Validating Software Requirements

John K. Sharp
Sharp Informatics

Requirements validation must convincingly establish that the design meets actual requirements of the process being supported. This type of validation ensures that the software meets the form, fit, and function to solve the problem that is known to the subject area experts. Although validation of requirements must be considered a technical effort, the individuals with the real world knowledge (i.e. subject matter experts) are seldom skilled enough to read, much less approve, the technical models that specify the design. The challenge for requirements validation is to explain the technical specification in such a way as to make the subject matter experts capable of precisely validating the design so that they can be held accountable for the resulting application. This article will present an approach for validating the requirements contained in an information model.

Graphical information model diagrams contain a condensed form of the subject area knowledge. This presentation of knowledge in a formal syntax of knowledge benefits the communication among analysts and implementers, but it usually creates a barrier between the information technology (IT) professionals and the subject matter experts. This barrier is unique in the world of engineering because it is one of the few times that the person with the knowledge (e.g. subject matter expert) is not responsible for the resulting product. Would management allow the manufacturing of a prototype part without having the most knowledgeable person (subject matter expert who understands form, fit, and function) being accountable for the resulting part?

Current Problems in Validating Software Requirements

All too often, validation of requirements is left to the analyst, who by learning about the subject area/technology under investigation becomes the de-facto expert. This absolute reliance on the analyst also allows the implementer (who may also be the analyst) to depart from the design as the system is built and modified because the subject matter expert is only interested in the result and not the design. The resultant product may function correctly, but have an out-of-date design specification. This complicates maintenance.

Software tools support the implementation side of application development. They document the results of the analysis and significantly reduce the time required for coding. These tools improve productivity but they assume that the specified design meets the requirements. If this assumption is not valid, the software tools speed up the production of nonfunctional software.

Studies of the software life cycle have found that 45 to 60 percent of all errors in resulting applications come from errors in the design [1]. Rework during the implementation phase of the life cycle is also impacted because of discovered design errors. Errors in the resulting application lead to an unsuccessful project. Maintenance takes over and the application must be reworked or abandoned. Furthermore, a graphical design may not be complete and the implementer is forced to talk to the subject matter expert as well as the analyst and others to determine what should be delivered. Without requirements validation being included in the software life cycle, all of the participants could have done the right job and the project could still be unsuccessful. A design containing errors can be wonderfully implemented in an impressive application without being of any value to the user. All of this state-of-the-art technical work is useless because the real world process needs are not supported by the application. The conversion of the incorrect application into one that supports the process encourages short cuts and fixes that hinder future maintenance. This may also result in an incomplete and undocumented built design. Establishing the validity of the design should be a required step.

In attempts to get better requirements, I know of cases where the subject matter experts have learned to be information systems (IS) analysts because the IS analysts could not do the required job. At other times, the IS analysts have become the subject matter experts by default. The amount of wasted corporate resources is significant in all such efforts. With this history it is understandable that management has grown to distrust the IS capabilities of their IT organizations.

Requirements Validation Using Natural Language Modeling

The Natural Language Modeling (NLM) analysis procedure for model validation consists of a set of deterministic steps that generate questions about simple sentences that are based on the graphical design. The precision contained in the graphical model is maintained through the application of the NLM procedure. This procedure focuses on simple sentences that are understood by the expert. The expert's "yes" and "no" answers to the NLM questions are compared to the answers required by the graphical design. If the expert's answers and the design do not agree, then the involved rule is investigated and the correct rule is determined. The model is validated when all of the answers from the expert and the model are the same.

Every "yes" and "no" answer pattern results in one of three outcomes:

1. validating a portion of the model
2. developing new questions that lead toward validation
3. detecting an inconsistency that must be resolved

In many cases the clarity of the NLM questions allow the expert to fully understand the implication of the requirements. The analyst is able to assist in the clarification of subject area rules that may be incomplete or inconsistent. This results in better requirements and in applications that are produced with less rework during implementation and less maintenance after release. The subject matter experts become accountable for the final design. The NLM procedure also allows other subject matter experts who have not been involved in the initial validation to collaborate in the independent validation of the design. None of these subject matter experts are required to understand any aspects of the NLM modeling procedure or to understand a graphical presentation of the modeling results. The analyst understands both the graphical model and the NLM procedure, but functions only as a facilitator for extracting the information required for the validation of the model. The analyst does not personally validate any of the subject area knowledge in the model.

Validation Examples

The validation of a graphical model using the NLM procedure is possible because of the model's unique yes/no answer pattern for the NLM questions and the subject area expert's ability to independently answer the same questions. A simple example will be presented here to show how the procedure identifies an error in a particular type of graphical model called an IDEF1x model. The type of graphical model is not important because the underlying knowledge rules will be the same for a subject area no matter what graphical form is used to present the design. An object-oriented model [2] example can be validated using this same technique.

Because the NLM validation procedure is deterministic, the analyst's knowledge of the subject area is not required to produce or validate the model. This allows two approaches for validation. The first approach uses the real world objects and sentences from the subject area and the second uses variables in place of the real world constructs. Models have always been constructed by educating the analyst about the subject area and then having him/her develop the required model. The precision of the NLM procedure allows the analyst to produce models without knowledge of the subject area being addressed. Although the variable approach may only have direct use with proprietary or security sensitive applications, it can also allow an independent analyst to lead a totally unbiased validation effort. Since the independent analyst could not apply any existing real world knowledge about the subject area, the validation will come only from the involved subject matter experts. Furthermore, the results will be precise and will possibly lead to an implementation that is correct the first time.

Real World Validation Example

The example problem will be a movie marquee that presented the movies being shown in a theater. Although this example is simple because of the shared real world knowledge among all movie goers, it highlights the common problem that the analyst assumes that his/her knowledge is sufficient or needed for producing the model.

Monday Movie Presentation

Session	Theater 1	Theater 2	Theater 3
10 a.m.	*Jaws*	*Snow White*	*Invisible Man*
noon	*Jaws*	*Mad Max*	*Invisible Man*
3 p.m.	*Mad Max*	*Fantasia*	*Invisible Man*
6 p.m.	*Jaws*	*Fantasia*	*Invisible Man*

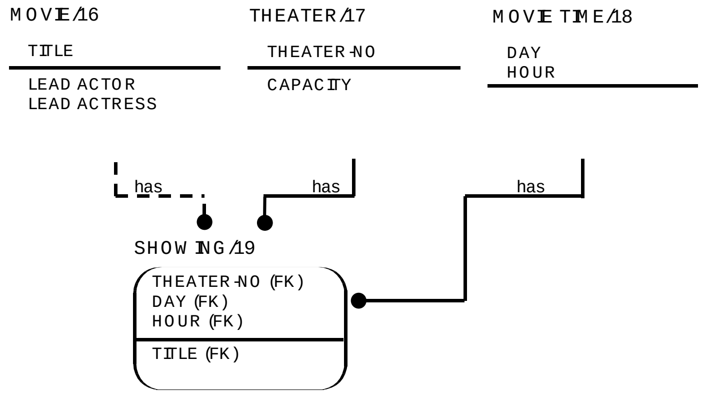
Figure 1. *Movie marquee.*

The analyst would interview the theater manager and review appropriate examples, such as the marquee in Figure 1. The model in Figure 2 reflects the results of the modeling effort. The model presented is based on the constructs specified in the IDEF1x standard [3]. The manager can understand at least some of the knowledge in the diagram when it is explained by the analyst. The manager can agree with what was said, but he/she cannot be held accountable for the contents of the graphical model. Implementation should be based on the graphical model, so errors in the resulting system due to errors in the graphical model will put the analyst and the expert in the position of blaming each other for the result. No accountability is possible in this situation.

The model in Figure 2 documents the results of analysis for the movie marquee problem. Some of the efforts in reviewing this graphical model deals with the decision to make "Movie Time" an entity as opposed to having it be part of "Showing." Although this could be of interest, the main effort should be to determine if this model contains the correct rules for the subject area. The only way to validate the rules is to involve the subject matter expert in a process that will make him/her accountable for the resulting model. The NLM procedure has the expert create sentences based on the model that contain instances from examples used in the modeling process or independently supplied by the expert. The validation is accomplished by asking questions about the sentences.

Figure 2 is a standard IDEF1x model. The child entity (rounded corners) has allowed populations (foreign keys - FK) that come from the parent entities (square corners). Using a

Figure 2. *Example of IDEF1x model.*



population instance from Figure 1 and the “Showing” entity in Figure 2 the subject matter expert could generate the following sentence:

1. In theater 1 on Monday at 1 p.m. the movie *Jaws* is showing.

The instances in sentence one from Figure 1 are shown in bold type. Each one of these instances can be replaced with an appropriate label as shown in sentence two to form a type sentence. These two sentences, one instance and one type, are now used in the NLM procedure to determine if the associated portion of the model is correctly specified.

2. In theater <Theater-No> on <Day>at <Hour> the movie <Title> is showing.

The NLM analysis procedure consists of a set of questions that the expert answers about the involved sentences. The first question deals with the ability to replace one instance value in sentence one and have both the original sentence and the modified sentence exist at the same time. The formal statement of question one in the NLM procedure is:

Q1.1. Given that fact “In theater 1 on Monday at 1 p.m. the movie *Jaws* is showing.” is true, is it allowed for another valid Theater Number [for example “2”] to exist such that the fact “In theater 2 on Monday at 1 p.m. the movie *Jaws* is showing.” is also true?

The subject matter expert would answer this question “no.” This answer is placed at the end of the first row of the matrix in Figure 3. The Q1 is asked in sequence for each individual placeholder. The matrix presentation of the answer vector for subject matter expert’s Q1 answers is shown in Figure 3.

For model validation the NLM procedure now references the graphical model to answer the same questions. A skilled analyst can obtain these answers directly from the IDEF1x model in Figure 2. The answer vector for the IDEF1x model is shown in Figure 4.

The NLM procedure specifies the next analysis step based on the yes/no answer vector. For model validation the NLM procedure now references the graphical model to answer the same questions. A skilled analyst can obtain these answers directly from the IDEF1x model in Figure 2. The answer vector for the IDEF1x model is:

2. In theater <Theater-No> on <Day>at <Hour> the movie <Title> is showing.

The model is validated when the subject matter expert’s answers are the same as the model’s answers. For this example three of the four answers are the same. The first answer is different. This difference must be investigated. A dependency exists for every “no” answer. The independent portion of the sentence that creates the dependency must be established. The NLM procedure does this through generating a set of questions that will establish all rules that apply to this sentence. In this example two rules are identified.

Figure 3. *Expert’s answers to Question 1.*

1	Monday	1 p.m.	Jaws	
				Allowed
another	Monday	1 p.m.	Jaws	no
1	another	1 p.m.	Jaws	yes
1	Monday	another	Jaws	yes
1	Monday	1 p.m.	another	no

1	Monday	1 p.m.	Jaws	
				Allowed
another	Monday	1 p.m.	Jaws	yes
1	another	1 p.m.	Jaws	yes
1	Monday	another	Jaws	yes
1	Monday	1 p.m.	another	no

Figure 4. *Graphical model’s answers to Question 1.*

The last “no” answer is dependent on the identifier for “Showing” in the model in Figure 2. The real world situation for this dependency is that an individual theater can show only one movie at a time. Showing two or more movies at one time in the same theater would be chaotic. The analyst as well as all moviegoers understands this rule. The first “no” answer is dependent on the movie and movie time. The real world situation for this dependency is that only one copy of a particular movie is available at one time. This rule is due to the flat fee that the theater must pay for each copy of a movie and the manager’s decision to rent only one copy of a movie at a time. This second rule is only uncovered if the manager provides this information directly or the analyst asks the necessary questions. In most cases subject matter experts do not know what information is of value to the analyst and critical information may not be given. In too many cases the analyst uses his/her knowledge and may not ask the expert. Model quality may also depend on the amount of time spent with the manager. This often results in an extended amount of training for the analyst in certain aspects of the process and requires a significant amount of the manager’s time. It may be more efficient for the analyst to follow a procedure that requests the analyst to ask the needed questions.

Obviously, all of the correct data could be stored in an application developed using the first model, but data that violates a business rule could also be stored in this application. The model shown in Figure 5 enforces both of the rules and does not allow bad data to appear in the application. The new rule is expressed as the alternate key (AK) in the “Showing” entity.

All of the rules within the model can be properly established using a series of questions that are asked of the subject matter expert. The analyst’s independent knowledge about the subject area is not needed. Because of this, real world instances in the sentences can be replaced with variables so that the validation procedure could still be completed without exposing confidential information to the analyst.

Variable Validation Example

For validation with variables, each instance in a sentence is

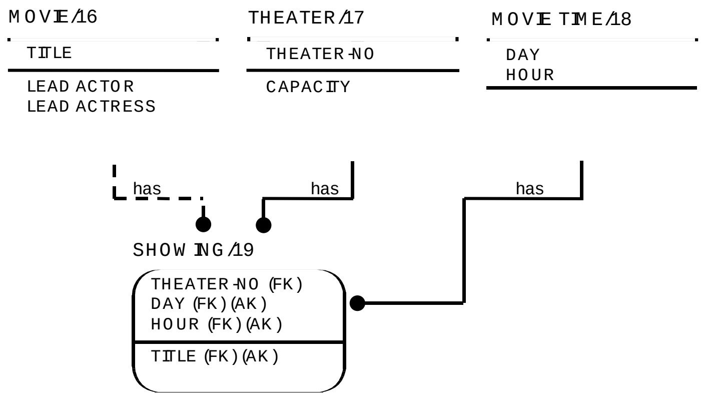


Figure 5. *Corrected IDEF1x model.*

replaced by a variable and the text segments in the sentence are replaced with constant text segments. The expert would know the meaning or semantics of the sentence and the objects. The "yes/no" answer pattern would be the same because the expert answers the same questions. An analyst who is knowledgeable of the subject area can produce the yes/no answer pattern for the graphical model. The validation would proceed by comparing the yes/no answer vectors as previously discussed. Using this approach, an analyst that was not associated with the generation of the model could independently validate the model while being prevented from knowing the technical content of the model.

3. Text1 a1 text2 b1 text3 c1 text4 d1 text5.
4. Text1 <A#> text2 <B#> text3 <C#> text4 <D#> text5.

Sentences three and four have been created from sentences one and two with:

Text1= In theater	A#=Theater-No.	a1=1
Text2= on	B#=Day	b1=Monday
Text3= at	C#=Hour	c1=1 p.m.
Text4= hr the movie	D#=Title	d1=Jaws
Text5= is showing		

The subject matter expert knows the meanings of the variables so the Q1 answer vector in Figure 6 is the same as the one in Figure 3.

The analyst who developed the graphical model would create the answer vector in Figure 4. The comparison of the vectors would proceed as before and two rules would be found. The analyst validating the model without knowledge of the subject area would know the two rules that exist, while the knowledgeable analyst and the subject matter expert could also attach the real world meaning to the rules. The analyst validating the model has no knowledge of what is being validated, but by following the NLM procedure he/she establishes whether or not the model precisely reflects the expert's knowledge of the subject area.

Figure 6. *Expert's answers to Question 1 using variables.*

a1	b1	c1	d1	
				Allowed
another	b1	c1	d1	no
a1	another	c1	d1	yes
a1	b1	another	d1	yes
a1	b1	c1	another	no

Conclusion

The result of model validation using the NLM procedure is that the subject matter expert is accountable for the contents of the model, even though the expert cannot read the developed graphical model. This type of validation is critical in the development of high reliability/high risk systems. By requiring that the analyst follow a deterministic validation procedure, the results of the validation effort depends on the expert's knowledge of the subject area and not on the analyst's knowledge of the subject area. The integrity of the resulting application is improved because the subject matter expert is accountable.

Validating graphical models using NLM is one way to minimize some unnecessary costs of IS efforts. Correcting the initial design also allows for an understanding of failures in the original modeling process. Problems can be identified and internal procedures can be modified as required. The need for staff training can also be assessed. ◆

About the Author

John K. Sharp is the founder and principal consultant for Sharp Informatics. Before starting Sharp Informatics in 1997, he was employed by Sandia National Laboratories in Albuquerque, N.M. for 18 years. While at Sandia he held staff and management positions in all areas of information technology, including analysis, design, implementation, maintenance, information architecture, data administration, and information technology research. Dr. Sharp is the developer of a mathematically precise information analysis procedure. Sharp was the editor of the international standard for conceptual schemas. He has co-chaired two international conferences on Natural Language Modeling and he has presented numerous papers and seminars at professional conferences.

Sharp Informatics
Albuquerque, N.M.
Voice: 505-243-1498
Internet: www.sharp-informatics.com

References

1. Jones, Capers, Software Quality Forum, Sandia National Laboratories, April 1-3, 1997, Albuquerque, N.M.
2. Sharp, John, "Validating an Object Oriented Model," *Journal of Conceptual Modeling*, December 1998, www.inconcept.com
3. IDEF1x Data Modeling Standard, Federal Information Processing Standards Publications, FIPS PUB 184, December 1993.