Data Organization - Comparison for Model Validation in FRTS

Dimosthenis Anagnostopoulos and Mara Nikolaidou
Harokopeio University of Athens
El. Venizelou, 17671, Athens, Greece
Email: dimosthe@hua.gr, mara@hua.gr

KEYWORDS
model validation, simulation methodology**,** faster-than-real-time simulation

ABSTACT

Using faster-than-real-simulation (FRTS) to reach predictions for the near-future imposes that models are thoroughly validated. We discuss essential features of model validation, which is accomplished through comparing system observations with model results, and the relationship between the conditions determining model validity and the model/system data under comparison in order to realize the transition from the conceptual model validation design to its efficient execution in real time. A data organization scheme (data model), data structures for system/model data comparison and algorithms for constructing and accessing these structures are introduced for this objective. Alternative techniques for comparing model results and system observations are also discussed, considering the nature and availability of data. Realization of model validation in a FRTS experiment on a single-queue/multi-server processing system is also presented to exhibit the applicability of the proposed approach.

1. INTRODUCTION

When simulation reaches conclusions for systems behaviour in real time, it is known as real-time simulation. The term *real time*, as it relates to simulation, denotes that advancement of simulation time must occur in the real world time (i.e. not faster or slower). In faster-than-real-time simulation (FRTS), results are delivered earlier than real-time. In this case, we are capable of using system observations and model results to both test model validity and, in case of a valid model, reach predictions for the system future states. The quality of predictions can be ensured based on validation tasks involving only past and the current time points.

A conceptual faster-than-real-time simulation methodology has been introduced in (Anagnostopoulos et. al. 1999), providing a framework for conducting experiments dealing with the complexity and the hard real-time requirements. Model execution in FRTS is depicted in figure 1. Experimentation consists of the following:

1. Monitoring, that is, obtaining and storing system and model data during an auditing interval. *Monitoring variables* are used for maintaining system and model results and realizing the necessary comparisons. Considering that *k* monitoring variables, MV_1-MV_k, are used, the respective values of variable *i* for the system and the model are denoted as $MV_i.r$ and $MV_i.s$.
2. Auditing, that is, examining a) if the system has been modified during the preceding auditing interval (system changes), b) if the model no longer provides a valid representation of the system (deviations), and c) if predictions should be used in plan scheduling. Evidently, if conditions (a) or (b) are fulfilled, remodelling is invoked without examining condition (c). Auditing thus realizes an extended model validation process.

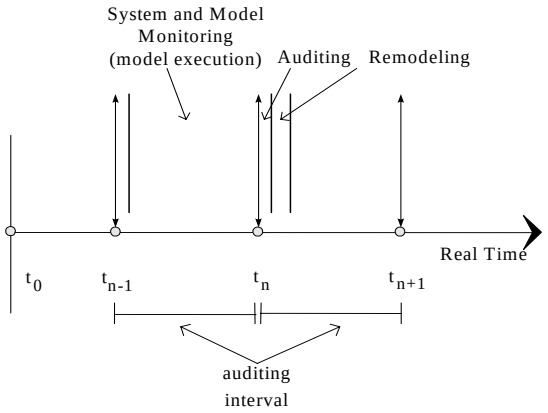

Figure 1: FRTS activities

Validation is the process of determining whether a simulation model is an accurate representation of the system for the particular objectives of the study (Balci 1997). In FRTS, we have the unique capability to use system observations and model results both to test model validity and – in case of a valid model – to estimate future states of the system with simulation predictions. This is based on the simple assumption that, if model validity can be consecutively ensured up to the current real-time point, it would be most probable that simulation predictions are also valid. A methodological approach for model validation in FRTS bas been presented in (Anagnostopoulos 2002), aimed at increasing the level of confidence for simulation

[1]This research was supported by Pythagoras program (MIS 89198) co-funded by the Greek Government (25%) and the European Union (75%).

predictions concerning the time-dynamic system under study. To achieve FRTS validation, system observations and model data must be compared in real time using a computationally efficient process.

As no relevant approaches exist in the literature, the paper contribution is to propose an approach dealing with the following validation process design and implementation issues:

1. Accomplishing validation as a real-time, automated process, consuming low time overhead.
2. Determining the nature of validation data and deal with the complexity encountered, as the number and type of the data compared may be constantly changing. We establish a formal data organization scheme (data model) for this objective, based on the E-R and relational models.
3. Implementing the transition from the validation process conceptual design to efficient real-time execution. Data structures for system/model data comparison and algorithms for constructing and accessing these structures are thus introduced.

In section 2, we review the essential requirements for accomplishing model validation in FRTS, emphasizing the transition from remodelling conditions to the actual comparisons between monitoring variables. In section 3, the *auditing tree* structure is described, which is constructed and accessed to realize monitoring variable comparison. In section 4, a data organization scheme (data model) for validation data and appropriate algorithms for implementing validation are proposed. In section 5, appropriate comparison techniques are discussed for various types of comparisons in FRTS, depending on the nature and availability of data. Conclusions reside in section 6.

2. MODEL VALIDATION REQUIREMENTS IN FRTS

Numerous model validation types and techniques have been discussed in the literature (Balci 1997). In FRTS, both *conceptual model validation* and *computerized model verification* should be pre-assured, as only preconstructed models can be used when modifying a composite model in real time. *Operational validity* is concerned with determining that the model output behaviour has the accuracy required for the intended purpose over the domain of its intended applicability (Sargent 2000). This is where most of the validation testing and evaluation must take place in FRTS. To compare model results and real-time process observations, the following essential features must be provided:

1. Validation must be a real-time, computationally efficient procedure.
2. Conditions examined in model validation, potentially causing remodelling (thus denoted as remodelling conditions), must be explicit. Each remodelling condition involves one or many comparisons between specific -primitive or statistical- variables, i.e. the monitoring variables. The number of monitoring variables corresponding to a single condition depends on the current system configuration (Anagnostopoulos 2002). For instance, comparing a single-queue, multi-server system with its G/G/s model representation may involve the average service time of each server. As the number of servers (s) may be modified (e.g. whenever a server is activated), the number of monitoring variables corresponding to this specific condition may be also modified.
3. A single comparison method is appropriate for each specific monitoring variable comparison. For single-valued variables (e.g. operation parameters), a comparison between model and system values is required. Multi-valued statistical results are compared using other approaches, such as the inspection approach, confidence intervals and hypothesis tests. The acceptable comparison parameters (or deviation ranges) must also be determined for each monitoring variable comparison. Comparison techniques are further discussed in section 5.
4. Remodelling conditions must be characterized as either of AND type (many conditions of this type need to be fulfilled to cause remodelling) or of OR type (any such condition causes remodelling, when fulfilled) (Anagnostopoulos 2002).
5. Validation data (i.e. control data determining how validation is to be performed) must be appropriately organized and a relationship between remodelling conditions, monitoring variables and monitoring variable comparisons must be established.
6. Discriminating among the conditions that do not autonomously cause remodelling (i.e. AND conditions) is required. As conditions are not equally significant, a *weight* factor must be assigned to each AND condition (Anagnostopoulos 2002). Weight factors determine the significance of each condition.
7. A *global scoring algorithm* is required for determining if remodelling must be performed, accessing all AND comparison weights. Weights (or scores) are determined subjectively when conducting various aspects of the validation process and then combined to determine an overall score for the model. The model is considered as valid when this score is higher (or lower) than a threshold (Sargent 2000). Scoring models have been used extensively for model validation purposes (Balci 1989).
8. A formal algorithm and appropriate data structures must be introduced for realizing comparisons. These are discussed in section 3 and section 4.

3. THE AUDITING TREE STRUCTURE

Monitoring variable comparison can be realized using the *auditing tree* (Anagnostopoulos 2002), which is a conceptual tree structure (that is, it does not follow the formal definition of a tree). It is divided into two subtrees and includes two corresponding types of end nodes, *OR* and *AND*, as depicted in figure 2. Each node corresponds to a single monitoring variable comparison. End nodes of type *OR* represent comparisons that autonomously - if fulfilled - cause remodelling. Nodes of type *AND* are aggregately evaluated to determine if remodelling is required. End nodes are directly accessed from *Root*. There are a_1 OR nodes and a_2 AND nodes, all of which are created as children of *Root*. In this way, a_1+a_2 total accesses are required for all nodes. End nodes are created and inserted in the appropriate subtree

whenever the auditing tree is formed (i.e. during auditing). Thus, the number of tree nodes may be variable, corresponding to the number of comparisons to be accomplished. As previously mentioned, each end node is formed for realizing a comparison corresponding to a single remodelling condition. However, a single condition may be expressed via more than one end node.

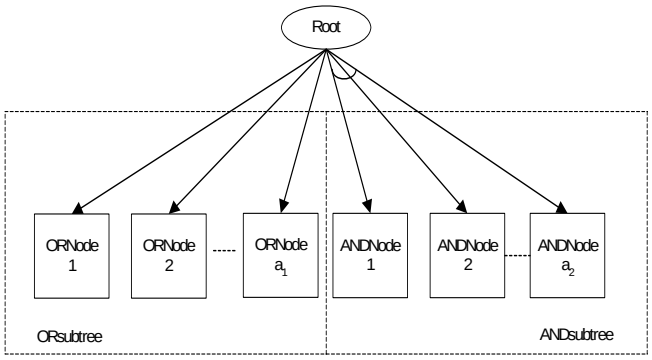

Figure 2: Auditing tree structure

Accessing all nodes, we ensure that all remodelling conditions are evaluated prior to the initiation of remodelling and all reformations/deviations are detected, so that appropriate remodelling actions can be considered. Upon completion of auditing, end nodes are removed. The auditing algorithm concludes that the model is invalid if at least one of the a_1 *OR* node conditions or the aggregate evaluation of the a_2 *AND* node conditions are fulfilled, that is:

$$(C_1=TRUE\ OR\ C_2=TRUE\ \ldots\ OR\ C_{a_1}=TRUE\ OR$$
$$evaluation\ (C_i, C_{ii}, \ldots C_{a_2}) = TRUE),\ \text{where}$$
$$C_1, C_2, .., C_{a_1}\ \text{are } OR \text{ nodes and}$$
$$C_i, C_{ii}, .., C_{a_2}\ \text{are } AND \text{ nodes}$$

The auditing tree provides the following capabilities for realising auditing:

1. Consistent realisation of the aggregate comparison process, dealing with the complexity imposed by the various comparison data/techniques.
2. Assigning priorities to specific comparisons through the AND/OR distinction, thus enabling alternative (optimised) algorithms to be effective, i.e. search the auditing tree for a single condition that may be fulfilled, and then invoke remodelling without accessing the overall tree structure. Such a search would cost considerably less than a_1+a_2.
3. Ensuring that $k=a_1+a_2$ direct accesses to an auditing node single structure will be required when all comparison results need to be evaluated (worst-case scenario).

4. VALIDATION DATA ORGANIZATION

To ensure that validation is executed as a real-time, automated process, and determine an efficient organization scheme for validation process data, we introduce a data organization scheme for remodelling conditions, monitoring variables and monitoring variable comparisons, all of which are considered as discrete entities. To represent each comparison as a discrete node, we are based on the following conclusions:

1. There can be no identical nodes in the two subtrees.
2. As any remodelling condition involves one or many monitoring variables (Section 2), all monitoring variables of a single condition have the same type (AND, OR).
3. Considering that a single condition has a specific degree of significance, all variables of an AND remodelling condition have the same weight.
4. There is a single appropriate comparison method for each monitoring variable (section 2) and, thus, specific comparison parameters for each comparison. However, the appropriate comparison techniques may be different for variables of the same remodelling conditions, depending on the amount and nature of the available system and model data. The same applies for comparison parameters.
5. The number r_i of monitoring variables corresponding to a single condition *i* can be different whenever auditing is executed, depending on the current system/model configuration.

Data Representation

We use the E-R model for establishing a formal data model, depicted in figure 3, based on the following conclusions:

1. Any remodelling condition involves one or many monitoring variables, which are not used by any other condition (1:N relationship).
2. Each monitoring variable corresponds to a single comparison method, i.e. we consider that there is a single, case-specific, optimal comparison technique for each monitoring variable, but not for all variables of the same condition. This is due to the fact that a different amount of either system observations or model data may be available for monitoring variables of the same condition.
3. There is a given set of comparison methods, each being appropriate for specific comparison characteristics (N:1 relationship).

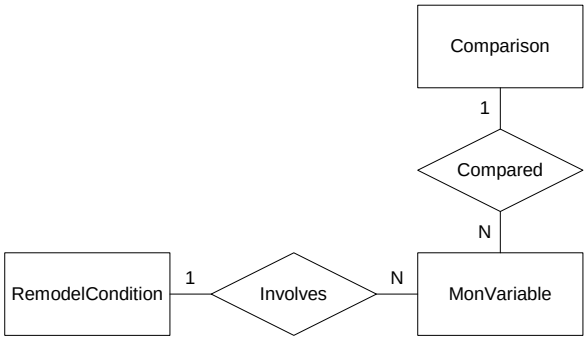

Figure 3: Remodelling condition/monitoring variable/ comparison technique relationship (low-level description)

Realizing the low-level E-R model (figure 3), each remodelling condition is ultimately maintained as follows:

RemodelCondition (<u>rcname</u>, rctype, weight)

where *rcname* is unique for each condition, while *rctype* holds the its type, set to either AND or OR. Note that, according to the relational model, attribute *weight* should not be part of this relation, as it is functionally dependent of

attribute *rctype*, as conditions of type OR have a null weight; however, we present this simplified approach, as this point can be easily handled. Monitoring variables are maintained as:

MonVariable (mvid, mvname, rcname, comptype, comp_params)

where both *mvid* and *mvname* are unique for each monitoring variable, holding the variable id and name, respectively. Attribute *rcname* holds the name of the corresponding remodelling condition, while *comptype* holds the comparison type that is appropriate for this specific variable, given a specific set of comparison methods. Attribute *comp_params* holds the comparison parameters for this specific variable (e.g. the acceptable deviation range, as discussed in section 5). Comparison types are maintained as:

Comparison (comptype)

All three aforementioned relations are derived from the E-R diagram of figure 3.

Forming and Accessing the Auditing Tree

Considering there are r conditions causing remodelling (r_o of type OR and r_a of type AND, $r=r_o+r_a$) and k monitoring variables, k comparisons are made. If condition i involves r_i monitoring variables, remodelling decision is based on $k=a_1+a_2$ ($a_1=\sum_{i=1}^{r_o} r_i$, $a_2=\sum_{i=r_o+1}^{r_o+r_a} r_i$) accesses to OR/ AND comparison results, respectively. Representing each comparison as a separate node, the structure of each auditing node includes identification attributes (i.e. condition name and variable name), system and model results, comparison parameters and the weight attribute (only for AND comparisons).

(rcname, mvname, $MV_i.r$, $MV_i.s$, comptype, comp_params, {weight})

This structure extends the initial node structure introduced in (Anagnostopoulos 2002), in order to accommodate comparison types. Using the above relational structures for maintaining validation data, the algorithm that constructs the auditing tree nodes is depicted in figure 4. Each node includes two additional fields for storing system observation data and model results (i.e. $MV_i.r$, $MV_i.s$). Nodes are inserted in the appropriate subtree as direct descendants of *Root*.

```
(OR nodes}
create  new_OR_node as
select  RemodelCondition.rcname, mvname, comptype,
comp_params
from    RemodelCondition, MonVariable
where
RemodelCondition.rcname = MonVariable.rcname and
RemodelCondition.rctype = 'OR'

{AND nodes}
create  new_AND_node as
select  RemodelCondition.rcname, mvname, comptype,
comp_params, weight
from    RemodelCondition, MonVariable
where
RemodelCondition.rcname = MonVariable.rcname and
RemodelCondition.rctype = 'AND'
```

Figure 4: Construction of auditing tree nodes (AND, OR)

A code fragment for the implementation of the extended node structure as object classes (in Modsim III) is depicted in Figure 5.

```
ORNode = OBJECT;
rcname: STRING;
comptype: STRING;
mvname: STRING;
comp_params: REAL;
systemvalue: ARRAY[INTEGER] OF REAL;
systemvaluenum: INTEGER
modelvalue: ARRAY[INTEGER] OF REAL;
modelvaluenum: INTEGER;
END OBJECT;
ANDNode = OBJECT;
rcname: STRING;
comptype: STRING;
mvname: STRING;
comp_params: REAL;
weight: REAL;
systemvalue: ARRAY[INTEGER] OF REAL;
systemvaluenum: INTEGER
modelvalue: ARRAY[INTEGER] OF REAL;
modelvaluenum: INTEGER;
END OBJECT;
```

Figure 5: Node structure implementation

A sample auditing algorithm implementation (in MODSIM III) is thus depicted in Figure 6.

```
FOREACH Node IN ORsubtree
IF Deviates(systemvalue,systemvaluenum,
modelvalue,modelvaluenum,comptype,comp_params)
    Remodelling(rcname, mvname);
END IF;
END FOREACH;

FOREACH Node IN ANDsubtree
IF Deviates(systemvalue,systemvaluenum, modelvalue,
modelvaluenum,comptype,comp_params)
    CalcWeight (TotalWeight, weight);
    BuildRemodelCondition (RemodelCondition,
rcname, mvname);
END IF;
END FOREACH;

IF TotalWeight> Threshold
    Remodelling (RemodelCondition);
END IF;
```

Figure 6: Auditing algorithm implementation

To conclude, according to the proposed approach, model validation is accomplished in real time comprising the following steps (real-time activities are explicitly stated):

1. Determine remodelling conditions, monitoring variables, comparison techniques and comparison parameters for the purposes of a particular experiment, and maintain all relevant data.
2. Construct the auditing tree nodes, as described in figure 4 (RT).
3. Assign system observations and model result data to the appropriate auditing node fields (RT).
4. Access the auditing tree to determine model validity (RT).

5. COMPARISON TECHNIQUES

Numerous comparison techniques have been established for testing model validity based on system observations and model output data. We employ the widely known classification of statistical techniques proposed by Law and Kelton (Law and Kelton 2000) due to its generality, and

discuss how these techniques can be efficiently employed in FRTS. We propose the following three techniques as most appropriate for realizing monitoring variable comparison:

1. System - model value comparison, for single-valued variables.
2. Inspection approach, for statistical variables with one system observation data set and *n* model result data sets (Law and Kelton 2000). In FRTS, it is evident that only a single system data set will be available in almost all cases, as system observations are produced within a single auditing interval.
3. Confidence interval approach, for statistical variables with *m* system observation data sets and *n* model data sets) (Law and Kelton 2000). We suggest the classical approach proposed by Welch for building a confidence interval based on a different number of independent data sets (Welch 1938), as other approaches are more restrictive, such as the paired-t approach (Law and Kelton 2000), imposing that $n=m$, which can be only rarely ensured.

Comparison parameters define the acceptable deviation range (*dr*) for each comparison, so that, when model value deviation from the corresponding system value exceeds the predetermined range, invalidity is detected. However, the meaning of this range is different for each comparison technique. Each deviation range depends on the nature of the experiment (i.e. how close should model states be to system states) and the specific technique used to compare system observations and model data. In the following, we discuss the meaning of comparison parameters and how data comparison is performed in each of the three above cases. In cases (1) and (2), deviation range determines the lower and upper endpoints of the interval $[l(MV_i.r), u(MV_i.r)]$ and the model is considered as valid when:

$$MV_i.s \in [l(MV_i.r), u(MV_i.r)],$$

$$l(MV_i.r) = MV_i.r(1-dr),\ u(MV_i.r) = MV_i.r(1+dr)$$

For single-valued variables, system and model variables ($MV_i.s$, $MV_i.r$) are directly obtained. For statistical variables with a single system observation data set and *n* model data sets,

$$MV_i.s = sum\ (MV_{i1}.s, MV_{i2}.s, \ldots, MV_{in}.s)/n,$$

where $MV_{ij}.s$ is the statistical sample obtained from replication *j* when *n* replications are made (i.e. in the case of terminating simulations).

In the third case, where statistical variables with *m* system observation data sets and *n* model data sets are available, we build a confidence interval based on a different number of independent data sets (Welch 1938). According to the Welch approach,

$$\overline{MV_i.r} = \frac{\sum_{j=1}^{m} MV_{ij}.r}{m},\quad \overline{MV_i.s} = \frac{\sum_{j=1}^{n} MV_{ij}.s}{n}$$

$$S^2(MV_i.r) = \frac{\sum_{j=1}^{m} [MV_{ij}.r - \overline{MV_i.r}]^2}{m-1}$$

$$S^2(MV_i.s) = \frac{\sum_{j=1}^{n} [MV_{ij}.s - \overline{MV_i.s}]^2}{n-1}$$

The estimated degrees of freedom are computed as

$$f = \frac{[S^2(MV_i.r)/m + S^2(MV_i.s)/n]^2}{[S^2(MV_i.r)/m]^2/(m-1) + [S^2(MV_i.s)/n]^2/(n-1)}$$

The following interval as an approximate *100(1-a)%* confidence interval for $MV_i.r - MV_i.s$

$$\overline{MV_i.r} - \overline{MV_i.s} \pm t_{f,1-a/2}\sqrt{\frac{S^2(MV_i.r)}{m} + \frac{S^2(MV_i.s)}{n}}$$

Evidently, in case (3), the deviation range defines the value *a*, meaning that we wish the confidence interval to cover $MV_i.r - MV_i.s$ with probability *1-a*. Suppose that the upper and lower endpoints of the interval are marked as *u(a)* and *l(a)*, respectively. If $0 \notin [l(a), u(a)]$, the difference between $MV_i.r$ and $MV_i.s$ is statistically significant at level *a* and we consider the model to be invalid.

6. CONCLUSIONS

We explored two aspects of FRTS that have not been widely discussed: data organization and comparison techniques for model validation purposes. The proposed data model can be updated in real time, e.g. to include new monitoring variables conforming to the current system configuration or modify the technique used for comparing specific data, when the amount of available data is changed. This model is also accessed in a consistent way and with low time overhead to form the structures realising data comparison in real-time, enabling validation to execute as a well-defined process that can be adapted to the current FRTS conditions without any manual intervention.

REFERENCES

1. Anagnostopoulos D., M. Nikolaidou, P. Georgiadis. 1999. “A Conceptual Methodology for Conducting Faster-Than-Real-Time Experiments”. *SCS Transactions on Computer Simulation*, vol. 16, no 2.
2. Balci, O. 1997. “Verification, Validation and Accreditation o Simulation Models”. *Proceedings of WSC’97*, IEEE Computer Press.
3. Anagnostopoulos D. 2002. “A Methodological Approach for Model Validation in Faster-than-Real-Time Simulation“. *Simulation Modeling Practice and Theory*, Elsevier, vol. 10, no. 3-4.
4. Sargent, R. 2000. “Verification, Validation and Accreditation of Simulation Models”. *Proceedings of WSC’00*, IEEE Computer Press.
5. Balci, O. 1989. “How to Assess the Acceptability and Credibility of Simulation Results” *Proceedings of WSC’89*, IEEE Computer Press.
6. Law A.M. and W.D. Kelton. 2000. *Simulation Modelling and Analysis*, McGraw Hill.
7. Welch B. L. 1938. “The Significance of the Difference Between Two Means when the Population Variances are Unequal”, Biometrica, vol. 25, pp. 350-362.