The CSI Journal on
Computer Science and Engineering
Vol. 1, No. 4 (b), Winter 2003
Pages 19-29
Regular Paper

OAV-VVT Expert, an Active System for Verification and Validation of Knowledge Base Using OAV Knowledge Representation

Chitra Dadkhah Ahmad Abdollahzadeh Barforosh

Computer Engineering Department, Amirkhabir University of Technology
Tehran, Iran

Abstract

In this paper, we present an active expert system (OAV-VVT expert) to verify and validate the knowledge base (KB). Our work is based on new approach to representation and reasoning techniques, E_x-OAV KB, based on Object-Attribute-Value (OAV) knowledge representation. All common issues of verification and validation are considered in OAV-VVT expert. OAV-VVT expert can be either used during the building of knowledge base or refinement of existing KB. OAV-VVT expert is an expert system with reasoning and explanation mechanisms. It is application and knowledge representation independent. OAV-VVT expert improves the performance of V&V phase in building the knowledge base by checking unreference attribute values and illegal attribute values, during transforming the existing KB to E_x-OAV KB. All necessary algorithms to transform different knowledge representation techniques have been presented in this paper to justify that the E_x-OAV KB is feasible.

Keyword: verification and validation, knowledge base, knowledge representation, OAV triple, expert systems, AI systems

1. Introduction

Verification and Validation (V&V) are the most crucial phases in building knowledge base systems. Researchers and practitioners in the field of AI applications generally agreed that knowledge base of any intelligent systems must be adequately verified and validated. To build an AI system, the quality of KB is crucial. The need for an integrated approach towards V&V of knowledge base is quite obvious and in this paper we present an active OAV-VVT expert to verify and validate the knowledge during the building of knowledge base or refinement of existing KB. In order to build such a V&V expert tool, it is necessary to have a common definition of V&V. Table 1 shows different approaches to the definition of V&V process in knowledge base system. We define "validation" as the correctness and existence of system output according to the test cases that have been defined by expert and "verification" as the checking of the consistency, completeness and logical & semantic contradiction of KB. All of these definitions are presented by logical formal language. Consequently with this definition, OAV-VVT expert will cover all the common issues in V&V of the knowledge. Majority of the previous works in this field are concentrated on the verification of KB and less attention has been paid to the validation. The evaluation of system's output based on expert opinion is very complex and researchers have limitedly dealt with the issues of completeness, consistency or correctness in certain KB systems as expert system [5, 8, 21, 23, 30, 32].

We categorize the researches work in the field of V&V of the KB as below:

1. Representation of a new method for description of knowledge [27, 29, 34].

Table 1. The definitions of V&V by the researchers and practitioners

References	Validation	Verification
BOEHM [1]	Building the right system.	Building the system right.
IGNIZIO [2]	. Justification for the employment of an expert system. . The verification of the overall performance of the expert system.	The validation of the consistency and completeness of the expert system's rule base.
MENSHOEL & DELAB [3]	Addresses software system's usefulness with respect to some real-world task, regardless of its specification.	Concerns a software system's conformance to its specification.
KANDEL & SMITH [4]	A test of whether the Expert System matches the design ideas, i.e., whether it matches the technical requirements and expectations.	Examine the technical aspects of an Expert System in order to determine whether the Expert System is built correctly.
WU & al [5]	The correctness of inference without considering the Input/Output.	The correctness of systems outputs according to specific Inputs.
ROUSSET & al [6]	The correctness of system's output depends on Test cases and Initial valid Fact Base.	Examining the logical and semantic contradiction of Rule Base.
TSAI &al [7]	The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirement.	1) The process of evaluating a system or component to determine whether the products of the given phase satisfy the conditions imposed at the start of that phase. 2) Formal proof of program correctness.
KNAUF & al [8]	Asks whether or not a system is considered to be the required one, something that somehow lies in the eyes of the beholder.	Provides a firm basis for the question of whether or not a system meets its specifications.

2. Representation of a new algorithm to process [24], [28].
3. Development or extension of an existing method of process [26, 28].
4. Design a new structure for construction of the KB to simplify the V&V techniques [22, 27, 29].
5. Definition of new issues for V&V of the KB and its related solution, apart from completeness, consistency or correctness [18, 31].

All the previous research works carried out in this field are included in [38].

In OAV-VVT expert, we introduce E_x-OAV KB based on new approach to representation and reasoning techniques to create new KB from the existing KB. E_x-OAV KB is based on Object-Attribute-Value(OAV) Knowledge representation. We will present and demonstrate how the major techniques of knowledge representation (i.e. semantic network, frames, production rules) could be transformed to E_x-OAV KB. In our work the syntax of knowledge is changed but the semantic of knowledge has been remained unchanged. E_x-OAV KB will be verified and validated by OAV-VVT expert and this process is iterative and all the invalid knowledge will be recorded and then modified. The process of modification of invalid knowledge in OAV-VVT expert will be done by using the existing V&V methods [18], [35, 36, 40, 41] or knowledge engineer. This process will be completed when not new invalid knowledge is discovered in E_x-OAV KB and the process will start as soon

as the new knowledge submits to the E_x-OAV KB. The knowledge flow in OAV-VVT expert is shown in Figure 1.
OAV-VVT expert by using the explanation component, which is an essential part in any expert system, can explain "how" an invalid Knowledge has been discovered and "why" it is invalid. All common issues of verification and validation (V&V) such as consistency, completeness, logical & semantic contradiction and correctness of knowledge base are considered in OAV-VVT expert.

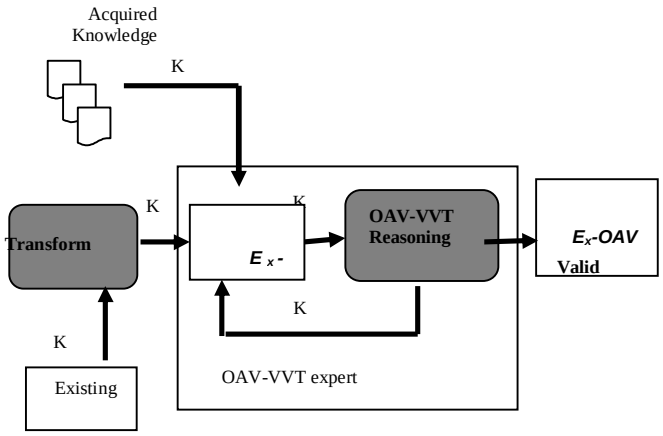


Figure 1. The knowledge flow in OAV-VVT expert

We present the design and implementation of the OAV-VVT expert. OAV-VVT expert is a tool to V&V of knowledge during the submitting a new knowledge or refinement of the existing knowledge base.
The structure of this paper is as follows:

- Section 2 describes the structure of E_x-OAV KB and present procedures to transform the major knowledge representation techniques to E_x-OAV KB.
- Section 3 defines the issues of V&V in a formal specification language.
- Section 4 presents the architecture and reasoning technique of OAV-VVT expert.
- Section 5 reports the implementation phase and describes detail information about OAV-VVT expert.
- Section 6 presents the evaluation of OAV-VVT expert.
- Section 7 will present the conclusion of this work.

2. The Structure of E_x-OAV KB

Object – Attribute – Value (OAV) provides a particularly convenient way in which to represent certain facts and heuristic rules within the KB. Each OAV triple is present with specific entity or object and a set of attributes with their values associated to every object. [2]
All knowledge in Ex - OAV KB is Rule Base System using OAV triple. The structure of E_X -OAV KB is shown in Figure 2.
In the next section we present the process of transforming the major knowledge representation techniques to E_X -OAV KB.

<Rule-name>: IF <premise> Then <conclusion>
<Premise>: <Object> . <Attribute> <Operator> <Value>| AND <Premise>
<Conclusion>: <Object> . <Attribute> <Operator> <Value>
<Object>: string
<Attribute>: string | Boolean | real
<Operator>: < | > | = | ≠
<Value>: real | string | TRUE | ?Attribute | @Attribute

Figure 2. The structure of the rules in E_X -OAV KB

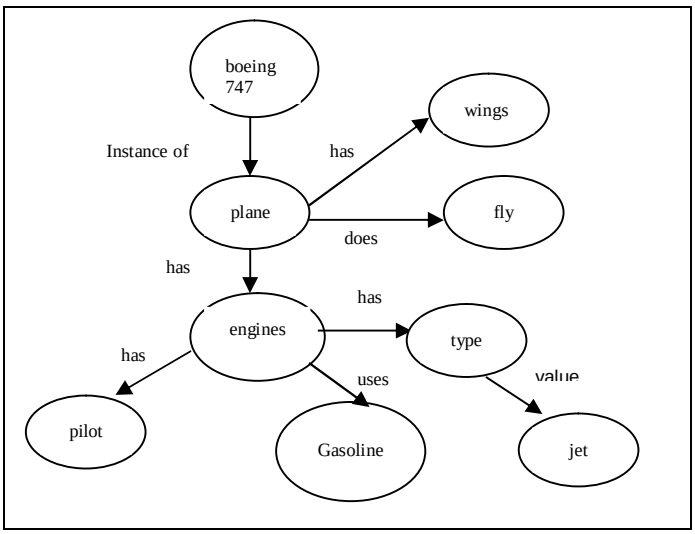
To justify the algorithms, we used the classical examples from following references [2, 32, 33, 37].

2.1 Transforming algorithms

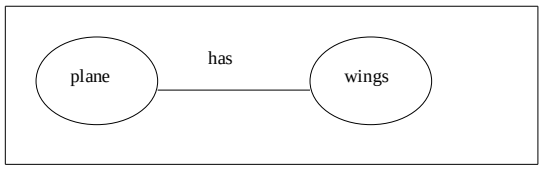
OAV-VVT expert verifies and validates the E_x-OAV KB. The existing KB should transform to E_x-OAV KB and in this section, we introduce the algorithms for transforming the major techniques of knowledge representation to E_x-OAV KB.

2.1.1 Semantic network to E_X -OAV KB

A semantic network might be seen as a network that is composed of multiple OAV triple in network as illustrated in Example 1 [9]. In this paper, we only consider the "has" relationship between nodes and demonstrate how this relationship transforms to the OAV triple. The other relations described in the technical report [10]. The nodes of semantic network could be one of the following types: object, attribute or value. Example 2 shows the subset of the above semantic network that contains only two nodes with a "has" relationship. We assume the "plane" node as a starting point and will be defined as a "beginnode" and the other node "wings" will be introduced as an "endnode". We conclude that the "beginnode" is an object and the "endnode" is an attribute. The "has" relationship serves as the equal operator between object – attribute and its value, which matches the structure of E_X -OAV KB, as shown in Figure 3. The "?" sign indicates that the value of attribute is variable and with this format, it is possible to assign a value and add a new knowledge in the E_x-OAV KB.



Example 1. The semantic network of plane's characteristic



Example 2. The semantic network with the HAS relationship

plane . wings = ?wing

Figure 3. The OAV triple associated to Example 2

Figure 4 illustrates the procedures for transforming the "has" relation part of semantic network to E_x-OAV KB.

Function has-relation (beginnode, endnode) **returns** OAV-knowledge
Input : beginnode, the node sending the arc
endnode, the node receiving the arc
Output: OAV-knowledge , object-attribute-value associated to subset of Snet.
Object ← beginnode
Attribute ← endnode
Operator ← " = "
Value ← "?" Attribute
OAV-knowledge ← <Object> . <Attribute> <Operator> <Value>
Return OAV-knowledge
end

Figure 4. The procedure for transforming the "has" relation to OAV triple

Figure 5 shows the final result of transforming the KB of example 1 to OAV triple.

Plane.wings = ?wing
Plane.engines = ?engine
Engine.pilot = ?pilot
Engine.type = jet
Plane.does = ?fly
Engine.uses = ?gasoline
Boeing747.wings =?wing
Boeing747.engine =?engine
Boeing747.does =?fly

Figure 5. OAV triple associated to Example 1

2.1.2 Production rules to E_X -OAV KB

Rule-based technique of knowledge representation is known as production rule in literature [2]. Such rules are typically of the IF-THEN format. The Example 3 shows an example of the Rule-based system.

The premise and conclusion of rules contain attributes and values (AV) associated to an object either implicit or explicit as below:

Rule 1: If student.GREscore >= 1350
Then addmision.status = yes
Rule 2: If GPA >= 3.5
Then accept = into honor society

We can simply define an OAV triple for each premise and conclusion and when there is an implicit object we use the system KB for finding the associated object to AV pair.

The Figure 6 presents " rbs-to -oav " procedure for transforming the rule into an OAV triple, which will be active in event of receiving new rule. The function "Find-obj" in this procedure searches in the KB systems for finding the object related to AV pairs. The OAV triple associated to the Example 3 are shown in Figure 7.

Rule 1: If the student's GRE score is 1350 or more
Then admit the student to the graduate program.
Rule 2: If grade point average equals or exceeds 3.5
Then accept into honor society.

Example 3. The rule-based system

Function rbs-to-oav (obj,att,oper, val, OAV-knowledge)
Input: obj, the object of clause or the "implicit"
att, The attribute of clause
oper, The operator of clause
val, The value of clause
Output: OAV-knowledge, The OAV triple associated to clause
If obj = "implicit" **Then**
Begin
Find-obj (obj,att,val,expobj)
Object ← expobj
End
Else Object ← obj
Attribute ← attr
Operator ← oper
Value ← val
OAV-knowledge ← <object> . <Attribute> <operatot> <value>
Return OAV-knowledge
End

Figure 6. The procedure for transforming the rule to OAV triple

Student . GREscore >= 1350
Student . admissionstatus = yes
Student . GPA >= 3.5
Student . accept = into honor society

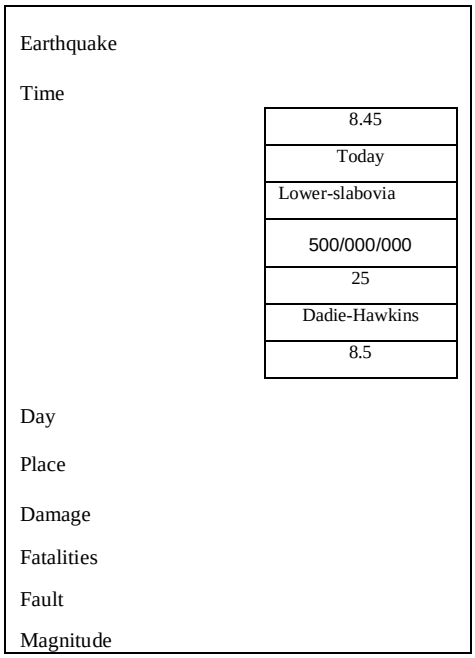
Figure 7. The OAV triple associated to Example 3

2.1.3 Frame to E_X -OAV KB

The Frames are a robust way of presenting knowledge. A frame presents all the available fact as an object plus slots and value for all the information related to the objects. A frame has a name, slots with labels describing the concept's major attributes or features and possible values for each attributes [2,37]. The Example 4 illustrates a frame-based knowledge representation (FB KR) for the earthquake.

We consider all the slots described by a frame as the object's attributes and its values and the frame name serves as the object, their slots serve as a list of attributes common to that object and the value of each slot is considered as the attribute value. Figure 8 shows the procedure for transforming FB KR to E_X-OAV KB. Figure 9 illustrates the OAV triple associated to the Example 4.

A frame may have attached procedures that capture procedural information about the concept. A facet can be used to define constraints on the property value or to execute procedure for obtaining the property value or what to do if the value changes.



Example 4. A frame-based knowledge representation for Earthquake

```
Function fb-to-oav (Framename, {(slot,slotvalue)} return {OAV-Knowledge}
        Input: Framename, The name of frame
               {(slot,slotvalue)}, the list of slots and their values
        Output: {OAV-knowledge}, The list of OAV triple associated to Frame
   Object ← Framename
For each (s,v) in {(slot,slotvalue)} do
        Attribute ← s
        Operator ← "="
        Value ← v
        OAV-Knowledge ← <Object> . <Attribute> <operator> <value>
Add OAV-knowledge to {OAV-knowledge}
end
Return {OAV-knowledge}
end
```

Figure 8. The procedure for transforming the FB KR to E_X -OAV KB

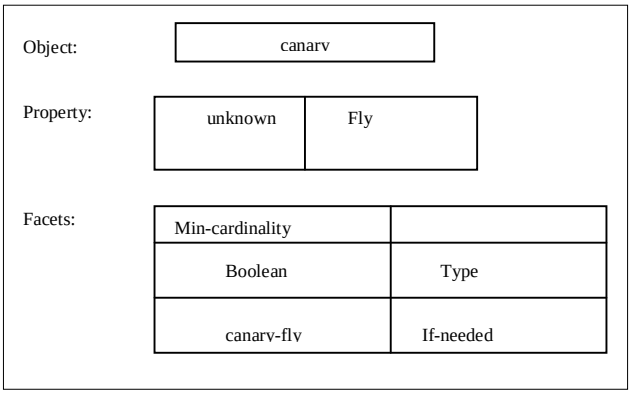
```
Earthquake.Time = 8.45
Earthquake.Day = Today
Earthquake.place = Lower-slabovia
Eathquake.Damage = 500/000/000
Earthquake.Fatalities = 25
Earthquake.Fault = Dadie-Hawkins
Earthquake.Magnitude = 8.5
```

Figure 9. The OAV triple related to the Example 4

The "if-needed" and "if-changed" facets are important features of frame-based systems. An "if-changed" facet, like the "if-needed" facet, executes some method (canary-fly) as shown in the Figure 10. In general the method attached to an object designs in a Rule-Base system and will be executed whenever requested.

The method "canary-fly" defines that a canary can fly if it has a minimum of two wings.

Figure 11 shows the procedure for transforming "if-needed" or "If-changed" facets to OAV triple. The "facet-to-oav" procedure call another procedure ,"met-to-oav", to transform the method to OAV triple (figure 12).



Example 5. The frame-based for the canary object

The "@" sign means that a method must be executed to assign or change the value of the associated object attribute. By calling the "rbs-to-oav" function, we transform the method to OAV triple and it will be recorded in OAV-VVT KB. Figure 13 shows the OAV triple related to the Example 5. OAV-VVT expert infers the new knowledge from OAV-VVT KB and assign or change the value of property during the reasoning. For example, if we have a fact like "canary.wing = 2" then OAV-VVT expert assigns the "True" value to "fly" attribute of "canary" object by reasoning strategy.

```
If Canary.wings < 2
Then Canary.fly = False
If Canary.wings = 2
Then Canary.fly = True
```

Figure 10. The "canary-fly" method

```
Function facet-to-oav (Objectname, {(property,{facet}}
        return {OAV-Knowledge}
        Input: Objectname, The name of object
               {(Property,{facet}}, the list of facets for
               property as {(Attr,{(s1,v1),(s2,v2),…}}
        Output: {OAV-knowledge}, The list of OAV triple associated
                to Frame
Object ← Objectname
For each (p,{facet}) In {property,{facet}} do
    property ← p
    For each (s, v) In {facet} do
    Attribute ← s
    Operator ← "="
    Case (s),
    "Type" :        Value ← ?v
    "if-needed" or "if-changed" :
        Begin
          Attribute ← property
          Value ← @ v
          met-to-oav(v)
          end
          otherwise:    Value ← v
          OAV-Knowledge ← <Object> .<Attribute>
          <operator> <value>
          Add OAV-knowledge to {OAV-knowledge}
      end
```

Figure 11. The procedure for transforming the facets to OAV triple

Function met-to-oav (object,property,metname, {ifpart},{thenpart}) **return** ({if-OAV-Knowledge} , {then-OAV-knowledge})
Input: Metname, The name of method
Object, the name of object
Property, the name of property
{ifpart}, The list of attribute value pair of premises
{thenpart}, The list of attribute value pair of conclusion
Output: {if-OAV-knowledge}, The list of OAV triple associated to premises {then-OAV-knowledge}, The list of OAV triple associated to conclusion
For each (A,O,V) **in** {ifpart} **do**
rbs-to-oav(object,A,O,V,OAV-knowledge)
Add OAV-knowledge **to** {if-OAV-knowledge}
end
For each (A1,O1,V1) **in** {thenpart} **do**
rbs-to-oav(object,property,O1,V1,{OAV-knowledge})
Add OAV-knowledge **to** {then-OAV-knowledge}
End
Add ({if-OAV-knowledge},{then-OAV-knowledge}) **TO** E_x-OAV KB
Return ({if-OAV-knowledge} ,{then-OAV-knowledge})
end

Figure 12. The procedure for transforming the method to OAV triple

Canary.min-cardinality = 2
Canary.type = ?boolean
Canary.fly = @canary-fly

Figure 13. The OAV triple associated to the Example 5

3. The Issues of V& V

The V&V activities which has been addressed in OAV-VVT expert are the following steps: [11, 12, 13, 14, 15, 16, 17].

- Completeness checking.
- Consistency checking.
- Logical & semantic contradiction checking.
- Validation checking.

In order to be able to define a detail description in OAV-VVT expert, we introduce the abbreviations based on the work of Bouali and Ignizio [18, 33] as shown in Figure 14. It shows the abbreviations in 3 sections. The first, second and third sections are based on the work of Bouali, Ignizio and ours respectively.

Def1: c (R) is an *unachievable intermediate conclusions* iff
{ c(R) ≠ {G}, {∀ S | S ∈ RB , S ≠ R , c (R) ∉ {P(S)}}}

Def2: c (R) is an *unachievable intermediate goal* iff
{c (R) = {G} ,{∀ S , p(R) | S ∈ RB , S ≠ R , p(R) ∈ {P(R)} , p(R) ∉ {C(S)},

Def3: p(R) is an *unachievable premise* iff
{ ∀ S, | S ∈ RB , S ≠ R , p(R) ∉ {C(S)}, p(R) ∉ ask(voc)}

Def4: V is an *unreferenced attribute value* associated to attribute "A" of object "O" if

RB: Rule Base
FB: Fact Base
RBxFB: Inferred knowledge from RB based on FB(mapping RB on FB)
ask(voc): the askable attributes

{P(R)}: the set of premise clauses for rule R.
p(R): one premise clause from the set {P(R)}.
c(R): the conclusion clause for rule R .
{S} = {T}: the set S is equal to the set T.

{G}: Goals of system
{Val (obj / att)}: the set of values for the attribute "att" associated to object "obj" in the premise or conclusion clause of rules.
OA-LST: list of illegal values associated to the attribute "A" related to the object "O".
{TSTCASE}: List of test cases.
Tstcase: One test case from {TSTCASE} in the form of (FB, output)
O.A = V: the V values for A attribute of O object
Ø: Not

Figure 14. The abbreviations used in the OAV-VVT expert

Our definitions of the five types of incompleteness issues are as follows:

{ ∀ V| V ∈ OA-LST, V ∉ {VAL(O/A)}}

Def5: V is an *illegal attribute values* associated to attribute "A" of object "O" if
{ ∃ V | V ∈ {VAL(O / A)}, V ∉ OA-LST}.

Our definitions of consistency issues in the OAV-VVT expert are as follow:

Def6: Rule r is *redundant rule* if
{∃ s | s ∈ RB, {P(r) = P(s)}, c (r) = c(s) }

Def7: it is a *logical contradiction* between rule r,s if
{{P(r) = {P(s)}, (O.A = V1) ∈ c(r) , (O.A = V2) ∈ c(s), V1 = ¬ V2

Def8: Rule r is *subsumed* by rule s if
{c(r) = c(s), {P(s) ⊂ P(r)}}

Def9: A set of rules, r_1, r_2, r_3, …, r_n are *circular rules* if
{$c(r_n)$ ∈ {G} , c (r_{n-1}) ∈ p(r_n), c (r_{n-2}) ∈ p(r_{n-1}), …….. , c (r_1) ∈ p(r_2) , c (r_n)∈ p(r_1)}

Def10: Rule r contains *unnecessary premise clause* if
{∃s | s ∈ RB , c(r) = c(s) , {∃ p1, p2 | p1 ∈ {P(r)} , p1 = {O1.A1= V1), p2 ∈ {P(s)}, p2 = (O1.A1 = V2) , V1 = ¬ V2 , p1 ≠ p2} ,{∀pi | pi ≠ p1, pi ∈ {P(r)}, pi ∈ {P(s)}}

The "∇ " and "* " symbols introduce the logical & semantic contradiction in Def11 and Def12.

Def11: E_x-OAV KB contains a *logical contradiction* iff
$\nabla \in$ (LCxFB).

Def12: E_x-OAV KB contains a *semantic contradiction* iff
$* \in$ (SCxFB).

We define three types of Validation that may be detected by the OAV-VVT expert as shown below:

Def13: E_X-OAV KB has a *Contradiction output* if
{∀tstcase | tstcase ∈ {TSTCASE},
tstcase = (FB1, output), RBxFB1 = ¬ output

Def14: E_X-OAV KB has *any output* if
{∀tstcase | tstcase ∈ {TSTCASE},
{tstcase = (FB1, output), RBxFB1 = { }

Def15: E_X-OAV KB has an *incorrect output* if
{∀tstcase | tstcase ∈ {TSTCASE}, },
tstcase = (FB1,output), RBxFB1 ≠ output

4. The Architecture of OAV-VVT Expert

The OAV-VVT expert contains knowledge base, reasoni and explanation components. The architecture a knowledge flow of OAV-VVT expert is illustrated Figure 15. In this section, first the steps involved to apply t V&V algorithm are presented and then the KB and definition of logical and semantic contradiction are presented. Based on KB and these definitions, the description of reasoning and explanation are followed.

The V&V algorithm of OAV-VVT expert contains some steps as shown below:

1. Define the test cases (facts/ goal) by the expert and the set of LC & SC. Transform the existing KB to the E_x-OAV KB.
2. Completeness checking of E_x-OAV KB.
3. If the OAV-VVT expert finds any invalid knowledge from step 2, knowledge engineer records the invalid knowledge and modifies the E_x-OAV KB accordingly and then go to step 2, otherwise go to step 4.
4. Consistency checking of E_x-OAV KB.
5. If the OAV-VVT expert finds any invalid knowledge from step 4, Knowledge engineer records the invalid knowledge and modifies the E_x-OAV KB and then go to step 2, otherwise go to the step 6.
6. Logical & semantic contradiction checking of E_x-OAV KB.
7. If the OAV-VVT expert finds any invalid knowledge from step 6, Knowledge engineer records the invalid knowledge and modifies the E_x-OAV KB accordingly and then go to step 2, otherwise go to step 8.
8. Validation checking of E_x-OAV KB.
9. If the OAV-VVT expert finds any invalid knowledge from step 8, Knowledge engineer modifies the E_x-OAV KB and then goes to step 2, otherwise go to step 10.
10. Stop.

The KB of OAV-VVT contains following rules:

LC = { LC1 : IF (?O.?A = ?V) & (?O.?A ≠ ?V) THEN ∇
LC2 : IF (?O.?A < ?V) & (?O.?A = ?V) THEN ∇
LC3 : IF (?O.?A > ?V) & (?O.?A = ?V) THEN ∇
LC4 : IF (?O.?A < ?V) & (?O.?A > ?V) THEN ∇
LC5 : IF (?O.?A = ?V1) & (?O.?A = ?V2) & (V1 ≠ V2) THEN ∇
LC6 : IF (?O.?A < ?V1) & (?O.?A = ?V2) & (V1 > V2) THEN ∇
LC7 : IF (?O.?A > ?V1) & (?O.?A = ?V2) & (V1 > V2) THEN ∇
LC8 : IF (?O.?A < ?V1) & (?O.?A > ?V2) & (V1 < V2) THEN ∇ }

It contains the rules for methods associated to facet as explained in the section 2.1.3. We also record the semantic contradiction rule (Rule RC) in KB of OAV-VVT expert.

Rule SC: If man ^ Pregnant
Then *

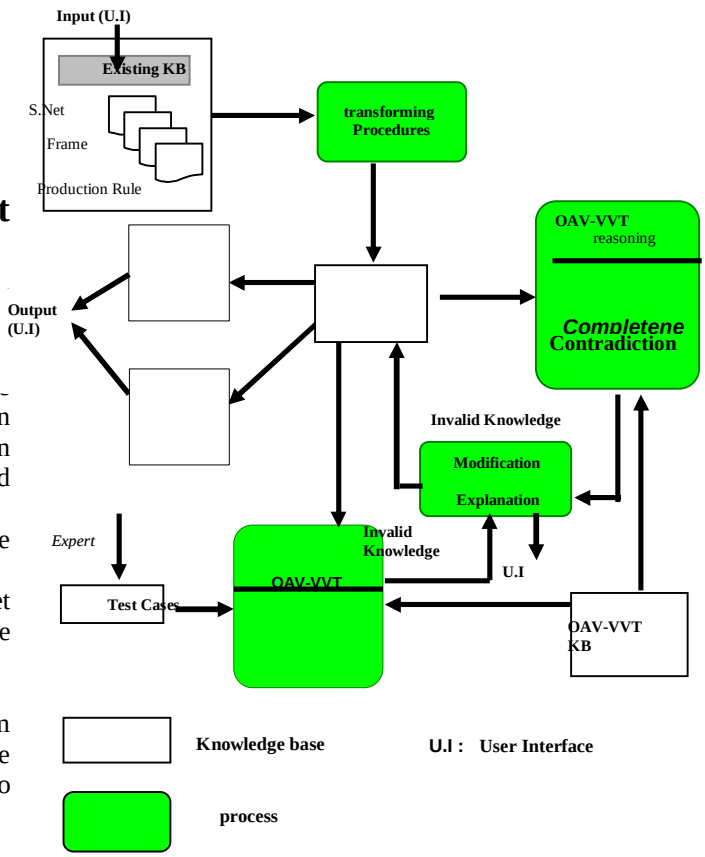


Figure 15. The architecture of OAV-VVT expert

The reasoning of OAV-VVT expert uses the backward strategy in order to discover the invalid knowledge in E_x-OAV KB based on our V&V definition as defined in the section 3. We applied the backward chaining for all goals, which are detailed in application domain, to find all associated facts. All the facts that resulted from goals will be recorded in a list. If these facts do not match the testcases based on Def13, Def14, Def15, then OAV-VVT expert discovers an invalid knowledge. Previous works shows that it is possible to have more than one list for a goal [18, 19] and we will checked all of lists. If the facts of these lists trigger the rules of KB then the system discovers any logical & semantic contradiction based on LC, SC rules.

Whenever the OAV-VVT expert discovers any invalid knowledge, it is able to explain how the system discovers this invalid knowledge and show the trace of discovering it. As the process of V&V in OAV-VVT expert is iterative and dynamic, we consider a threshold (some iteration of V&V

process) for copying the E_x-OAV KB to E_x-OAV KB dual. The application could use the E_x-OAV KB dual any time and the system continues the V&V of E_x-OAV KB. Whenever the algorithm of V&V in OAV-VVT expert stops, we will have an E_x-OAV valid KB.

5. Implementation

The prerequisites, which have been described in OAV-VVT expert, are as follow:

- All the rules of canonical KB (E_X -OAV KB) are backward chaining rules.
- The rule set is deterministic (there is no confidence factor).
- Only conjunctive clauses are employed in the premise and a single clause appears in the conclusion.

We designed a user-friendly tool to transform the semantic network or frame, to the E_x-OAV KB (Figure 16). We implemented this tool with Visual Basic 2000. We could draw and edit the semantic network or frame and construct the KB with this tool. This tool uses the standard algorithms

that have been described in section 2 for transforming the knowledges to OAV triple. The transformed knowledges are shown in three forms: OAV triple, OAV hierarchy and OAV table.

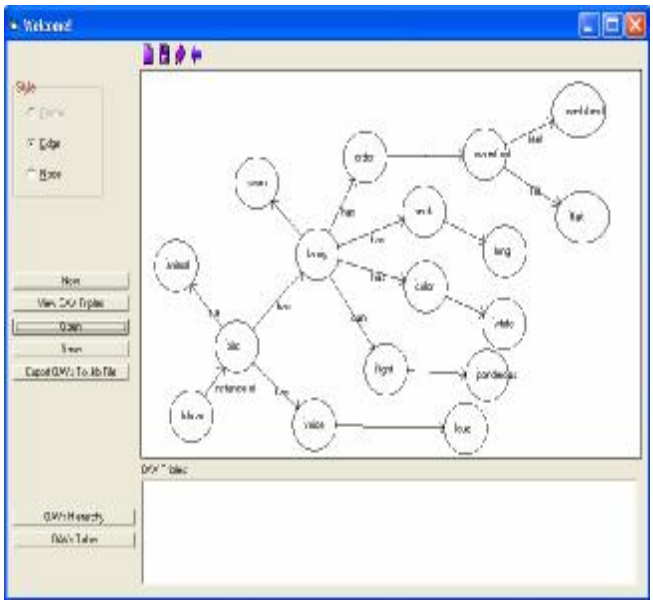


Figure 16. The tool for transforming the KB to E_x-OAV KB

Figure 17 shows the OAV triple associated to the KB of Figure 16. The OAV hierarchy and the OAV table of the constructed KB with this tool are shown in Figure 18 and figure 19.

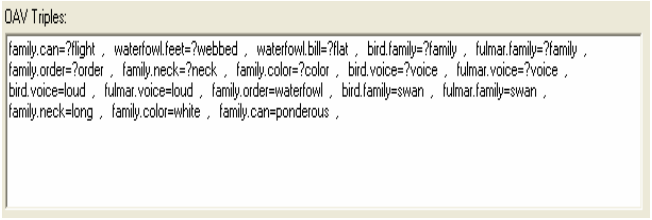


Figure 17. The OAV triple associated to the Figure 16

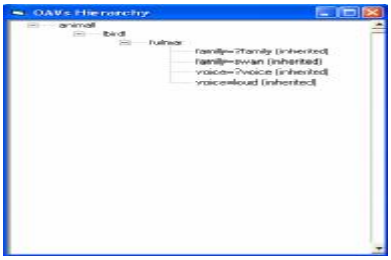


Figure 18. The OAV hierarchy associated to the Figure 16

Figure 19. The OAV table associated to the Figure 16

Finally, the tool will be constructed the knowledge base which its knowledges are represented in OAV triple as shown below:

```
bird(fulmar) :- voice(loud),
                family(swan).
family(swan) :- order(waterfowl),
                neck(long),
                color(white),
                can(ponderous).
order(waterfowl) :- feet(webbed),
                bill(flat).
```

This KB is ready to be used by any expert system shell for reasoning or the verification and validation tools. OAV_VVT expert, could verify and validate the knowledge of this knowledge base as shown below:

You can choose one of the categories of V&V issues.

******V&VTool**********
Choose one of category:
1. Completeness
2. Consistency
3.Logical Contradiction
4. View Rule
5. Edit Rule
6. Exit
****
item(1-6)?1
*********V&VTool**********
Choose one of category:
1. Illegal Value
2. Unreferenced Attribute Value
3. Unachievable Premises

4. Unachievable Intermediate Conclusion
5. Unachievable Goal
6. Return
****
item(1-6)?3
Rule No. 1 has unachievable premise.
Why(y/n)?y
Clause No.1 is not an askable Attribute or in any Then part of Rules

6. Evaluation

The evaluation of V&V tool is based on the evaluation parameters. We are defined some evaluation parameters as shown bellow:

- Kind of KB: the knowledge representation techniques that the tool could verify and validate.
- Issues of V&V: The issues of V&V that the tool could check.
- Speed: The performance of V&V process to discover an invalid knowledge
- V&V process time: When the tool could verify and validate the knowledge?
- V&V process mode: The mode of V&V process can be dynamic or static.
- Stop time: If the V&C process mode is dynamic, when the V&V process will stop?
- Explanation technique: Is the tool able to explain "how" and "why" the invalid knowledge has been discovered?

Table 2 shows the value of each evaluation parameters of OAV-VVT expert tool in a qualitative format.

Table 2. The evaluation parameters for OAV-VVT expert tool

Evaluation Parameters	OAV-VVT Expert
Kind of KB	Semantic network, Rule-base, Frame
Issues of V&V	Completeness, consistency, correctness, semantic & logical contradiction, validation
Speed	It is able to check unreference and illegal attribute value at the transforming the existing KB to E_x-OAV KB to increase the performance.
V&V process time	During the building of knowledge base or refinement of existing knowledge base
V&V process mode	Dynamic
Stop time	The V&V process will be stopped when not new invalid knowledge is discovered in E_x-OAV KB. The system copies the E_x-OAV KB to the dual KB for using the applications after some iteration.
Explanation technique	It can explain "why" and "how" the invalid knowledge has been discovered.

We designed an expert system, EVAL-VVT expert, to evaluate the V&V tools. EVAL-VVT expert ask the value of each evaluation parameters from user for the V&V tools. It

uses forward chaining for reasoning with these informations. The KB of EVAL-VVT expert contains the rules, which have been designed for assigning an evaluation value to V&V tool as shown below:

Rule1: IF KB-type is Frame and
Speed is medium and
Op-period is one cycle and
VV-issue is "3-6" and
Mode is static
THEN eval is 37

EVAL-VVT expert can explain "how" the evaluation value assigns to the V&V tool by explanation component. EVAL-VVT expert plots the results of evaluation process (figure 20) for different V&V tools based on evaluation value. It shows that the CHECK Tool scored a lower value by comparison to the other V&V tools such as: KBDR, OAV-VVT, KRFOCL, VITAL.

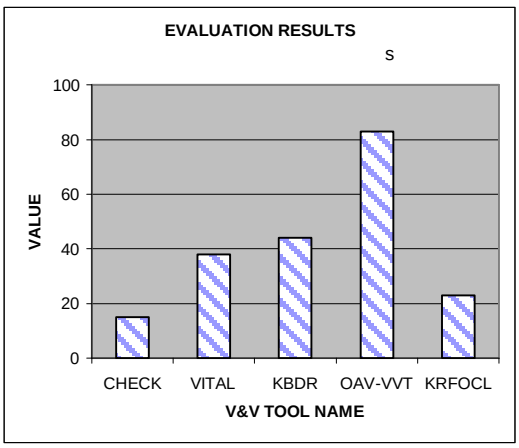


Figure 20. The evaluation chart of V&V tools.

7. Conclusions

This paper addresses the verification and validation of knowledge in KB systems. In previous works less attention has been paid to present a general procedure to cover all

knowledge representation techniques, the major objective of our work was to design an active expert system to V&V all KB based on different knowledge representation techniques in AI systems. The important aspects of our work are:

- Defining a canonical knowledge base (E_x-OAV KB) and necessary algorithms to convert all knowledge representation techniques to E_x-OAV KB. Having all knowledges in one knowledge representation technique make the system independent to knowledge representation technique. It is also application and domain independent.
- OAV-VVT expert in addition of covering all common issues of V&V considers the semantic & logical contradiction too.
- OAV-VVT expert consists reasoning component in order to make the tool more powerful.
- OAV-VVT expert can be either used during the building of knowledge base or refinement of existing knowledge base.
- OAV-VVT expert consists an explanation component in order to describe and answer "how" and "why", in presence of invalid knowledge in KB.
- OAV-VVT expert is able to check the illegal attribute value as well as unreferenced attribute value during the transforming of the existing KB to E_x-OAV KB to increase the performance of V&V.

In this work, OAV-VVT expert is presented as a V&V tool with new approach, E_x-OAV KB, which is concentrate on OAV knowledge representation technique.

OAV-VVT expert is a good test bed for future work and use the ontology for finding the semantic contradiction. The issue of ambiguity in V&V should consider in future work by classifying the invalid knowledge in different categories.

References

[1] B.W. Boehm, "Verifying and Validating Software Requirements and Design Specifications," *IEEE software*, vol. 1,no. 1, pp.75-88, 1984.

[2] J. Ignizio, *The Development and Implementation of Rule-based Expert systems*, McGraw-Hill, 1991.

[3] S. Delab and O.J. Mengshoel, "Knowledge Validation: Principle and Practice," *IEEE Expert*, pp. 62-68, June 1993.

[4] S. Smith and A Kandel, *Verification and Validation of Rule-based Expert Systems*, CRC Press, 1993.

[5] C.H. Wu, S. Lee and H.S. Chou, "Dependency Analysis for Knowledge Validation in Rule-based Expert Systems," *The* 10nd *IEEE Conference on AI for applications*, 1994.

[6] M.C. Rousset, F. Bouali and S. Loiseau, "Rule base Debugging: a Proposal based on diagnosis theory," *5th International workshop on principle of diagnosis DX-94*, pp. 42-46, New Paltz, NY, 1994.

[7] W.T. Tsai, R. Vishuvajjala and D. Zhang, "Verification and Validation Of Knowledge-Based Systems," *IEEE Transaction On Knowledge and Data Engineering*, vol. 11, no. 1, 1999.

[8] R. Knauf, I. Phillippow and A.J. Gonzales., "Towards Verification and Refinement of Rule-based Systems," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 12, pp. 421-431, 2000.

[9] P.H. Winston, *Artificial Intelligence*, Addison-Wesley Publishing Company, 1984.

[10] C. Dadkhah, "The Transforming Algorithms: Transform the Major Knowledge Representation Method to E_x-OAV KB," Technical Report, No: CE/TR.DS/82/03, Computer engineering Department, Amirkabir University of Technology, 2003.

[11] M. Ayel and J.P. Laurent, *Validation, Verification and Test of Knowledge-based Systems*, John Wiley & Sons, 1991.

[12] T. Bench-Capon, F. Coenen, H. Nwana, R. Paton and M. Shave, "Two Aspects of the Validation and Verification of Knowledge-based Systems," *IEEE Expert*, pp. 76-81, 1993.

[13] A. Ginsberg, "knowledge - base Reduction: a New Approach to Checking Knowledge bases for Inconsistency and Redundancy, " *AAAI88*, vol. 2, pp. 585-589, 1988.

[14] J. Debenham, "Validity of First Order Knowledge Base," *14th Int'l Conf. Florida AI Research Society (FLAIRS)*, pp. 21-23, 2001.

[15] T. Onoyama and S. Tsurutu, "Validation Method for Intelligence Systems," *Journal of Experimental and Theoretical Artificial Intelligence*, vol 12, no. 4, pp. 461-472, 2000.

[16] R. Knauf, L. Philippow, A. J. Konzales and K. P. Jantke, " Towards Validation of Rule Based Systems, The loop is Closed," *13th Int'l Conf. FLAIRS*, pp. 331-335, 2000.

[17] A. Vermeasan and F. Coenen, *Validation and Verification of Knowledge- based systems (Theory, Tools and Practice)*, Kluwer, 1999.

[18] F. Bouali, "Diagnostic, Validation et Reparation de bases de Connaissances: Le Systeme KBDR," Theses de doctorat, l'universite de Paris-Sud, Orsay-France, 1996.

[19] M.C. Rousset, "On the Consistency of Knowledge bases: the COVADIS System," *ECAI88*, pp. 79-84, 1984.

[20] H. Lounis, "Vérification de bases de Connaissance : une Approche base sur L'apprentissage Automatique," *Actes des JFVAV 93*, pp. 1-15.

[21] V. Dotsh, K.P. Jantke, "TIC-A Toolkit for Validating in Formal Language Learning," *13th Int'l Conf. FLAIRS*, 2000.

[22] J. Yen, J. Lee and D. Haminlton, "Designing Verifiable Expert Systems," *Proceeding of IEEE 2nd International*

Conference on Tools for AI (ICTAI'90), pp. 878-884, 1990.

[23] J. Mohnar and C. Sarteschi, "Validation of the Technicien's Assister System," *IEEE Conference on Managing Expert Systems Programs and Projects*, pp. 201-204, 1990.

[24] J.G. McGuire, "Uncovering Redundancy and Rule Inconsistency in Knowledge Bases via Deduction ," *IEEE, Proceeding of the Fifth Annual Conference on Systems Integrity, Software Safety and Security, COMPASS90*, pp. 56-67, 1990.

[25] S. Loiseau, " Refinement of Knowledge Bases based on Consistency,*" ECAI 92*, pp. 845-849, 1992.

[26] J. Haugh, "Formal Software Development Techniques Applied to Expert System Production," Information Technology Division, Admiratly research establish ports down, Hampshire, PO6 4 AA, 1990.

[27] L. Brisoux and E. Gregoire and S. Lakhdar, "Validation of Knowledge-based Systems by Means of Stochastic Search," *Proc. IEEE, 1st International Workshop on Validation & Verification & Integrity, Issues of Expert and Data Base Systems*, pp. 41-46, 1998.

[28] A. Beavieux and P. Dague, "A General Consistency Checking and Restoring Engine for Knowledge Bases," *ECAI90*, pp. 77-82, 1990.

[29] S. Smith.and A. Kandel, "Verification and Validation in Fuzzy Expert Systems, " *IEEE Systems, Man and Cybernetics*, pp 3647-3651, 1990.

[30] A. Vermeasan, "Software Certification for Industry: Verification and Validation Issues in Expert Systems," *IEEE Standard for Software Quality Assurance*, pp. 3-14, 1998.

[31] R. Levin, D. Drang and B. Edelson, *AI and Expert Systems*, McGraw-Hill, 1990.

[32] S. Russel and P. Norving, *Artificial Intelligence, A Modern Approach*, Prentice Hall, New Jersey, 1995.

[33] J. E. Tatem, "Knowledge Base Documentation: A Productivity Tool for Large Knowledge Bases," *IEEE Aerospace & Electronics Conference, NAECON90*, pp. 1082-1094, 1990.

[34] R. Reiter, "A theory of Diagnosis from First Principals," *Artificial Intelligence* , vol 32, pp. 57-95, 1987.

[35] W. Nejdl and J. Bachmayen, "Diagnosis and Repair Iterative Planning Verses V-step Look Ahead Planning," *Fourth International Workshop on Principals of Diagnosis*, 1993.

[36] J. Darkin, *Expert Systems Design and Development*, Macmillan, 1998.

[37] C. Dadkhah, "OAV-VVT, New method for Verification and Validation of AI system using OAV knowledge representation," Doctoral Proposal, No: CE/PD.DS/81/01, Computer engineering Department, Amirkabir University of Technology, 2002.

[38] R. Greiner, B.A. Smith, and R.W. Wilkerson, "A Correction to the Algorithm in Retiers' Theory of Diagnosis," *Artificiel Intelligence*, vol. 41, pp. 79-88, 1989.

[39] G. Friedrich, G. Gottlob and W. Medjl, "Formalizing the Repair process," *Proceeding of the Europeen Conference on AI, ECAI92*, pp. 703-713, 1992.

Chitra Dadkhah received the B.S degree in Software Engineering from Shahid Beheshti University, Tehran, Iran in 1990. she received the Management Information System Courses of M.S degree from Azad University, Tehran, Iran in 1992. She also received the M.S degree in Computer Engineerimg (Artificiel Intelligence) from Paris XI (Orsay) University , Paris, France in 1994. At present, she is a Ph.D student in the Computer Engineering Department at Amirkabir University of Technology in Tehran, Iran. Her research interests include Expert System, Knowledge Representation Techniques, Artificiel Intelligent Techniques, Object –Attribute-Value Triple, Verification & Validation of the Knowledge.
Email: Dadkhah@ce.aut.ac.ir

Ahmad Abdollahzadeh Barforosh received the B.S degree in accounting from Tehran University in Tehran, Iran in 1975. He received the M.S degrees in Computer science from West Coast University, Los Angeles, california, U.S.A in 1980. He also received the Ph.D in computer science from university of Bristol,England,1990. Dr.Abdollahzadeh serves as visiting professor in department of computer science of Maryland university at college park, USA and Orsay university in Paris, France from 2000-2002. Currently, he is a associate professor in the IT & Computer Engineering Departement at AmirKabir University of Technology in Tehran, Iran. His research interests include, Information Retrieval, Artificiel Intelligence Techniques, Expert System, Natural Language Proceesing, Decision Support System, Knowledge Representation, Software Engineering.
Email: ahmad@ce.aut.ac.ir