.

$$
\begin{aligned}
&\smile\_DRS\_\_\_\_\_\\
&\rightarrow users: \Pi\, PERSON\\
&\rightarrow questions: \Pi\, MESSAGE\\
&\rightarrow answers: \Pi\, MESSAGE\\
&\rightarrow systemWebservices: \Pi\, SYSTEM\_WEBSERVICES\\
&\rightarrow outsideWebservices: \Pi\, WEBSERVICES\\
&\rightarrow conformance1: WEBSERVICES \varphi\, WS\_SPEC\\
&\rightarrow conformance2: SYSTEM\_WEBSERVICES \varphi\, WS\_SPEC\\
&\rightarrow answerbase: MESSAGE \clubsuit MESSAGE\\
&\rightarrow selection: PERSON \clubsuit SYSTEM\_WEBSERVICES\\
&\rightarrow call\_out: SYSTEM\_WEBSERVICES \clubsuit WEBSERVICES\\
&\rightarrow send1: PERSON \clubsuit MESSAGE\\
&\rightarrow receiveby1: MESSAGE \clubsuit SYSTEM\_WEBSERVICES\\
&\rightarrow send2: SYSTEM\_WEBSERVICES \clubsuit MESSAGE\\
&\rightarrow receiveby2: MESSAGE \clubsuit PERSON\\
&\rightarrow send3: SYSTEM\_WEBSERVICES \clubsuit MESSAGE\\
&\rightarrow receiveby3: MESSAGE \clubsuit WEBSERVICES\\
&\rightarrow send4: WEBSERVICES \clubsuit MESSAGE\\
&\rightarrow receiveby4: MESSAGE \clubsuit SYSTEM\_WEBSERVICES\\
&\frown\_\_\_\_\_\\
&\rightarrow \text{dom } answerbase = questions\\
&\rightarrow \text{ran } answerbase = answers\\
&\rightarrow \text{dom } conformance1 = outsideWebservices\\
&\rightarrow \text{dom } conformance2 = systemWebservices\\
&\rightarrow \text{dom } selection = users\\
&\angle\_\_\_\_\_
\end{aligned}
$$

Fig. 2 State schema for WSDRS

$$
\begin{aligned}
&\smile\_Select\_info\_\_\_\_\_\\
&\rightarrow \Xi DRS\\
&\rightarrow patron? : PERSON\\
&\rightarrow active\_system\_ws! : SYSTEM\_WEBSERVICES\\
&\rightarrow action! : ACTION\\
&\frown\_\_\_\_\_\\
&\rightarrow patron? \in users\\
&\rightarrow selection\, patron? = infor\_service\_ws\\
&\rightarrow action! = invoke\_infor\_service\_ws\\
&\rightarrow active\_system\_ws! = infor\_service\_ws\\
&|\_\_\_\_\_
\end{aligned}
$$

Fig. 3 Schema for selecting information service

Next we start specifying the operations in WSDRS model. An operation schema represents some operation that the system can perform. Fig. 3 illustrates selection operation where the user needs help on information service of the library.

The selection operation when the user chooses guidance as the area where he/she needs help is depicted in Fig. 4:

$$
\begin{aligned}
&\smile\_Select\_guid\_\_\_\_\_\\
&\rightarrow \Xi DRS\\
&\rightarrow patron? : PERSON\\
&\rightarrow active\_system\_ws! : SYSTEM\_WEBSERVICES\\
&\rightarrow action! : ACTION\\
&\frown\_\_\_\_\_\\
&\rightarrow patron? \in users\\
&\rightarrow selection\, patron? = guidance\_ws\\
&\rightarrow action! = invoke\_guidance\_ws\\
&\rightarrow active\_system\_ws! = guidance\_ws\\
&\angle\_\_\_\_\_
\end{aligned}
$$

Fig. 4 Schema for selecting guidance service

The selection operation when the user chooses to have a direct communication with the librarian is depicted in Fig. 5:

$$
\begin{aligned}
&\smile\_Select\_directcomm\_\_\_\_\_\\
&\rightarrow \Xi DRS\\
&\rightarrow patron? : PERSON\\
&\rightarrow active\_system\_ws! : SYSTEM\_WEBSERVICES\\
&\rightarrow action! : ACTION\\
&\frown\_\_\_\_\_\\
&\rightarrow patron? \in users\\
&\rightarrow selection\, patron? = direct\_comm\_ws\\
&\rightarrow action! = invoke\_direct\_comm\_ws\\
&\rightarrow active\_system\_ws! = direct\_comm\_ws\\
&\angle\_\_\_\_\_
\end{aligned}
$$

Fig. 5 Schema for selecting direct communication

The overall selection operation is obtained from the schema composition of the schemas for the three possible selections.

$$Selection\_Operation \mid Select\_info \varpi Select\_guid \varpi Select\_directcomm$$

Fig. 6 The Selection_Operation schema

Next is the question operation described in the schema depicted in Fig. 7:

Fig. 7 The question operation schema

Next is the answer operation, which is described in two different schemas of Figs. 8 and 9. The first one handles the situation when the system can help the user independently and thus the answer will be sent from the system's web service. While the second schema handles the situation when the system must seek the help of other systems before it can satisfy the user and thus the answer will be sent from the outside system's web service.

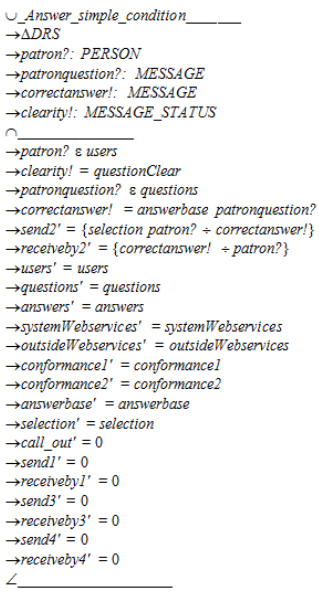

Fig. 8 Answer_simple_condition schema

The general answer operation is obtained from the composition of the schemas for the three possible cases of providing answer to the user.

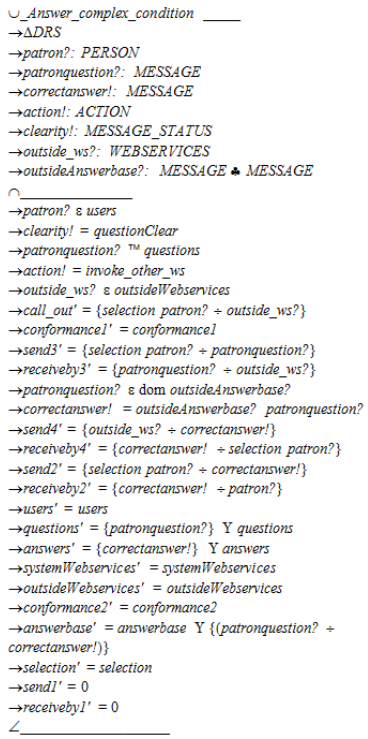

Fig. 9 Answer_complex_condition schema

Answer_operation | *Answer_simple_condition* ϖ *Answer_complex_condition*

Fig. 10 The Answer_operation_schema

The next is the request clarification operation, which is described in two different schemas of Figs. 11 and 12. One handles the situation when the system can help the user independently and thus requested the clarification itself. The second schema handles the situation when the system has seeked the help of another system and thus the request was sent from the outside system.

Fig. 11 The Request_Clarification_Simple schema

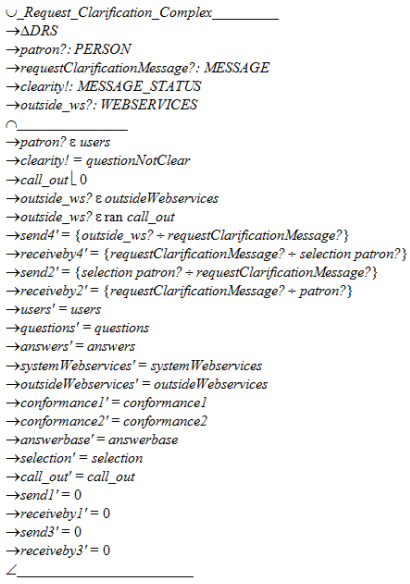

Fig. 12 The Request_Clarification_Complex schema

The general request clarification operation is obtained from the composition of the schemas for the two possible cases of sending a request clarification message.

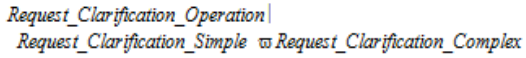

Fig. 13 Request_Clarification_Operation schema

Next operation is the clarification operation. It is described in two different schemas of Figs. 14 and 15. Each is a pair to one of the schemas for request clarification operation that corresponds to it.

The whole clarification operation is obtained from the composition of the schemas for the two possible cases of sending a clarification message.

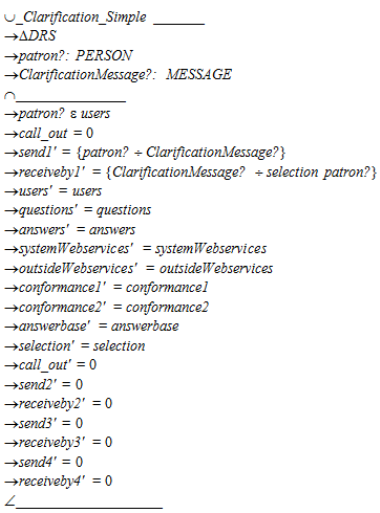

Fig. 14 The Clarification_Simple schema

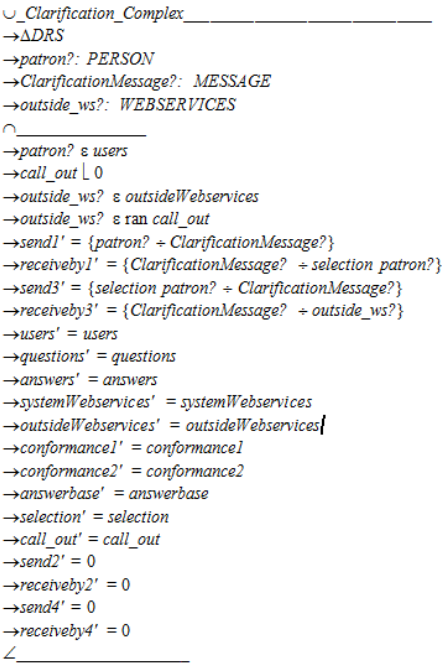

Fig. 15 The Clarification_Complex schema

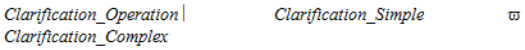

Fig. 16 Clarification_Operation schema

Next is the constraint operation. It is also described in two schemas. One handles the situation when the user sends the constraint message to the web service and the other handles the case when the constraint message was sent to the user by web service.

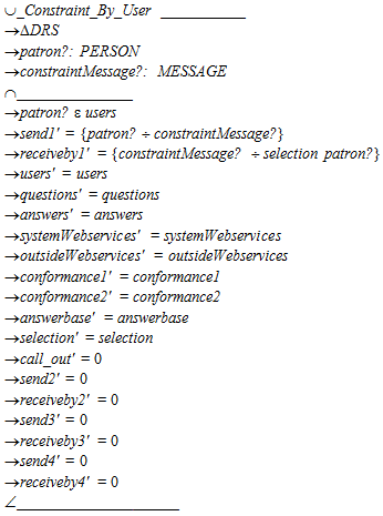

Fig. 17 The constraint by user schema

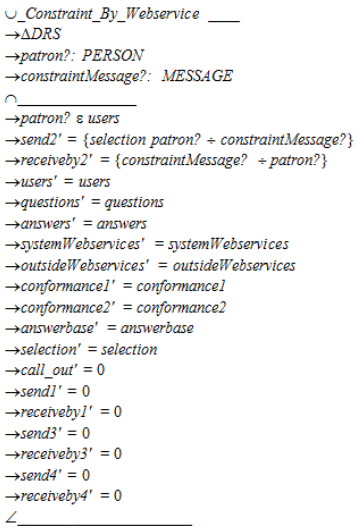

Fig. 18 The constraint by web service schema

The complete constraint operation is obtained from the composition of the schemas for the two possible cases of sending a constraint message.

$$Constraint\_Operation \mid Constraint\_By\_User \varpi Constraint\_By\_Webservice$$

Fig. 19 The Constraint_Operation schema

Next is the constraint reply operation. It is also described in two schemas for each to responds to its corresponding constraint schema. They are depicted in Figs. 20 and 21.

The complete constraint reply operation is obtained from the composition of the schemas for the two possible cases of sending a constraint reply message.

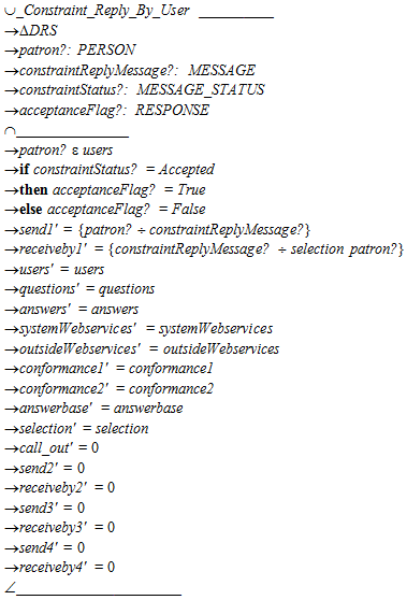

Fig. 20 The constraint reply by user schema

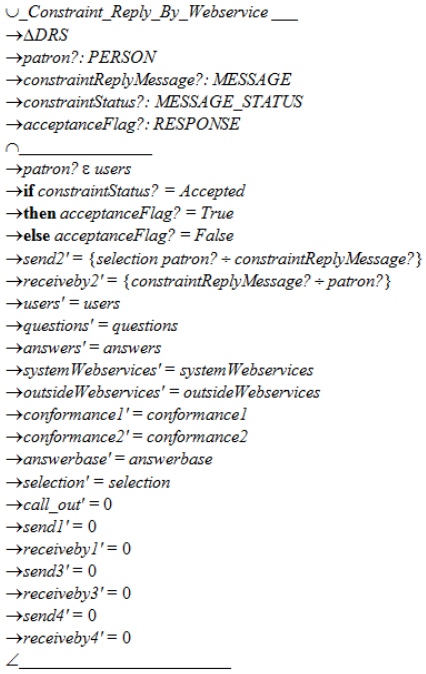

Fig. 21 The constraint reply by web service schema

$$Constraint\_Reply\_Operation \mid Constraint\_Reply\_By\_User \varpi Constraint\_Reply\_By\_Webservice$$

Fig. 22 The Constraint_Reply_Operation schema

Next is the action request operation, which is specified in the schema depicted in Fig. 23:

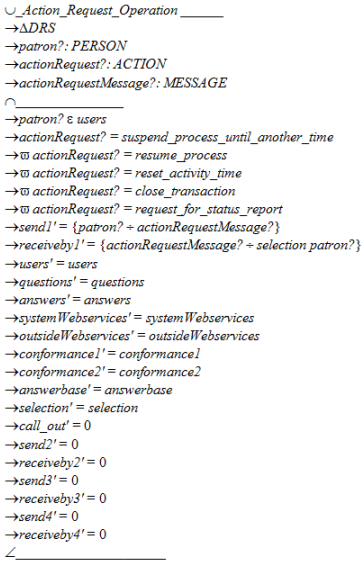

Fig. 23 The Action_Request_Operation schema

The operation that pairs with the action request is the status reply operation and it is described in Fig. 24:

$\cup$_Status_Reply_Operation _____
→ΔDRS
→patron?: PERSON
→statusReplyMessage?: MESSAGE
→actionRequestStatus?: MESSAGE_STATUS
→statusReply?: RESPONSE
∩_____
→patron? ε users
→**if** actionRequestStatus? = Accepted
→**then** statusReply? = Success
→**else** statusReply? = Failure
→send2' = {selection patron? ÷ statusReplyMessage?}
→receiveby2' = {statusReplyMessage? ÷ patron?}
→users' = users
→questions' = questions
→answers' = answers
→systemWebservices' = systemWebservices
→outsideWebservices' = outsideWebservices
→conformance1' = conformance1
→conformance2' = conformance2
→answerbase' = answerbase
→selection' = selection
→call_out' = 0
→send1' = 0
→receiveby1' = 0
→send3' = 0
→receiveby3' = 0
→send4' = 0
→receiveby4' = 0
∠_____

Fig. 24 The Status_Reply_Operation Schema

The next is the direct communication operation specified in the schema of Fig. 25:

$\cup$_Connection_Operation _____
→ΞDRS
→patron?: PERSON
→Select_Comm?: PERSON ♣ COMM_MEDIUM
→action?: ACTION
∩_____
→patron? ε users
→selection patron? = direct_comm_ws
→patron? ε dom Select_Comm?
→**if** Select_Comm? patron? = Call
→**then** action? = connect_video_and_voice_call_service
→**else** action? = connect_text_chat_service
∠_____

Fig. 25 The Connection_Operation schema

The last operation in WSDRS system is the error operation depicted in Fig. 26:

$\cup$_Error_Operation_____
→ΞDRS
→patron?: PERSON
→report!: RESPONSE
→clearity?: MESSAGE_STATUS
→userQuestion?: MESSAGE
∩_____
→patron? ε users
→userQuestion? ™ questions
→clearity? = questionEmpty
→report! = questionInvalid
∠_____

Fig. 26 The Error_Operation schema

IV. Verification of the Formal Specifications

Verification must be done on Z specifications in order to confirm their correctness. Using Z/EVES, verification is done by checking the syntax errors if any, for all the specifications. In Z/EVES both specifications and theorems are perceived as paragraphs.

The status of each paragraph is shown in two columns to the left of the paragraph. The leftmost column shows one of three symbols:

- "?" indicates that the paragraph has not been checked.
- "Y" indicates that the paragraph has been checked and has no syntax or type errors.
- "N" indicates that the paragraph has been checked and has errors.

The next column shows the proof status for the paragraph, using one of three symbols

- "?" indicates that the paragraph has not been successfully checked (so proof status cannot be determined).
- "N" indicates that the paragraph has an associated goal that is unproven.
- "Y" indicates that the paragraph has no unproved goals [9].

Looking at Fig. 27, it is clear that all the specifications and theorems developed were checked to have neither syntax nor type error because each paragraph has a 'Y' in both the two status columns.

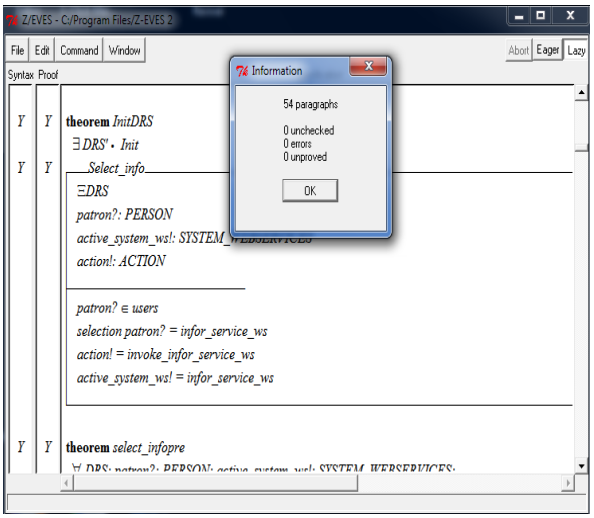

Fig. 27 The Z/EVES window showing verification result

Fig. 27 shows that there are a total of 54 paragraphs in the specification, all the paragraphs are checked, all of them have no syntax error and all of them are proved. Some of the paragraphs are types, some are schemas and some are theorems. This result clearly shows that the Z specifications of WSDRS of LIS are verified by the Z/EVES prover to be correct and thus the WSDRS formal model is accurate, unambiguous and highly reliable. It can be adopted and adapted by any library so as to provide a user satisfying digital reference service.

V. Conclusion and Future Work

In this paper, we demonstrate Z formal specification of WSDRS model. Using Z/EVES theorem prover, we fully

verified the Z specification and found them to be correct. One of our future works shall deal with the validation of the formal specifications of WDRS which we developed in this work.

REFERENCES

[1] Lee R. Y., 2013. "Introduction to Software Engineering" in *Software Engineering: A Hands-On Approach.* pp 3 – 4.

[2] Lamsweerde, A. V., 2000. "Formal Specification: A Road Map." in *ACM 2000 article.* pp 223-234. *citeseerx.ist.PSV,ed/viewdoc/summary?doi=10.1.1.35.4492/accessed on 17/05/2014.*

[3] Somerville, I., 2009. "Formal Specification" in *Software Engineering, 9th edition.* Ch27.

[4] NASA, 1995. "Formal Methods Specification and Verification Guide Book for Software and Computer Systems. Volume: Planning and Technology Insertion".

[5] Spivey J. M. (1998). "Tutorial Introduction" in *The Z Notation: A Reference Manual.* pp 1-17

[6] Nordin S. K., Kassim N. A. and Baharaddin K., 2012. "Evaluating Digital Reference Service in University Libraries". In *IEEE Symposium on Business, Engineering and Industrial Applications*. 202-206.

[7] Man M., Jusoh J.A., Rahim M.S.M. and Zakaria M.Z., 2010. "Formal Specification for Spatial Information Databases Integration Framework (SIDIF)". In *Telkomnika*. 9:81–88.

[8] Jusoh J.A., Saman M.Y.M. and Man M., 2011." Formal Validation of DNA Database Using Theorem Proving Technique". In *International Journal of the Computer, the Internet and Management.* 19:74 – 78.

[9] Saaltink M. (1999). "Proving Theorems" in *The Z/EVES 2.0 Us er's Guide* Ch. 5.

[10] Musa Z. M., Rahman N. M. A and Jusoh J. A., 2014. "Digital Reference Services: Overview, Design and Specifications". *To be published in journal of theoretical and applied information technology Vol 80 October 2015*