See published version:

Osman Balci (2001), “A Methodology for Certification of Modeling and Simulation Applications,” *ACM Transactions on Modeling and Computer Simulation (TOMACS) 11*, 4 (Oct.), 352-377.

A Methodology for Certification of Modeling and Simulation Applications

OSMAN BALCI
Virginia Tech

Certification of modeling and simulation (M&S) applications poses significant technical challenges for M&S program managers, engineers, and practitioners. Certification is becoming increasingly more important as M&S applications are used more and more for military training, complex system design evaluation, M&S-based acquisition, problem solving, and critical decision making. Certification is a very complex process, involves the measurement and evaluation of hundreds of qualitative and quantitative elements, mandates subject matter expert evaluation, and requires the integration of different evaluations. Planning and managing such measurements and evaluations require a unifying methodology and should not be performed in an *ad hoc* manner. This paper presents such a methodology. The methodology consists of the following body of methods, rules, and postulates: (a) employment of subject matter experts, (b) construction of a hierarchy of indicators, (c) relative criticality weighting of indicators using the analytic hierarchy process, (d) using a rule-based expert knowledge base with an object-oriented specification language, (e) assignment of crisp, fuzzy, and nominal scores for the indicators, (f) aggregation of indicator scores, (g) graphical representation of the indicator scores and weights, (h) hypertext certification report, and (i) interpretation of the results. The methodology can be used for certification of any kind of M&S application either throughout the M&S development life cycle or after the development is completed.

Categories and Subject Descriptors: I.6.4 [**Simulation and Modeling**]: Model Validation and Analysis; D.2.4 [**Software Engineering**]: Software/Program Verification—*validation*

General Terms: Measurement, Verification

Additional Key Words and Phrases: accreditation, certification, credibility assessment, evaluation, quality assessment, verification, validation

1. INTRODUCTION

A *model* is a representation or abstraction of something such as an entity, a system or an idea. *Simulation* is the act of experimenting with or exercising a model or a number of models under diverse objectives including acquisition, analysis, and training. For example, if the analysis objective is to predict the performance of a complex system design, we *experiment* with a model or a distributed set of models representing the system design. If the predicted performance is used in making an acquisition decision, the process is called *simulation-based acquisition.* If the training objective is to teach military commanders how to make decisions under a combat scenario, we *exercise* a

Author's address: Osman Balci, Department of Computer Science, 660 McBryde Hall, MC 0106, Virginia Tech, Blacksburg, VA 24061; e-mail: balci@vt.edu, web: http://manta.cs.vt.edu/balci.

model or a distributed set of models in an interactive manner by using the trainees as part of the simulation. We refer to a specific simulation created for a particular objective as a modeling and simulation (M&S) application (MSA).

The U.S. Department of Defense (DoD) is the largest sponsor and user of MSAs in the world. DoD uses many different types of MSAs such as continuous, discrete-event, distributed, hardware-in-the-loop, human-in-the-loop, Monte Carlo, parallel, and synthetic environments bringing together simulations and real-world systems. The DoD Instruction 5000.61 states that "it is the DoD policy that: … models and simulations used to support major DoD decision-making organizations and processes … shall be accredited for that use by the DoD component sponsoring the application" [DoDI 1996, p. 2].

The DoD Instruction 5000.61 defines accreditation as "the official certification that a model, simulation, or federation of models and simulations is acceptable for use for a specific purpose" [DoDI 1996, p. 9]. On the other hand, the International Organization for Standardization (ISO) defines accreditation and certification as follows:

- *Accreditation* is a "procedure by which an authoritative body gives formal recognition that a body or person is competent to carry out specific tasks." [Rae, Robert, and Hausen 1995, p. 287]
- *Certification* is a "procedure by which a third party gives written assurance that a product, process or service conforms to specified characteristics." [Rae, Robert, and Hausen 1995, p. 288]

The above ISO definitions conflict with the definitions given in [DoDI 1996]. We use the ISO terminology in this paper. It should be noted that the DoD's definition of "accreditation" corresponds to the ISO's definition of "certification."

Component-based development is becoming increasingly important [Brown 1996]. Component-based software development technology creation is currently led by NIST under the advanced technology program on Component-Based Software [NIST 2002]. NIST cites many advantages of component-based development that can also be realized for MSA development conditioned on the following:

1. Establishment of a marketplace for component-based MSA development so that M&S technology users can realize significant economic benefits through (a) reduced M&S project costs, (b) enhanced M&S credibility, and (c) expanded applicability of less expensive technology.

2. Increased automation and productivity in MSA development enabling (a) improved M&S quality characteristics, (b) reduced time to develop, test, and certify MSAs, and (c) increased amortization of costs through M&S component reuse.

3. Increased productivity of M&S project teams by (a) permitting specialists in the application domain to create higher fidelity M&S components, and (b) providing a focus on discourse in M&S development at a level far more comfortable to application domain users than a programming language.

4. Expanded markets for the producers of MSAs and M&S components by promoting (a) the creation of systematically reusable M&S components, (b) increased interoperability among M&S software and non-M&S software products, and (c) convenient and ready adaptation of M&S components.

Component-based development is an effective and affordable way of creating MSAs and conducting MSA verification, validation, and certification (VV&C) [Balci *et al.* 2002; Glasow and Pace 1999]. A verified and validated M&S component can substantially decrease the MSA VV&C effort when reused. Such practice can significantly decrease the time and cost of MSA development. Component-based MSA development technology might be the "silver bullet" for effective and affordable MSA VV&C.

Similar to the manner accreditation and certification are carried out in other disciplines, Balci *et al.* [2002] propose the scheme shown in Figure 1. The scheme assumes two scenarios: (a) MSA development under contract, and (b) a marketplace where developers fabricate reusable M&S components for sale, and MSAs are developed by way of component reuse.

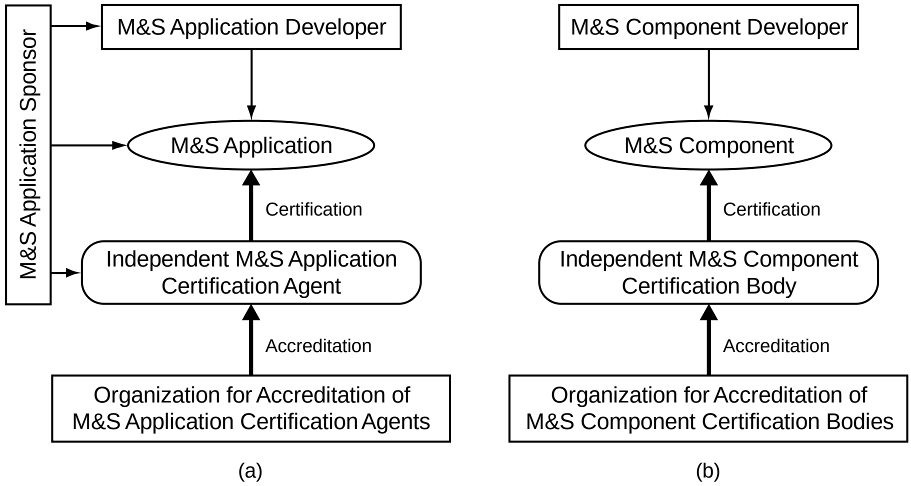

Fig. 1. Accreditation and certification practice.

Under scenario (a), an MSA sponsor hires an independent MSA certification agent, which is accredited by an external accreditation authority to conduct the required certification. Currently, this scenario is practiced in DoD under the label of “accreditation”, where no authority exists to accredit the work of the agent. Certainly, the MSA sponsor should be assured that the practice of the certification agent satisfies minimum standards established by an independent accreditation authority yet to be founded in the United States.

Under scenario (b), an M&S component developer fabricates a reusable M&S component for sale. The buyer of such a component seeks assurance about the component quality. An independent M&S component certification body can provide such an assurance by awarding a “mark of conformity” (e.g., certified to be HLA compliant), a “seal of approval”, or a “certificate of excellence.” There may be many certification bodies, which should provide certifications that are unbiased, fair, cost-effective, and reproducible. Therefore, the certification bodies should be accredited by an external accreditation authority. For example, the SGS International Certification Services (SGS-ICS) group of companies (http://www.sgsgroup.com/sgsgroup.nsf/pages/home.html) is a leading international certification body, which is accredited in the United Kingdom by the National Accreditation Council for Certification Bodies (NACCB) and in Belgium by the Dutch Council for Certification (RvC).

In the United States, for example, the Accreditation Board for Engineering and Technology (ABET) accredits educational programs that award diplomas and certificates. ABET [2002] states that “the diversity of educational programs in the United States is one strength of the American educational system. Such a large selection of educational offerings makes quality a vital issue. Accreditation is the quality assurance that education is meeting minimum standards.”

When the component-based MSA development technology is commonly used and an M&S component marketplace is established, organizations such as SGS-ICS can be founded to independently perform M&S component certification. Organizations similar to NACCB, RvC, and ABET can be founded to accredit, regulate, and monitor the M&S component certification bodies.

Certification of software products by independent evaluation has been practiced in the software industry since early 1990s, especially in Europe and recently in the United States. The following references describe how certification of software products and components is currently practiced: [ISACC 1999; Loesh *et al.* 1999; Poore, Mills, and Mutchler 1993; Rae, Robert, and Hausen 1995; Rodríguez-Dapena 1999; Vermesan 1997, 1998; Voas 1998a,b,c, 1999a,b,c,d, 2000a,b; Wakid, Kuhn, and Wallace 1999; Wallace 1999; Wohlin and Regnell 1998; Wohlin and Runeson 1994].

Certification of M&S products by independent evaluation has been practiced within DoD under the name of M&S accreditation. Certification of M&S products is a very complex process and poses significant technical challenges for M&S program managers, engineers, and practitioners [Balci *et al.* 2002]. It involves the measurement and evaluation of hundreds of qualitative and quantitative elements, mandates subject matter expert (SME) evaluation, and requires the integration of different evaluations. Planning and managing such measurements and evaluations require a unifying methodology and should not be performed in an *ad hoc* manner [Balci *et al.* 2002].

This paper presents such a methodology. The methods, rules, and postulates that make up the methodology are described in section 2. Section 3 presents a strategy for creating indicators for MSA acceptability assessment for certification and introduces the higher levels of a generic hierarchy of indicators that can be used under the methodology for certification of any kind of MSA. Section 4 provides guidance in applying the methodology based on the author's experience. Conclusions are given in Section 5.

2. THE METHODOLOGY

Preliminary research for the methodology described herein was conducted by the author between 1983 and 1991. Between 1992 and 1995, an earlier version of the methodology was developed for application to complex system design evaluation at Virginia Polytechnic Institute and State University (Virginia Tech) under funding from the Naval Surface Warfare Center Dahlgren Division (NSWCDD) [Talbert 1995]. A software tool, Evaluation Environment™ [Orca Computer 1999a,b], was designed and developed at Orca Computer, Inc. with significant improvements to the methodology under funding from NSWCDD between 1997 and 1999. The software tool facilitates the application of the methodology. The methodology and the tool have been used since 1999 in many evaluation projects under the U.S. National Missile Defense program.

The Merriam-Webster dictionary defines methodology as "a body of methods, rules, and postulates employed by a discipline **:** a particular procedure or set of procedures." Our methodology consists of the following body of methods, rules, and postulates:

- employment of subject matter experts,
- construction of a hierarchy of indicators,
- relative criticality weighting of indicators using the Analytic Hierarchy Process,
- using a rule-based expert knowledge base with an object-oriented specification language,
- assignment of crisp, fuzzy, and nominal scores for the indicators,
- aggregation of indicator scores,
- graphical representation of the indicator scores and weights,
- hypertext certification report, and
- interpretation of the results.

Each part of the methodology is described in a separate section below.

2.1 Employment of Subject Matter Experts

Subject matter experts (SMEs) are commonly employed for M&S evaluation for certification [Glasow 1998; Pace 1998]. Under the methodology, the knowledge and experience of SMEs are utilized for:

- constructing a hierarchy of indicators,
- relative criticality weighting of indicators,
- building a rule-based expert knowledge base, and
- assigning scores for the indicators.

SMEs should be employed to cover all areas of the problem domain and all phases of the development life cycle. For example, for a ground-based radar (GBR) simulation, technical SMEs knowledgeable about the GBR systems and operational SMEs and veterans who have intimate knowledge and experience about the operations of the GBR systems should be employed. In addition, technical SMEs knowledgeable about MSA development should be employed. A different set of SMEs may be selected for a different phase of the development life cycle such as requirements, design, implementation, and integration.

Usually an SME is qualified to assess some of the leaf indicators in the hierarchy. For example, a technical SME specializing in M&S designs may be designated to assess only those leaf indicators dealing with the M&S design assessment. Therefore, SMEs must be identified to score on a particular leaf indicator based on their expertise as depicted in Figure 2.

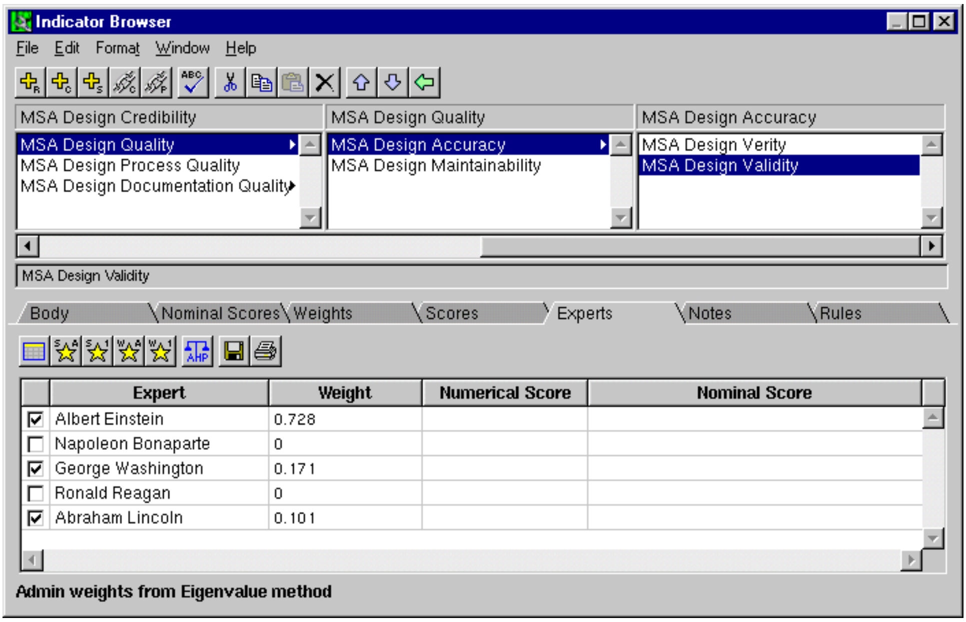

Fig. 2. Selecting and weighting SMEs qualified to score on a leaf indicator.

The SMEs who are qualified to assess a leaf indicator may have different levels of expertise. One SME may be more knowledgeable than another; therefore, his or her assessment should be taken into consideration more than the other's. Thus, the qualified SMEs must be weighted among themselves. Figure 2 depicts the relative criticality weighting of the three qualified SMEs using the Eigenvalue method underlying the Analytic Hierarchy Process discussed in section 2.3.

2.2 Construction of a Hierarchy of Indicators

Measurement and evaluation of qualitative and quantitative concepts is an area of interest in many disciplines. Examination of these disciplines reveals a common approach that uses, what we call, indicators. An *indicator* is an indirect measure of a qualitative concept (e.g., M&S design quality) or a direct measure of a quantitative concept (e.g., utilization). The following terms are used in the designated disciplines to imply what we mean by indicators: metrics (software engineering); measures, indexes (computer performance evaluation); measures of effectiveness, measures of merits, measures of performance (engineering); and indicators (economics, psychometrics). The term "indicator" is used in this paper to be synonymous to any one of the terms listed above.

Figure 3 shows how a qualitative concept can be measured by using a hierarchy of indicators. Since the qualitative concept cannot be directly measured, a set of indicators is created to measure it indirectly at the top level. However, the top-level indicators may still not be directly measurable and therefore, a set of indicators is created to measure each one of them. Those indicators that are not directly measurable in the new set are further decomposed into other indicators. Decomposition of indicators continues until the leaf indicators (i.e., the ones that are not further decomposed) can be directly measured or assessed. This decomposition results in a hierarchy of indicators with the structure depicted in Figure 3. Note that the structure is not a tree. An indicator may influence more than one parent indicator.

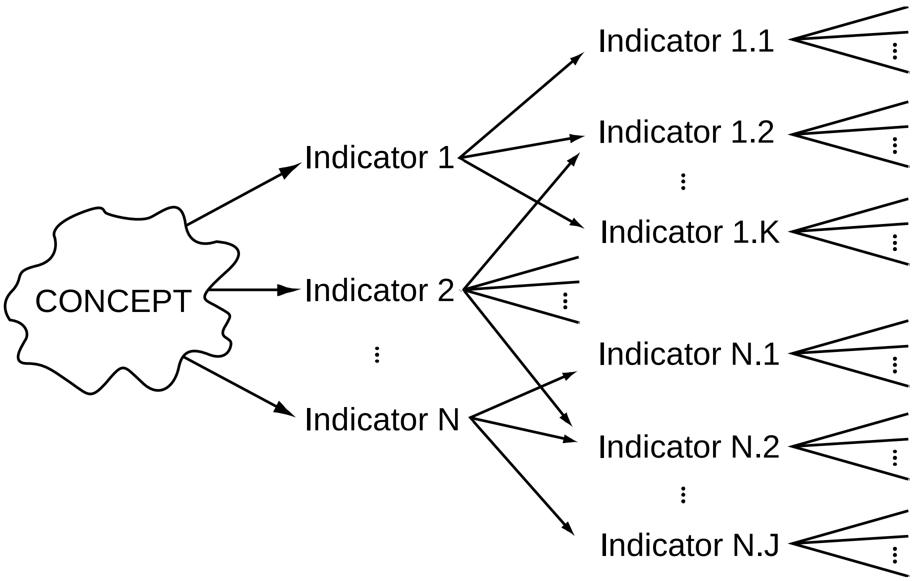

Fig. 3. A hierarchy of indicators.

Three types of indicators exist in the hierarchy: root, branch, and leaf.

1. The *root indicator* is at the apex of the hierarchy representing the qualitative concept. For MSA certification, the root indicator is the MSA acceptability that must be measured and evaluated to formulate a certification decision.

2. A *branch indicator* is one that has at least one parent indicator and at least one child indicator.

3. A *leaf indicator* is one that has at least one parent indicator and no child indicator.

Only the leaf indicators need to be assessed in the hierarchy. The assessments of leaf indicators are aggregated throughout the hierarchy in a bottom-up fashion as discussed in section 2.6.

A certification decision for an MSA is made based on the results of the acceptability assessment of the MSA. MSA acceptability is a qualitative concept that cannot be directly measured or assessed. Therefore, indicators are created to indirectly measure it. The following indicators are proposed at the top level:

- MSA Formulated Problem Credibility
- MSA Requirements Credibility
- MSA Application Credibility
- MSA Experimentations Credibility
- MSA Project Management Quality
- MSA Cost
- MSA Risk

However, neither of these indicators can be directly measured or assessed requiring further decomposition of each into a set of other indicators. The higher levels of a generic hierarchy of indicators for assessing the acceptability of any kind of MSA are presented in section 3.

2.3 Relative Criticality Weighting of Indicators Using AHP

As depicted in Figure 3, a child indicator may have more than one parent and it influences the scores of all of its parent indicators. For example, the model complexity indicator may influence the scores of its parent indicators: model development cost, model maintainability, model reusability, model reliability, and model testability.

A child indicator may influence the score of a parent indicator differently from the other sibling indicators. For example, the effect of model complexity on model reusability may be much higher than its effect on model development cost. Therefore, the child indicators must be weighted among themselves in their influence on the score of a parent indicator. Since a child indicator can influence more than one parent indicator, it can have more than one weight.

Weights are used to express a child indicator's level of influence. A relative *weight* is a fractional value between zero and one. The weights of the child indicators belonging to the same parent must sum to one. Since the weights are assigned relative to the sibling indicators to express criticality of influence on the parent indicator, the process is called *relative criticality weighting*.

Given a list of *n* indicators, it is very difficult for an SME to come up with weights especially when $n > 5$. To facilitate the relative criticality weighting among *n* indicators or decision elements, the mathematical approach called the Analytic Hierarchy Process (AHP) is often used in the multicriteria decision making field [Saaty 1980, 1990, 1994]. AHP enables the SME to make pairwise comparisons of importance between sibling indicators and computes the weights based on the SME's pairwise comparisons using methods such as the Eigenvalue, Mean Transformation, and Row Geometric Mean methods.

Saaty's pairwise comparison scale is described in Table I. This scale is considered to be the standard in the published literature. The use of the scale and the pairwise comparisons are illustrated in Figure 4.

Table I. Pairwise Comparison Scale

Comparative Importance	*Definition*	*Explanation*
1	Equally important	Two sibling indicators equally influence the parent indicator's score.
3	Moderately more important	One sibling indicator is moderately more influential than the other.
5	Strongly more important	One sibling indicator has stronger influence than the other.
7	Very strongly more important	One sibling indicator has significantly more influence over the other.
9	Extremely more important	The difference between influences of the two indicators is extremely significant.
2, 4, 6, 8	Intermediate judgment values	Judgment values between equally, moderately, strongly, very strongly, and extremely.
Reciprocals		If *v* is the judgment value when *i* is compared to *j*, then *1/v* is the judgment value when *j* is compared to *i*.

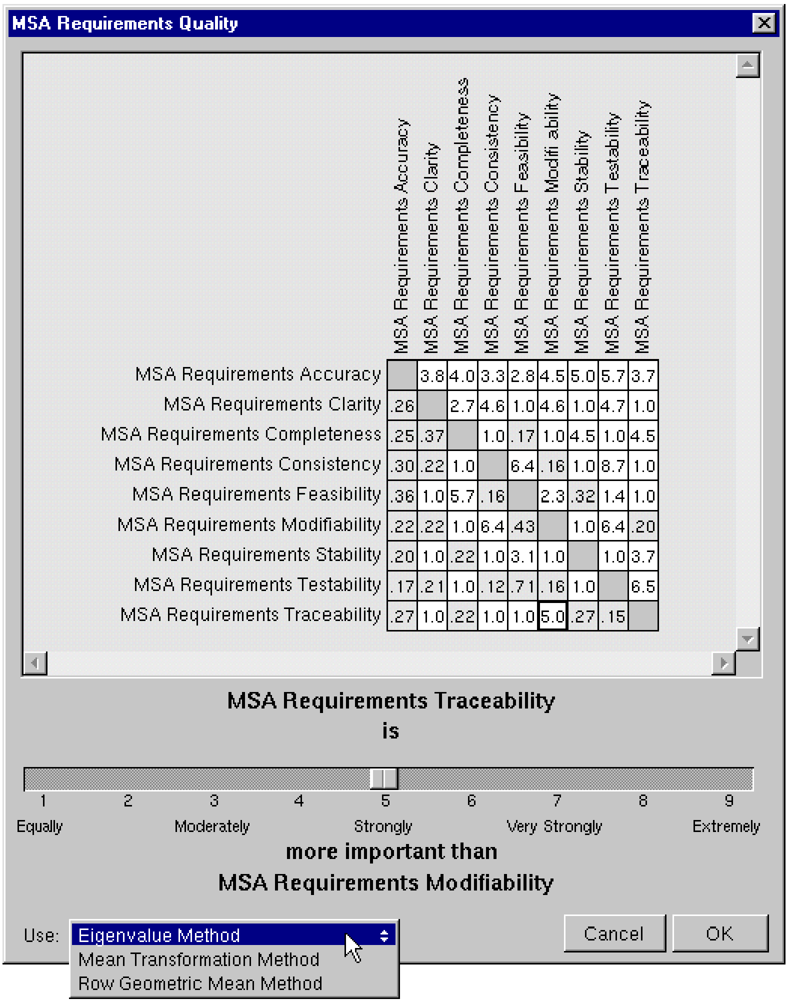

Fig. 4. Pairwise comparisons of indicators under AHP.

The sibling indicators to be weighted are presented in the form of a matrix as shown in Figure 4. Originally, each matrix cell has a value of 1 indicating equal weighting. The SME clicks on a matrix cell to display the names of the two corresponding indicators for comparison. Figure 4 illustrates the comparison of “MSA Requirements Traceability” versus “MSA Requirements Modifiability.” If the relationship is the reverse, i.e., modifiability is more important than traceability, then the corresponding cell on the other half of the matrix is selected. A judgment is specified by clicking on the sliding bar scale or by sliding the knob on the scale. The reciprocal of the judgment value is given to the corresponding cell on the other half of the matrix. Once the SME has performed all pairwise comparisons, a method (e.g., Eigenvalue Method) is selected from the pop-up menu and the OK button is clicked, upon which the fractional weights are computed by using the AHP technique.

SMEs may not be consistent in their judgments [Saaty 1990]. Before computing the weights, the degree of inconsistency is measured by the Inconsistency Index (II). Perfect consistency implies a value of zero for II. However, perfect consistency cannot be demanded since, as human beings, we are often biased and inconsistent in our subjective judgments. Therefore, the SME is not alerted about the inconsistency if II $\leq$ 0.1. For II values greater than 0.1, the SME is alerted and given a chance to revise his or her judgments.

2.4 Using a Rule-Based Expert Knowledge Base

The MSA acceptability should be assessed to make a certification decision by incorporating knowledge about

- the problem domain for which the MSA is intended,
- relationships and dependencies among the indicators,
- technical characteristics of the MSA, and
- the requirements under which the MSA is developed.

An object-oriented language was developed and implemented for rule-based knowledge specification [Orca Computer 1999a, b]. Using this language, rules can be constructed to provide four types of action:

1. *Prompters* are actions that require the SME to enter an explanation, justification, or description during the evaluation.
2. *Triggers* are internal actions that are not visible to the SME; they cause the occurrence of internal actions such as the recording of some values, comparisons, computations, and knowledge-based inferencing.
3. *Alerters* are actions that notify the SME about undesirable conditions.
4. *Informers* are actions that provide useful information to the SME during the evaluation.

A set of rules can be constructed to guard against inconsistencies and biasedness that may be exhibited by SMEs during the evaluation. Other rules can be created and executed manually to query the information contained within a hierarchy of indicators.

2.4.1 Class Library

A class library is provided to enable the user to access the built-in classes shown in Figure 5. This Figure also shows the class inheritance structure. A class has its own characteristics (i.e., instance variables, class variables) and behavior (i.e., instance methods, class methods), and inherits other characteristics and behavior from its ancestor classes. For example, the class `Expert` inherits the characteristics and behavior of its parent class `ObjectWithValues` and grandparent class `Object`.

```
Object
      String
      Iterator
      Range
      ObjectWithValues
            Expert
            Indicator
            Project
      Console
```

Fig. 5. Class inheritance hierarchy.

When an object (e.g., an SME, an indicator, a project, an iterator) is instantiated from its corresponding class, it inherits all of the characteristics and behavior of its class. This implies that the instantiated object can respond to all messages (i.e., method invocations) provided by its class. Altogether, 81 methods are available for use in the class library.

2.4.2 Message Passing

All objects instantiated from the various classes communicate with each other via message passing. Each object has a unique object reference that is used for sending a message to that object. Table II shows the message passing forms that can be used as statements or in expressions. The brackets imply message passing and can be embedded. For example, `[console print:[sme name]]` indicates that the message called "name" is sent to the object pointed to by the object reference "sme". This message passing returns the name of the subject matter expert, which becomes the value of the parameter of the method called "print". The colon indicates a parameter specification for a method. Then the "print" message with the parameter value just returned, is sent to the console object, which prints the name to the console window.

Table II. Message Passing Forms

Message Passing Forms	*Examples*	*Can be Used as a Statement?*	*Can be Used in an Expression?*
[object message]	*As a statement:* [console print:[sme name]]; *In an expression:* set rootIndicator to [project topIndicator];	Yes	Yes
[message **of** object]	if [[expertNamed:"Albert Einstein" of project] hasCompletedEvaluation] then …	No	Yes
[message]	set childIndicators to [children];	No	Yes
tell object **to** message	tell console to println:"The score range = " & [aggregateScore of [childNamed:"Risk"]];	Yes	No
tell object **that** message	if [[aggregateScore] low] < 60 then tell console that warning:"Score is too low";	Yes	No

If no object reference is specified in a message passing, e.g., [children], the object pointed to by the default object reference "self" receives the message. "Self" is an object reference pointing to the indicator object for which the rule is written. Thus, the message passing [children] is equivalent to [children of self] or [self children].

A message can be sent to any indicator in the hierarchy. If the object reference of the indicator of interest is stored in a variable, the variable name, e.g., "rootIndicator", is used for sending the message; otherwise, the object reference can be inserted in a rule by drag-and-drop. After clicking on the insertion point in a rule, while holding down the Alt key, the name of the indicator of interest in the browser is clicked, dragged, and dropped at the insertion point. A small icon is inserted representing the object reference of the indicator.

2.4.3 Rule Specification

A *rule* is specified in the form of a condition and action pair—specifically for a particular indicator. For example, three rules are shown in Figure 6 for the indicator "MSA Requirements Consistency". The left box is used for condition specification, the right box is used for action specification, and the lower box is used for rule documentation. Using the up and down arrow buttons, the rules can be ordered.

A condition is specified as a method that must return true or false upon execution. An action is also specified as a method but it does not return true or false; it simply performs a set of actions.

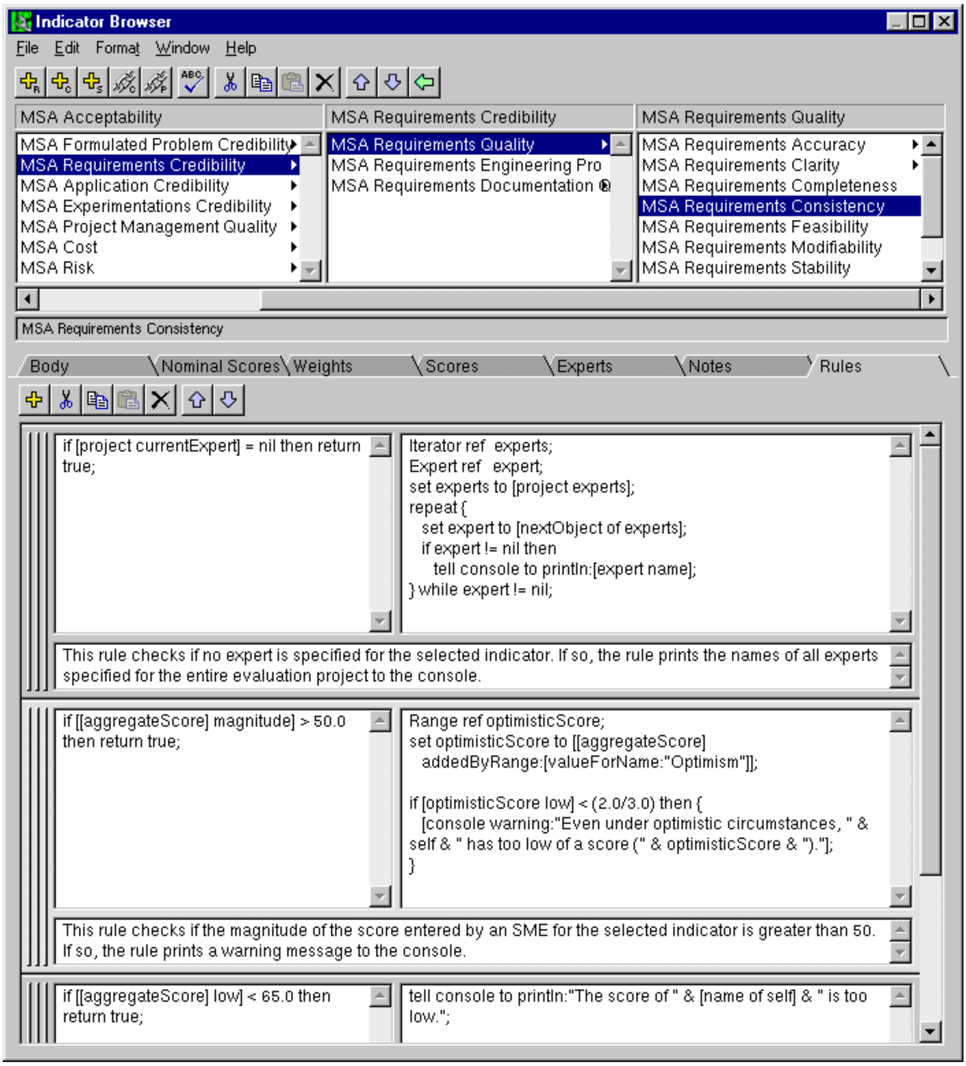

Fig. 6. Example rules.

When the SME assigns a score to or changes the weight of an indicator, the conditions of the rules specified for that indicator are scanned from top to bottom. If a condition method returns true, the corresponding action method is executed.

Figure 7 shows a more complex action method that verifies the consistency of indicator weights in the entire indicator hierarchy. This action method can be specified with any indicator in the hierarchy with a condition method "`return true;`". The rule is executed manually.

```
Iterator ref     descendants, children;
Indicator ref    indicator, childIndicator;
Real             sumOfWeights;

set indicator to [project topIndicator];
set descendants to [indicator descendants];

repeat while indicator != nil do {
     if [numberOfChildren of indicator] > 0 then {
          tell console to println:"\"" & [indicator name] & "\"";
          set children to [indicator children];
          set childIndicator to [nextObject of children];
          set sumOfWeights to 0.0;
          repeat while childIndicator != nil do {
               tell console to println:"\t\"" & [childIndicator name]
                    & "\"'s weight = " & [childIndicator
                    defaultWeightForParent:indicator];
               add [childIndicator defaultWeightForParent:indicator]
                    to sumOfWeights;
               set childIndicator to [nextObject of children];
          }
          if (1 - sumOfWeights) > 0.00001 or
               (sumOfWeights - 1) > 0.00001 then
                    tell self to emitError:"The weights of the child
                         indicators of \"" & [indicator name] & "\"
                         do not sum to 1.\n";
                    // emitError: message stops execution of the remaining statements.
     }
     set indicator to [nextObject of descendants];
}

tell console to println:"\n*** Congratulations! All indicator weights
          are found to be consistent. ***\n";
```

Fig. 7. An action method verifying the consistency of indicator weights.

2.5 Assignment of Crisp, Fuzzy, and Nominal Scores for the Indicators

For a leaf indicator, an SME can assign a crisp, fuzzy or nominal score, defined as follows.

1. A *crisp score* is a single real value between 0 and 100 (e.g., 75.8).
2. A *fuzzy score* is an interval of real values within the range of 0 and 100 (e.g., [72.5, 83.7]). The end points of the interval are always inclusive.
3. A *nominal score* is a named score with a predefined crisp or fuzzy value.

Table III shows an example set of nominal scores with predefined numerical fuzzy values or score ranges. Note that the high and low scores in adjacent score ranges are the same since the score values are real. The plus sign means “more” and implies an increase in the goodness or badness of the assessment. The minus sign means “less” and implies a decrease. The plus sign goes upward for positive assessments, and downward for negative assessments; for example, “Poor +” means more poor and “Poor –” means less poor. Therefore, “Poor +” has a lower score range than “Poor –”.

Table III. Example Nominal Scores

Nominal Score	*Numerical Score*
Excellent +	[97 .. 100]
Excellent	[94 .. 97]
Excellent –	[90 .. 94]
Good +	[87 .. 90]
Good	[84 .. 87]
Good –	[80 .. 84]
Satisfactory +	[77 .. 80]
Satisfactory	[74 .. 77]
Satisfactory –	[70 .. 74]
Poor –	[67 .. 70]
Poor	[64 .. 67]
Poor +	[60 .. 64]
Unacceptable –	[40 .. 60]
Unacceptable	[20 .. 40]
Unacceptable +	[0 .. 20]

Many approaches exist for determining a score for a leaf indicator. Four approaches are described below in no particular order.

2.5.1 Testing

Two types of testing can be performed: software testing and field testing. In *software testing*, tests are designed to perform either verification or validation or both. More than 100 techniques are available for MSA software testing [Balci 1998; Binder 2000]. The test results are transformed into value judgments, which determine scores assigned to the leaf indicators.

Field testing places the MSA in an operational situation (e.g., actual military exercises) for the purpose of collecting as much information as possible for testing the MSA. For example, for acceptability assessment of an MSA being developed for the National Missile Defense system design evaluation, actual field tests such as integrated flight tests (IFTs), integrated ground tests (IGTs), and integrated system tests (ISTs) were performed. The data collected during the IFTs, IGTs, and ISTs were used in assessing many leaf indicators.

2.5.2 Direct Measurement

Certain elements of an MSA can be directly measured, once the executable version is created. For example, for a distributed interactive MSA, the following elements can be directly measured: animation speed, network bandwidth, execution time, amount of computer memory used, and amount of disk space used. The measured value can be a crisp value or a statistical confidence interval (fuzzy value). An SME can judge the goodness of the value and assign a crisp, fuzzy or nominal score to express his or her judgment for the indicator.

If the system represented by the MSA exists and data can be collected about its input-output behavior, direct measurement of the system can be used. The data collected by instrumenting (i.e., inserting probes, breakpoints, and traps into) the real system are analyzed to make value judgments based on which scores are assigned to the leaf indicators.

2.5.3 Analysis

The analysis can be carried out by using a variety of techniques such as mathematical analysis, statistical analysis, probabilistic analysis, and metrics analysis. For example, the analysis results obtained by using a queuing theory model (e.g., M/M/s) can be used in judging the validity of an MSA component. The analysis results are interpreted by SMEs and converted into value judgments based on which scores are assigned to the leaf indicators.

Metrics are commonly used in the field of software engineering. A *metric* is a formula computed and evaluated to measure a certain aspect of the entity of interest (e.g., model complexity, model cohesion, model coupling). Metrics are commonly used for comparative analysis and metric values must be interpreted. An SME can interpret a metric value and assign a score to a leaf indicator.

2.5.4 Examination

An SME, based on his or her technical M&S expertise, can examine the MSA and assign a score for a leaf indicator. The SME is assumed to have expertise and experience about a particular phase, several phases, or all phases of the MSA development life cycle. For example, an SME on MSA implementation with years of experience can judge the goodness of an MSA implementation.

Certain indicators can only be assessed by SMEs who possess domain specific operational knowledge. For example, certain quality indicators of a GBR simulation can only be assessed by veterans who possess operational experience with the GBR systems. Such SMEs, although difficult to find, can be asked to examine the MSA and assign scores to the leaf indicators.

2.6 Aggregation of Indicator Scores

Only the leaf indicators are scored on. The scores of the leaf indicators are aggregated in a bottom-up fashion throughout the indicator hierarchy. The aggregation is carried out by the weighted average of scores with respect to the weights of child indicators and with respect to the weights of SMEs.

2.7 Graphical Representation of the Indicator Scores and Weights

Variations of Kiviat graphs are used to graphically represent the indicator scores and weights, as depicted in Figure 8. Each radius of the circle represents an indicator with its name shown just outside of the circumference. The center of the circle represents a score of zero and the circumference intersection point represents a perfect score of 100. The indicator score is designated on the radius either as a crisp value or an interval. The score closest to the circumference is the high score, the score closest to the center of the circle is the low score, and the middle score is the average score as shown in Figure 8.

Kiviat graphs enable the joint visual assessment of many sibling indicators. The graphical shape formed by connecting the indicator scores represents an overall assessment. The shape visually displays the high, average, and low scores of all indicators. By examining the form of the shape, one can visually perceive how well the assessment is with respect to each indicator. A perfect, ideal Kiviat graph has the shape of a circle, where each indicator has a perfect score of 100.

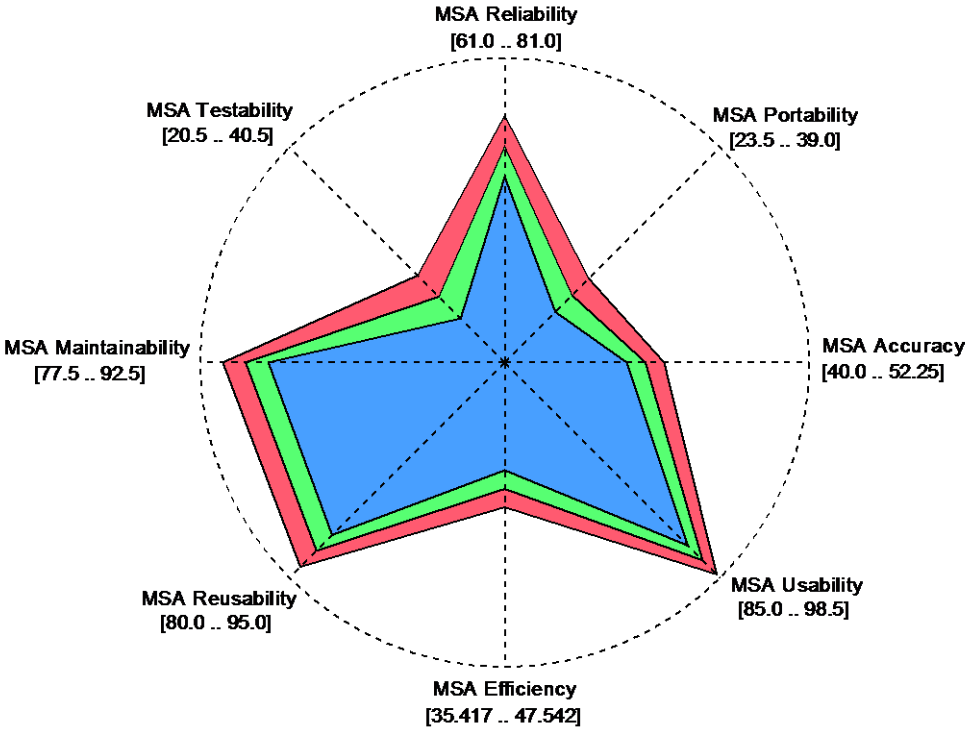

Fig. 8. An example Kiviat graph showing fuzzy scores for the sibling indicators.

2.8 Hypertext Certification Report

The Evaluation Environment™ software tool [Orca Computer 1999a] can generate a hypertext certification report in the HyperText Markup Language (HTML) and Rich Text Format (RTF). The report includes

- Certification project documentation
- Information about the SMEs
- Hierarchical list of indicators
- Alphabetical list of indicators
- Leaf indicators report
- Kiviat graphs of
    - Aggregate scores for an indicator's children
    - Default weights for an indicator's children
    - Scores assigned by the SMEs for an indicator
    - Weights of the SMEs for an indicator

The report can instantly be published on the World Wide Web (Web) and viewed by all people involved in the MSA project using a Web browser. When publishing the certification report on the Web, the identity of each SME can be hidden.

2.9 Interpretation of the Results

MSA acceptability assessment results are descriptive in nature, and they need to be interpreted to reach a certification decision. Often a certification decision is a confidence building activity. As such, the MSA acceptability root indicator score should be considered as one of the factors affecting the MSA certification decision. Other factors should also be considered in reaching a final MSA certification decision. Example factors include: independence in assessment, quality of the hierarchy of indicators, quality of SMEs, the scores assigned by SMEs, aggregated scores throughout the hierarchy, weights, and SME comments.

Presentation of indicator scores as interval values enables worst-case (pessimistic) analysis, most-likely analysis, and best-case (optimistic) analysis. Using the low-end values of the interval scores, the MSA acceptability can be judged under the worst-case scenario. Use of the mid-point values in the analysis provides a most likely assessment of acceptability. Using the high-end values of the interval scores, the acceptability can be assessed under the best-case scenario.

3. A HIERARCHY OF INDICATORS

Two of the strategic directions in verification, validation, and accreditation (VV&A) research and practice advocated by Balci *et al.* (2002) are:

- VV&A should be expanded from accuracy-centered assessment to quality-centered assessment.
- VV&A should be expanded from product-centered assessment to (product / process / project)-centered assessment.

The IEEE Standard 1059 indicates that “Software verification and validation employs review, analysis, and testing techniques to determine whether a software system and its intermediate products comply with requirements. These requirements include both functional capabilities and quality attributes” [IEEE 1993, p. 4]. The IEEE Standard 1059 includes quality assessment within the verification and validation (V&V) activities by listing 19 quality attributes (also called quality characteristics) including efficiency, interoperability, maintainability, reliability, reusability, testability, and usability.

A generic MSA development life cycle consists of processes and (work) products as shown in Figure 9. “Process” refers to a series of activities conducted to create a life-cycle product, such as engineering the MSA requirements, designing the MSA, or creating the MSA executables (implementation). “Product” refers to a different representation of the MSA during its development life cycle, such as the MSA requirements specification, MSA design specification, or MSA code (executables).

$\text{PRODUCT}_{\text{input}}$ →	PROCESS →	$\text{PRODUCT}_{\text{output}}$
Communicated Problem	Problem Formulation	Formulated Problem
Formulated Problem	Requirements Engineering	Requirements Specification
Requirements Specification	Conceptual Modeling	Conceptual Model
Conceptual Model	Design	Design Specification
Design Specification	Implementation	Executables
Executables	Integration	MSA

Fig. 9. MSA development life cycle products and processes.

An MSA representation is transformed from one product form (e.g., design specification) into another (e.g., executables) by carrying out a process (e.g., implementation) during the development life cycle.

The MSA acceptability assessment for certification should be concerned not only with the assessment of product quality, but also with the quality of the process used to create the product. The quality of the MSA project management also affects the overall MSA acceptability. Certainly documentation plays a critical role in the acceptability assessment and documentation quality should also be considered.

Consequently, the strategy for creating a generic hierarchy of indicators for MSA acceptability assessment for certification dictates the assessment of (a) quality of the product, (b) quality of the process used to create the product, (c) quality of the MSA project management, and (d) quality of the documentation that describes the product, process, and quality assurance of the product and process. The strategy is illustrated in Table IV.

Table IV. Strategy for Creating Indicators for MSA Acceptability Assessment for Certification

<table>
<tr><td rowspan="8">Credibility</td><td rowspan="3">Product Quality</td><td rowspan="2">Accuracy</td><td>Verity</td></tr>
<tr><td>Validity</td></tr>
<tr><td>Quality Characteristic 2
Quality Characteristic 3
:
Quality Characteristic k</td><td>…</td></tr>
<tr><td>Process Quality</td><td>Quality of Approach Used
:</td><td>…</td></tr>
<tr><td>Project Quality</td><td>Quality of Human Resource Management
:</td><td>…</td></tr>
<tr><td rowspan="3">Documentation Quality</td><td>Product Documentation Quality</td><td>…</td></tr>
<tr><td>Process Documentation Quality</td><td>…</td></tr>
<tr><td>Quality Assurance Documentation Quality</td><td>…</td></tr>
</table>

Table IV implies that the MSA acceptability is influenced by the credibility of a phase of the MSA development life cycle. The *credibility* is influenced by the qualities of the product, process, project, and documentation of that phase.

Product quality is the degree to which the product possesses a desired set of characteristics. The first product quality characteristic "product accuracy" is assessed by evaluating product verity and validity. Product verity is evaluated by conducting product verification and product validity is evaluated by conducting product validation.

- *Product verification* deals with the transformational accuracy of that product and addresses the question of "Are we building the product *right*?"
- *Product validation* deals with the representational or behavioral accuracy of that product and addresses the question of "Are we building the *right* product?"

We refer to product verification and product validation as simply V&V throughout the MSA development life cycle.

Other product quality characteristics change from one MSA project to another and are determined based on the MSA requirements and intended uses [Balci and Ormsby 2000].

Process quality is the degree to which the process possesses a desired set of characteristics. The set of desired characteristics depends on the process methodologies and techniques employed by the MSA developer. CMU SEI [1994] has developed the Software Capability Maturity Model (SW-CMM) as an application of the process management concepts of total quality management to software. SW-CMM is now very commonly used in the software industry as a means of judging software development process maturity and quality. Many indicators can be developed based on SW-CMM and applied to the assessment of process quality.

Project quality is the degree to which the project possesses a desired set of characteristics. Project quality is assessed by evaluating characteristics such as cost management, human resource management, integration management, quality management, risk management, and time management.

Document (or *documentation* or *report*) *quality* is the degree to which the document possesses a desired set of characteristics. The quality of a document is mostly determined by the quality of its content; however, other quality characteristics are also important. We have developed a hierarchy of more than 80 indicators to assess the document quality other than the quality of the document's content, which should be assessed in other V&V activities. The top-level indicators of this hierarchy are given as: accessibility, accuracy, completeness, consistency, clarity, maintainability, portability, and readability.

Following the strategy outlined in Table IV, a hierarchy of indicators can be developed for MSA acceptability assessment for certification. At the top level of the hierarchy, the following indicators can be defined to influence the MSA acceptability:

- MSA Formulated Problem Credibility
- MSA Requirements Credibility
- MSA Application Credibility
- MSA Experimentations Credibility
- MSA Project Management Quality
- MSA Cost
- MSA Risk

The acceptability is affected by the credibility of the formulated problem, which is expected to be solved by the creation and use of the MSA. An MSA is created to represent a system defined with respect to the formulated problem. If the formulated problem does not entirely contain the *real* problem, type III error occurs. *Type III error* is the error of solving the wrong problem [Balci and Nance 1985]. Based on the work of Balci and Nance [1985], we have developed a hierarchy of more than 85 indicators to assess the credibility of a given formulated problem.

Credibility assessment of MSA requirements is especially critical for MSAs created for training purposes. This indicator addresses the question of “Will the MSA meet the *real* needs of the application sponsor if it is developed under the specified requirements?" An MSA that does not meet the *real* needs for which it is created cannot be certified.

Credibility assessment of an MSA deals with the assessment of how well the application is developed throughout its entire development life cycle. It includes the assessment of conceptual model credibility, design credibility, implementation credibility, integration credibility, data credibility, configuration management quality, overall product quality, and documentation quality.

Designing and conducting statistical experiments with an MSA are known to be difficult. An MSA may be developed to be perfectly acceptable; but, if the simulation experiments are not properly designed and conducted under the prescribed intended uses [Balci and Ormsby 2000], the simulation results become erroneous. Therefore, the credibility of experimentations with an MSA significantly influences the MSA acceptability.

Quality of MSA project management affects our confidence in the MSA acceptability. Many management quality characteristics influence our confidence including the quality of project management of: planning, integration, scope, time, cost, quality, human resource, communications, risk, and procurement. Quality of the certification (or VV&A) plan and the quality of its execution also affect our confidence in the MSA acceptability [Balci *et al.* 2000].

Certainly, cost is a factor for acceptability. For example, an MSA created for training military personnel may not be acceptable if the cost of its use (e.g., performing training exercises), cost of its maintenance, cost of its technical support, and cost of learning how to use it, is beyond the budget of the application sponsor or potential user.

Any MSA has risks associated with its development. An MSA may not be able to pass the acceptance criteria specified in the contract, it may not satisfy the performance requirements, or different components developed by subcontractors may not be able to be integrated. Such risks should be assessed periodically during the development life cycle and should be minimized as much as possible to increase the probability of acceptance.

4. APPLICATION OF THE METHODOLOGY

There are three major approaches under which the methodology can be applied:

1. The MSA sponsor hires a certification agent *before* the contractor starts the development and requires the agent to apply the methodology throughout the entire MSA development life cycle to assess the MSA acceptability for a given set of intended uses.

2. The sponsor or potential user of an MSA hires a certification agent *after* the contractor completes the development of a new MSA or *after* a contractor modifies an existing MSA, and tasks the agent to assess the MSA acceptability for a given set of intended uses by applying the methodology.

3. The sponsor or potential user of an already developed or legacy MSA hires a certification agent to assess the MSA acceptability for a given set of intended uses by applying the methodology.

Approach 1 provides many advantages, including the following:

- The MSA developer gets feedback for acceptability as the MSA development progresses throughout the life cycle.
- MSA errors and deficiencies are discovered early in the development life cycle resulting in significant cost savings.
- The complexity of MSA acceptability assessment is significantly reduced.
- The Probability of Type II Error (MSA User's Risk) is significantly reduced. *Type II Error* is the error of certifying an MSA when in fact it should not have been certified.
- Communication between the independent certification agent and the MSA developer helps improve the MSA quality.
- The MSA developer is encouraged to implement an effective software quality assurance program.
- MSA product and resource risks are significantly reduced.

Below general step-by-step instructions and guidelines are provided for a certification agent to follow when applying the methodology. The steps may be revised, depending on which of the approaches described above is used.

Step 1. Employ SMEs who have technical and operational knowledge about the problem/application domain and about the development of MSAs in that domain.

Step 2. Working with SMEs, create a hierarchy of indicators specifically for the MSA, either as a whole or in part or for a particular phase of its development life cycle (e.g., requirements, design, implementation) or for a particular MSA build (increment or version) during its development. Note that acceptability, credibility, or quality is always assessed with respect to a set of intended uses [Balci and Ormsby 2000]. Different hierarchies of indicators may be required for different sets of intended uses. Section 3 provides some guidance for creating indicators.

Step 3. Form an independent review panel, and have the panel assess the appropriateness of the hierarchy of indicators created. The panel may consist of SMEs, veterans, engineers, managers, and others. The hierarchy should be judged to determine if: (a) it covers all essential elements of the problem/application domain, (b) it provides sufficient detail in measurement, and (c) the leaf indicators are directly measurable or assessable.

Step 4. If approach 1 is used, provide the MSA developer(s) with a copy of the hierarchy of indicators in advance, preferably as early as possible in the development life cycle, ideally before the development starts. Inform the developer(s) that the certification will be analogous to an "open book, take-home examination" and the indicator hierarchy clearly shows which questions will be on the "examination", i.e., certification. The indicator hierarchy should also be included in the MSA Certification Plan.

Step 5. Working with SMEs and using AHP, determine relative criticality weighting of indicators.

Step 6. Working with SMEs, specify rule-based knowledge by using the object-oriented language in conjunction with a particular indicator in the hierarchy where it is considered to be necessary.

Step 7. Employ SMEs for acceptability assessment. For each leaf indicator in the hierarchy, select the SMEs qualified to score on that leaf indicator, and determine relative criticality weighting of the qualified SMEs using AHP.

Step 8. Generate an evaluation file for a particular SME. The evaluation file contains the hierarchy, highlighting only those indicators the SME is qualified to score on.

Step 9. Together with the relevant documentation of the subject matter being evaluated, send the evaluation files to all SMEs. Using the Evaluation Environment™ *Expert* software tool, an SME opens his or her own evaluation file, and scores on the leaf indicators he or she is qualified to score on. SMEs can also work as a team and evaluate the subject matter altogether. However, each SME enters his or her own score for a leaf indicator.

Step 10. Receive the evaluation files from the SMEs, and import them all into the Evaluation Environment™ *Admin* software tool. All SME scores are automatically integrated and aggregated in a bottom-up manner throughout the entire hierarchy; interval scores are computed for all branch indicators as well as for the root indicator.

Step 11. Using the Evaluation Environment™ *Admin* software tool, generate a hypertext assessment report in HTML and RTF. The report may be published on the World Wide Web, with the option of hiding the identity of each SME.

Step 12. Review the assessment report, and make an acceptability decision. If the evaluation is conducted for the entire MSA, present the MSA acceptability assessment results, with a certification recommendation, to the application sponsor. If the evaluation is conducted for a particular phase of the MSA development life cycle or for a particular MSA build (increment or version), present the results to the MSA developer(s).

Certification is a situation-dependent activity and requires the assessment of many indicators specifically created for a particular MSA and for a particular set of MSA intended uses. Creation of a hierarchy of indicators is not trivial and requires significant experience and knowledge about the subject matter being evaluated.

The evaluation's level of granularity should be determined with respect to the evaluation objectives dictated by the MSA intended uses and MSA project objectives. For example, assume that there are 100 requirements for an MSA. We can specify "MSA requirements consistency" as a leaf indicator implying no further decomposition and expecting a score based on examination of all 100 requirements. This examination can be conducted by using a software tool for requirements analysis and the result can be converted into a score for the indicator. However, if the level of granularity is judged to be too high, then we can decompose the consistency indicator into 100 other consistency indicators, one for each requirement. If this level is judged to be too detailed, then we can partition the requirements into categories and create a consistency indicator for each category. Thus, indicators should be decomposed until a sufficient level of granularity is achieved in the measurement.

Although the methodology is presented for certification of MSAs, it is generically applicable for measurement and evaluation of any subject matter that requires expert judgment. For example, the methodology can also be used for complex system design quality assessment, data quality assessment and certification, government certification, government test and evaluation, software quality assessment, and software risk assessment. For each of the example areas above, a hierarchy of indicators can be created and the methodology can be applied.

5. CONCLUSIONS

A multifaceted methodology is presented for certification of MSAs. The methodology consists of the following facets: utilizing subject matter expert knowledge; employing indicators to assess qualitative and quantitative elements; relative criticality weighting of indicators; using an expert knowledge base during evaluation; assigning crisp, fuzzy, and nominal scores for the indicators; aggregation of scores; graphical representation of the evaluation results; and interpreting the evaluation results by using a hypertext assessment report. The methodology is generically applicable for certification of any kind of MSA, as well as for other applications where evaluation relies on subject matter expertise.

ACKNOWLEDGMENTS

The research and engineering that have led to the development of the certification methodology and the Evaluation Environment™ software tool were funded by the Naval Surface Warfare Center Dahlgren Division at Virginia Tech and Orca Computer, Inc. Contributions of the following people for the creation of the methodology and the software tool are gratefully acknowledged: Anders I. Bertelrud, William H. Farr, Joshua H. Levy, Richard E. Nance, William F. Ormsby, and Michael L. Talbert.

REFERENCES

ABET. 2002. Accreditation Board for Engineering and Technology, Baltimore, MD. (http://www.abet.org)

BALCI, O. 1998. Verification, validation, and testing. In *The Handbook of Simulation*, J. Banks, Ed. John Wiley & Sons, New York, NY, Aug., 335-393.

BALCI, O. AND NANCE, R. E. 1985. Formulated problem verification as an explicit requirement of model credibility. *Simulation 45*, 2 (Aug.), 76-86.

BALCI, O., NANCE, R. E., ARTHUR, J. D., AND ORMSBY, W. F. 2002. Expanding our horizons in verification, validation, and accreditation research and practice. In *Proceedings of the 2002 Winter Simulation Conference* (San Diego, CA, Dec. 8-11). E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, Eds. IEEE, Piscataway, NJ.

BALCI, O. AND ORMSBY, W. F. 2000. Well-defined intended uses: an explicit requirement for accreditation of modeling and simulation applications. In *Proceedings of the 2000 Winter Simulation Conference* (Orlando, FL, Dec. 10-13). J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, Eds. IEEE, Piscataway, NJ, 849-854.

BALCI, O., ORMSBY, W. F., CARR III, J. T., AND SAADI, S. D. 2000. Planning for verification, validation, and accreditation of modeling and simulation applications. In *Proceedings of the 2000 Winter Simulation Conference* (Orlando, FL, Dec. 10-13). J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, Eds. IEEE, Piscataway, NJ, 829-839.

BINDER, R. V. 2000. *Testing Object-Oriented Systems: Models, Patterns, and Tools*. Addison-Wesley, Reading, MA, 1191 pp.

BROWN, A. W., ED. 1996. *Component-Based Software Engineering*. IEEE Computer Society Press, Los Alamitos, CA.

CMU SEI. 1994. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Addison-Wesley, Reading, MA.

DoDI. 1996. DoD modeling and simulation verification, validation, and accreditation. Department of Defense Instruction 5000.61, Apr.

GLASOW, P. A. 1998. Characteristics, selection, and use of subject matter experts. Technical Report 99W0000048, POET WG2K IV&V Paper No. 25, The MITRE Corporation, McLean, VA, June.

GLASOW, P. A. AND PACE, D. K., EDS. 1999. Report on the simulation validation (SIMVAL) 1999—making VV&A effective and affordable—mini-symposium and workshop. Military Operations Research Society, Alexandria, VA, May.

IEEE. 1993. *IEEE Guide for Software Verification and Validation Plans*. IEEE Standard 1059-1993. IEEE, Piscataway, NJ.

ISACC. 1999. *Proceedings of the 1st International Software Assurance Certification Conference*. (Feb. 28 - Mar. 2, Chantilly, VA). Reliable Software Technologies, Sterling, VA.

LOESH, R. E., GOSNELL, A. B., BENOIST, T. F., WYSKIDA, R. M., AND JOHANNES, J. D. 1999. An engineering approach to critical software certification. In *Proceedings of the 32nd Annual Hawaii International Conference on System Sciences* (Jan. 5-8, Maui, Hawaii). IEEE Computer Society, Washington, DC.

NIST. 2002. Component-Based Software Advanced Technology Program. National Institute of Standards and Technology, Gaithersburg, MD. (http://www.atp.nist.gov/atp/focus/cbs.htm)

ORCA COMPUTER. 1999a. *Evaluation Environment User's Guide*. Orca Computer, Inc., Blacksburg, VA.

ORCA COMPUTER. 1999b. *Evaluation Environment Subject Matter Expert User's Guide*. Orca Computer, Inc., Blacksburg, VA.

PACE, D. K. 1998. Verification, validation, and accreditation. In *Applied Modeling and Simulation: An Integrated Approach to Development and Operation*. D. J. Cloud and L. B. Rainey, Eds. McGraw-Hill, New York, NY, 369-410.

POORE, J. H., MILLS, H. D., AND MUTCHLER, D. 1993. Planning and certifying software system reliability. *IEEE Software 10*, 1 (Jan./Feb.), 88-99.

RAE, A., ROBERT, P., AND HAUSEN, H.-L. 1995. *Software Evaluation for Certification: Principles, Practice, and Legal Liability*. McGraw-Hill, London, UK.

RODRÍGUEZ-DAPENA, P. 1999. Software safety certification: a multidomain problem. *IEEE Software 16*, 4 (July/Aug.), 31-38.

SAATY, T. L. 1980. *The Analytic Hierarchy Process*, McGraw-Hill, New York, NY.

SAATY, T. L. 1990. *Multicriteria Decision Making: The Analytic Hierarchy Process*, RWS Publications, Pittsburgh, PA.

SAATY, T. L. 1994. *Fundamentals of Decision Making and Priority Theory with the Analytic Hierarchy Process*. RWS Publications, Pittsburgh, PA.

TALBERT, M. L. 1995. A methodology for the measurement and evaluation of complex system designs. Ph.D. Dissertation, Department of Computer Science, Virginia Tech, Blacksburg, VA, Dec., 324 pp.

VERMESAN, A. I. 1997. Quality assessment of knowledge-based software: some certification considerations. In *Proceedings of the 3rd International Software Engineering Standards Symposium* (June 1-6, Walnut Creek, CA). IEEE Computer Society, Washington, DC, 144-154.

VERMESAN, A. I. 1998. Software certification for industry – verification and validation issues in expert systems. In *Proceedings of the 9th International Workshop on Database and Expert Systems Applications* (Aug. 26-28, Vienna, Austria). IEEE Computer Society, Washington, DC.

VOAS, J. M. 1998a. Certifying off-the-shelf software components. *IEEE Computer 31*, 6 (June), 53-59.

VOAS, J. M. 1998b. Certifying high assurance software. In *Proceedings of the 22nd International Computer Software and Application Conference* (Aug. 17-21, Vienna, Austria). IEEE Computer Society, Washington, DC.

VOAS, J. M. 1998c. The software quality certification triangle. *Crosstalk 11*, 11 (Nov.), 12-14.

VOAS, J. M., ED. 1999a. Special issue on software certification. *IEEE Software 16*, 4 (July/Aug.).

VOAS, J. M. 1999b. Guest editor's introduction: certification: reducing the hidden costs of poor quality. *IEEE Software 16*, 4 (July/Aug.), 22-25.

VOAS, J. M. 1999c. Certifying software for high-assurance environments. *IEEE Software 16*, 4 (July/Aug.), 48-54.

VOAS, J. M. 1999d. Can generic software be assured? In *Proceedings of the 23rd Annual International Computer Software and Applications Conference* (Oct. 25-26, Phoenix, AZ). IEEE Computer Society, Washington, DC.

VOAS, J. M. 2000a. Limited software warranties. In *Proceedings of the 7th IEEE International Conference and Workshop on the Engineering of Computer Based Systems* (Apr. 3-7, Edinburgh, Scotland). IEEE Computer Society, Washington, DC.

VOAS, J. M. 2000b. Developing a usage-based software certification process. *IEEE Computer 33*, 8 (Aug), 32-37.

WAKID, S. A., KUHN, D. R., AND WALLACE, D. R. 1999. Toward credible IT testing and certification. *IEEE Software 16*, 4 (July/Aug.), 39-47.

WALLACE, D. R. 1999. Software verification & validation's role in product certification. In *Proceedings of the First International Software Assurance Certification Conference* (Feb. 28 - Mar. 2, Chantilly, VA). Reliable Software Technologies, Sterling, VA.

WOHLIN, C. AND REGNELL, B. 1998. Reliability certification of software components. In *Proceedings of the 5th International Conference on Software Reuse* (June 2-5, Victoria, B.C., Canada). IEEE Computer Society, Washington, DC.

WOHLIN, C. AND RUNESON, P. 1994. Certification of software components. *IEEE Transactions on Software Engineering 20*, 6 (June), 494-499.