VOLIME 1, NUMBER 1, 2016

The Journal of Applied Science University

http://www.asu.edu.bh/asj

Published by Applied Science University, Kingdom of Bahrain.

ISSN 2210-1764

Scope

The ASU Journal publishes papers covering all aspects of Applied Sciences and in particular those core disciplines of the University, namely, Business Administration, Art abd Computer Science and Law. All articles (full length/short papers) should include a validation of the idea presented, e.g. through case studies, experiments, or systematic comparisons with other approaches already in practice.
The journal welcomes state-of-the-art surveys and reports of practical experience for all the topics of interest Submitted articles should not have been previously published or be currently under consideration for publication elsewhere. Conference papers may only be submitted if the paper has been completely re-written (taken to mean more than 50%) and the author has cleared any necessary permissions with the copyright owner if it has been previously copyrighted. All papers are refereed through a double-blind process.

Frequency

Initially, ASJ will have **three** issues annually. This may change depending on how the journal evolves.

Short Communications

The goal of the Short Communications corner is both to present information and to stimulate thought and discussion. Topics chosen for this coverage are not just traditional formal discussions of research work; they also contain ideas at the fringes of the field's "conventional wisdom". Articles in this category will succeed only to the extent that they stimulate not just thought, but action.

Boards

The Journal is governed and managed by an **Advisory Board** (AB) and an **Editorial Board** (EB). Each issue has its own EB with an Editor-in-Chief who is appointed by the AB.

Advisory Board.

The AB has eight members, two of whom are external. Members are

- Dean of Scientific Research and Graduate Studies (Chair)
- VP - Academics
- VP - Finance, Administration and Community Engagement
- Dean of Law
- Dean of Administrative Sciences
- Dean of Art and Science
- Professor Jonathan Blackledge - VP (Research), University of KwaZulu-Natal, Westville (South Africa)

- Professor George Tovestiga - Henley Business School, Reeding University (UK)

Preparation of Manuscript

There are no strict formatting requirements but all manuscripts must contain the essential elements needed to convey your manuscript, for example Abstract, Keywords, Introduction, Materials and Methods, Results, Conclusions, Artwork and Tables with Captions. If your article includes any Videos and/or other Supplementary material, this should be included in your initial submission for peer review purposes. Divide the article into clearly defined sections.
Please ensure the figures and the tables included in the single file are placed next to the relevant text in the manuscript, rather than at the bottom or the top of the file.

References

There are no strict requirements on reference formatting at submission. References can be in any style or format as long as the style is consistent. Where applicable, author(s) name(s), journal title/book title, chapter title/article title, year of publication, volume number/book chapter and the pagination must be present. Use of DOI is highly encouraged. The reference style used by the journal will be applied to the accepted article by Elsevier at the proof stage. Note that missing data will be highlighted at proof stage for the author to correct.

Use of word processing software

Regardless of the file format of the original submission, at revision you must provide us with an editable file of the entire article. Keep the layout of the text as simple as possible. Most formatting codes will be removed and replaced on processing the article. The electronic text should be prepared in a way very similar to that of conventional manuscript. To avoid unnecessary errors you are strongly advised to use the 'spell-check' and 'grammar-check' functions of your word processor.
We strongly recommend you to use the Elsevier article class elsarticle.cls [1] to prepare your manuscript and BibTeX [2] to generate your bibliography. For detailed submission instructions, templates and other information on LaTeX [3]

[1] (http://www.ctan.org/tex-archive/macros/latex/contrib/elsarticle)
[2] (http://www.bibtex.org)
[3] http://www.elsevier.com/latex.

Editorial Board: Volume 1, Issue 1 (2016)

Contents

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Changing Data Representations via Abstraction and Refinement in FermaT

Martin Ward
Software Migrations Ltd., U.K.

Usually in program transformation theory the aim is to preserve functional equivalence of programs. However, there are cases where exact functional equivalence is not desirable. When a program is converted from an implementation using one data model to an implementation using a different data model, the two versions of the program are not semantically equivalent, but there is a sense in which they implement the same function. In this paper we present a technique for changing data representations based on a process of abstraction and refinement and implemented in the FermaT program transformation system. A practical application of this technique is migrating from low-level assembler code on a mainframe, which uses bit fields and compressed data, to a COBOL system based on character fields and expanded data.

Categories and Subject Descriptors: []:

General Terms: D.2.4 [Software Engineering]: Software/Program Verification; D.2.5 [Software Engineering]: Testing and Debugging; D.2.2 [Software Engineering]: Design Tools and Techniques

Additional Key Words and Phrases: Abstraction; Refinement; Formal Methods; FermaT; Program Transformation; Software Development

1. INTRODUCTION

In this paper we will outline a formal method for converting a program from one data model to a different data model: i.e. changing the data representation of the program. The method is based on program transformation theory: specifically, program refinement and abstraction. We define an *abstraction function* which maps from the original concrete implementation of an abstract data type to the corresponding abstract value. A second abstraction function maps from the new concrete data type to the same abstract data type. We show that, given an abstraction function, we can derive a method to convert a program using one concrete representation to the corresponding abstract program, which can then be refined into a new program using the new concrete data representation.

The method is illustrated with a case study: migrating from low-level assembler code on a mainframe, which uses bit fields and compressed data, to a COBOL system based on

Correspondence address: Dr Martin Ward, Software Migrations Ltd., Spectrum House, Dunstable Road, Redbourn, St Albans, Hertfordshire AL3 7PR

character fields and expanded data.

2. PROGRAM TRANSFORMATION

A *program transformation* is any operation on a program which preserves the semantics. The semantics of a program is defined as a function which maps the program text (a syntactic object) plus information about the domain or environment, to a mathematical function or relation which defines the behaviour of the program. There are various functions and relations which can be used to define the semantics of programs, these include:

—A function which maps each initial state to the corresponding final state, or set of final states;

—A function which maps each initial state to the (finite or infinite) sequence of states which occur during the execution of the program;

—A relation which associates initial states with the corresponding final states.

If the semantics, or the definition of equivalence, involves only initial and final states then we classify it as a *denotational semantics*. If the semantics includes some or all of the states which occur during the execution of the program, then we classify it as an *operational semantics*. Any operational semantics can be used to define a corresponding denotational semantics via an equivalence relation which ignores the intermediate states: so if two programs are operationally equivalent, they will also be denotationally equivalent, but the converse is not true. A denotational semantics is more useful when program transformations are used to:

(1) Prove that a given program is a correct implementation of a given specification: here the specification may only define the relationship between the initial and final states, while the implementation will include many intermediate states;

(2) Prove the correctness of an optimising transformation. The more efficient program will generally take fewer intermediate states to compute the same output.

For the rest of this paper we will restrict attention to denotational semantics on non-deterministic programs. The semantics of a program is defined as a function from the initial state to the set of possible final states.

2.1 Data Representation

There are many different ways to implement a data structure in a program: for example, a stack can be implemented as an array with a pointer or a linked list of heap-allocated nodes.

Conceptually, a stack can be defined as an abstract sequence, with the stack operations

defined as operations on sequences:

$$\begin{aligned}
\mathsf{init}() &=_{\text{DF}} \langle\rangle \\
\mathsf{top}(s) &=_{\text{DF}} \begin{cases} \mathsf{Error} & \text{if } s = \langle\rangle \\ s[1] & \text{otherwise} \end{cases} \\
\mathsf{push}(s,x) &=_{\text{DF}} \langle x \rangle +\!\!+ s \\
\mathsf{pop}(s) &=_{\text{DF}} \begin{cases} \mathsf{Error} & \text{if } s = \langle\rangle \\ s[2..] & \text{otherwise} \end{cases} \\
\mathsf{is_empty?}(s) &=_{\text{DF}} \begin{cases} \textbf{true} & \text{if } s = \langle\rangle \\ \textbf{false} & \text{otherwise} \end{cases}
\end{aligned}$$

A proposed implementation of this abstract data type can be proved correct by means of an *abstraction function*: this is a function which maps from a concrete data structure to the corresponding abstract data value. There may be more than one concrete data structure which is a valid implementation of an abstract data value (for example: there are many ways to lay out a linked list structure in memory, all of which are valid representations of the same sequence).

Ideally, all abstract values should have at least one representation as a concrete value: but in practice this may not be possible. For example, if there are an infinite set of different abstract values, it is not possible to represent all of them in a finite computer system.

Suppose a stack is implemented as a linked list of nodes where, given a pointer `sp` to a node, `sp->data` is the data value and `sp->next` is a pointer to the next node in the list. A suitable abstraction function is:

$$\begin{aligned}
\phi(\texttt{NIL}) &=_{\text{DF}} \langle\rangle \\
\phi(\texttt{sp}) &=_{\text{DF}} \langle \texttt{sp->data} \rangle +\!\!+ \phi(\texttt{sp->next})
\end{aligned}$$

Proving the correctness of a concrete implementation therefore consists of proving that the abstract value of the output of a concrete function is the same as the output of the abstract function applied to the abstract value(s) of the input(s).

To be precise, suppose f is an abstract operation on one input, and F is the corresponding concrete implementation. To prove that the implementation to be correct, we need to prove that for all concrete values x:

$$f(\phi(x)) = \phi(F(x))$$

In other words, we need to prove that the diagram in Figure 1 commutes.

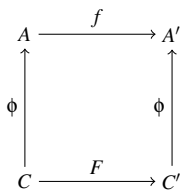


Fig. 1: The Abstraction Function ϕ

More generally, suppose $\mathbf{S}^A$ is an abstract program which uses abstract variable a and $\mathbf{S}^C$ is the corresponding concrete program which uses concrete variable x. The two programs $\mathbf{S}^A$ and $\mathbf{S}^C$ are not semantically equivalent: for a start, they operate on different variables! But using the abstraction function we can construct two programs which should be semantically equivalent. We use these WSL (Wide Spectrum Language) statements:

—**add**($\mathbf{x}$) adds the variables in the list $\mathbf{x}$ to the state space and assigns arbitrary values to them;

—**remove**($\mathbf{x}$) removes the variables in the list $\mathbf{x}$ from the state space.

See [5] for a description of the syntax and semantics of the WSL language. For brevity, we write $\mathcal{R}(x)$ as an abbreviation for **remove**($\langle x \rangle$), which removes a single variable from the state space. Any assignment to x before $\mathcal{R}(x)$ with no intervening reference to x can be deleted since it cannot affect the final state: there is no variable x in the final state space of a program which ends in $\mathcal{R}(x)$.

With this notation, we need to prove that the following semantic equivalence:

$$a := \phi(x);\ \mathcal{R}(x);\ \mathbf{S}^A \ \approx\ \mathbf{S}^C;\ a := \phi(x);\ \mathcal{R}(a)$$

One disadvantage of the above is that the programs take a concrete value as input and produce an abstract value as output: so they are a kind of "hybrid" between abstract and concrete.

Using the WSL specification statement we can use the abstraction function ϕ to convert abstract values to concrete values. A specification statement has the form:

$$\mathbf{x} := \mathbf{x}'.\mathbf{Q}$$

where $\mathbf{x}$ is a list of variables, $\mathbf{x}'$ is the corresponding list of "primed variables" and $\mathbf{Q}$ is any formula. This statement assigns new values to the variables in $\mathbf{x}$ such that the formula $\mathbf{Q}$ is satisfied, where within $\mathbf{Q}$ the primed variables $\mathbf{x}'$ are the new values (to be assigned to $\mathbf{x}$) and $\mathbf{x}$ are the original values. For example, a simple assignment statement $x := x + 1$ can be defined as:

$$\langle x \rangle := \langle x' \rangle.(x' = x + 1)$$

If there is more than one set of values which can be assigned to $\mathbf{x}'$ to satisfy $\mathbf{Q}$ then one set of values is selected non-deterministically. If there are no values which satisfy $\mathbf{Q}$, then the statement aborts.

The list of variables $\mathbf{x}$ can be empty: this means that the statement cannot change the value of any variable. Either $\mathbf{Q}$ is already satisfied (in which case the statement terminates normally) or $\mathbf{Q}$ is not satisfied (in which case the statement aborts). So a specification statement with an empty variable list is precisely equivalent to an assertion statement:

$$\langle\rangle := \langle\rangle.\mathbf{Q} \ \approx\ \{\mathbf{Q}\}$$

Using the specification statement we can write a statement which converts an abstract value to a suitable concrete value, with the particular value being selected nondeterministically:

$$\langle x \rangle := \langle x' \rangle.(\phi(x') = a)$$

With this statement, the semantic equivalence we need can be defined as an equivalence between two abstract programs:

$$\mathbf{S}^A \;\approx\; \mathbf{add}(\langle x\rangle);\ \langle x\rangle := \langle x'\rangle.(\phi(x') = a);\ \mathbf{S}^C;\ a := \phi(x);\ \mathcal{R}(x) \tag{1}$$

To prove the correctness of the concrete data representation we need to find a suitable concrete program and then prove the above program equivalence. Note that due to the non-determinacy in the specification statement, if we are able to prove the program equivalence, then we have proved that our concrete program is a correct implementation of the abstract program *regardless* of the concrete value selected to represent a given abstract value.

In the next section we describe a method for *deriving* the concrete program, given the abstract program and the abstraction function.

2.2 Deriving the Concrete Program

In [6,7] we present a method for deriving a program from an abstract specification by means of correctness-preserving refinement and transformation steps guided by informal implementation ideas. In this section we show how the program derivation method can be extended to include changes in the data representation: specifically from an abstract data type to a concrete implementation of the data structures.

We use the fact that any translation from an abstract to a concrete value is reversible, i.e.:

$$\mathbf{add}(\langle x\rangle);\ \langle x\rangle := \langle x'\rangle.(\phi(x') = a);\ a := \phi(x);\ \mathcal{R}(x) \;\approx\; \mathbf{skip}$$

So:

$$\mathbf{S}^A \;\approx\; \mathbf{add}(\langle x\rangle);\ \langle x\rangle := \langle x'\rangle.(\phi(x') = a);\ a := \phi(x);\ \mathcal{R}(x);\ \mathbf{S}^A$$

Note that the converse is not necessarily true: there may be several different concrete values which represent the same abstract value.

We may assume that the concrete variable x does not appear in $\mathbf{S}^A$, so the **remove** statement can be moved past $\mathbf{S}^A$:

$$\mathbf{S}^A \;\approx\; \mathbf{add}(\langle x\rangle);\ \langle x\rangle := \langle x'\rangle.(\phi(x') = a);\ a := \phi(x);\ \mathbf{S}^A;\ \mathcal{R}(x)$$

The next step is to "push" the assignment $a := \phi(x)$ into the structure of $\mathbf{S}^A$, through the structure, transforming the program as we go, and out at the end. We may assume that the concrete variable x does not appear in $\mathbf{S}^A$.

First, consider the atomic (non-compound) statements:

—If $\mathbf{S}^A$ does not reference a then:

$$a := \phi(x);\ \mathbf{S}^A \;\approx\; \mathbf{S}^A;\ a := \phi(x)$$

—If $\mathbf{S}^A$ is an assertion $\{\mathbf{Q}\}$ which refers to a, but not x, then:

$$a := \phi(x);\ \{\mathbf{Q}\} \;\approx\; \{\mathbf{Q}[\phi(x)/a]\};\ a := \phi(x);$$

where $\mathbf{Q}[\phi(x)/a]$ denotes $\mathbf{Q}$ with every free occurrence of a replaced by $\phi(x)$.

—If $\mathbf{S}^A$ is a specification statement which assigns to a, say $\langle a\rangle := \langle a'\rangle.\mathbf{Q}^A$, then define:

$$\mathbf{Q}^C =_{\text{DF}} \mathbf{Q}^A[\phi(x)/a][\phi(x')/a']$$

Any assignment can be inserted just before a **remove** statement, so:

$$\begin{aligned} a := \phi(x);\ \langle a\rangle := \langle a'\rangle.\mathbf{Q}^A;\ \mathcal{R}(x) \\ \approx\ & a := \phi(x);\ \langle a\rangle := \langle a'\rangle.\mathbf{Q}^A \\ & \langle x\rangle := \langle x'\rangle.\mathbf{Q}^C;\ \mathcal{R}(x) \\ \approx\ & a := \phi(x); \\ & \langle a,x\rangle := \langle a',x'\rangle.(\mathbf{Q}^A \wedge \mathbf{Q}^C);\ \mathcal{R}(x) \\ \approx\ & a := \phi(x); \\ & \langle a,x\rangle := \langle a',x'\rangle.(a' = \phi(x') \wedge \mathbf{Q}^C);\ \mathcal{R}(x) \end{aligned}$$

since $\phi(x')$ satisfies $\mathbf{Q}^A$, given that $a = \phi(x)$

$$\begin{aligned} \approx\ & a := \phi(x);\ \langle x\rangle := \langle x'\rangle.\mathbf{Q}^C;\ a := \phi(x);\ \mathcal{R}(x) \\ \approx\ & \langle x\rangle := \langle x'\rangle.\mathbf{Q}^C;\ a := \phi(x);\ \mathcal{R}(x) \end{aligned}$$

since the first assignment to a is overwritten by the second. So we can set $\mathbf{S}^C$ to $\langle x\rangle := \langle x'\rangle.\mathbf{Q}^C$.

For the compound statements, we assume that a suitable $\mathbf{S}^C$ exists for all statements with a lower degree of recursion nesting and for all smaller statements than $\mathbf{S}^A$:

—Suppose $\mathbf{S}^A$ is of the form $\mathbf{S}^A_1;\ \ldots;\ \mathbf{S}^A_n$. By the induction hypotheses, these exist statements $\mathbf{S}^C_1, \ldots, \mathbf{S}^C_n$ such that

$$a := \phi(x);\ \mathbf{S}^A_i;\ \mathcal{R}(x) \approx \mathbf{S}^C_i;\ a := \phi(x)\mathcal{R}(x)$$

Then:

$$\begin{aligned} a := \phi(x);\ \mathbf{S}^A_1;\ \ldots;\ \mathbf{S}^A_n;\ \mathcal{R}(x) \\ \approx\ & a := \phi(x);\ \mathbf{S}^A_1;\ \mathcal{R}(x);\ \ldots;\ \mathbf{S}^A_n \end{aligned}$$

since none of the $\mathbf{S}_i$ refers to x

$$\approx\ \mathbf{S}^C_1;\ a := \phi(x);\ \mathcal{R}(x);\ \ldots;\ \mathbf{S}^A_n$$

by the induction hypothesis for $\mathbf{S}^A_1$

$$\approx\ \mathbf{S}^C_1;\ \ldots;\ \mathbf{S}^C_n;\ a := \phi(x);\ \mathcal{R}(x)$$

by induction.

—Suppose $\mathbf{S}^A$ is of the form **if then** $\mathbf{S}^A_1$ **else** $\mathbf{S}^A_2$ **fi**. By the induction hypotheses, these exist statements $\mathbf{S}^C_1$ and $\mathbf{S}^C_2$ which satisfy the required condition. Then:

$$\begin{aligned} a := \phi(x);\ \textbf{if then}\ \mathbf{S}^A_1\ \textbf{else}\ \mathbf{S}^A_2\ \textbf{fi};\ \mathcal{R}(x) \\ \approx\ & \textbf{if}\ [\phi(x)/a]\ \textbf{then}\ a := \phi(x);\ \mathbf{S}^A_1;\ \mathcal{R}(x); \\ & \qquad \textbf{else}\ a := \phi(x);\ \mathbf{S}^A_2;\ \mathcal{R}(x)\ \textbf{fi} \\ \approx\ & \textbf{if}\ [\phi(x)/a]\ \textbf{then}\ \mathbf{S}^C_1;\ a := \phi(x);\ \mathcal{R}(x) \\ & \qquad \textbf{else}\ \mathbf{S}^C_2;\ a := \phi(x);\ \mathcal{R}(x)\ \textbf{fi} \end{aligned}$$

by the induction hypothesis

$$\approx \textbf{ if } [\phi(x)/a] \textbf{ then } \mathbf{S}_1^C;\ \textbf{ else } \mathbf{S}_2^C \textbf{ fi};$$
$$a := \phi(x);\ \mathcal{R}(x);$$

—Suppose $\mathbf{S}^A$ is a recursive procedure $(\mu X.\mathbf{S}_1^A)$. Each finite truncation $(\mu X.\mathbf{S}_1^A)^n$ of $\mathbf{S}^A$ has a lower level of recursion nesting, so by the induction hypothesis there exists a statement $\mathbf{S}_1^C$ such that:

$$a := \phi(x);\ (\mu X.\mathbf{S}_1^A)^n;\ \mathcal{R}(x) \approx (\mu X.\mathbf{S}_1^C)^n;\ a := \phi(x);\ \mathcal{R}(x)$$

Therefore:

$$\begin{aligned} a := \phi(x);\ (\mu X.\mathbf{S}_1^A)^{n+1};\ \mathcal{R}(x) &\approx a := \phi(x);\ \mathbf{S}_1^A[(\mu X.\mathbf{S}_1^A)^n/X];\ \mathcal{R}(x) \\ &\approx \mathbf{S}_1^C[a := \phi(x);\ (\mu X.\mathbf{S}_1^A)^n;\ \mathcal{R}(x)/X] \end{aligned}$$

by applying all the previous cases recursively over the structure of $\mathbf{S}_1^A$

$$\approx \mathbf{S}_1^C[(\mu X.\mathbf{S}_1^C)^n a := \phi(x);\ \mathcal{R}(x)/X]$$

by the induction hypothesis for n

$$\approx \mathbf{S}_1^C[(\mu X.\mathbf{S}_1^C)^n/X];\ a := \phi(x);\ \mathcal{R}(x)$$

by again applying all the previous cases recursively over the structure of $\mathbf{S}_1^C$

$$\approx (\mu X.\mathbf{S}_1^C)^{n+1};\ a := \phi(x);\ \mathcal{R}(x)$$

as required.

As so often happens, the base case for the induction is trivial since $(\mu.X.\mathbf{S})^0 = \textbf{abort}$ for any $\mathbf{S}$.

This completes the proof.

By applying the above rules, given an abstraction function and an abstract program we can *construct* a corresponding concrete program which applies the change in data representation (as defined by the abstraction function) to the abstract program.

In the development of a large program there may well be several places where the data representation is changed from a more abstract representation to a more concrete representation.

2.3 Example Data Representation Change

Consider the following example program:

proc $F(n)$ $\equiv$
  if $n > 0$ **then** $F(n-1);\ x := g(n,x);\ F(n-1)$ **fi.**

We can convert the parameter n to a global variable by decrementing and incrementing it around the recursive calls:

proc $F()$ $\equiv$
  if $n > 0$ **then** $n := n-1;$
    $F();\ x := g(n+1,x);\ F();$
    $n := n+1$ **fi.**

The generic recursion removal theorem, described in [2] uses a stack (L) to transform a recursive program to an equivalent iterative program. The application of this transformation is purely mechanical and automatic and results in the following iterative program which is equivalent to the procedure F:

var $\langle L := \langle\rangle, d := 0\rangle$:
while $n > 0$ **do** $n := n-1$; $L := \langle 1\rangle$ ++ L **od**;
do do if $L = \langle\rangle$ **then exit**(2) **fi**;
$d := L[1]$; $L := L[2..]$;
if $d = 1$
then $x := g(n+1,x)$; $L := \langle 0\rangle$ ++ L; **exit**
else $n := n+1$ **fi od**;
while $n > 0$ **do**
$n := n-1$; $L := \langle 1\rangle$ ++ L **od od end**

Note that the only values pushed onto the stack are 0 and 1, so we can represent the stack using an integer c such that the digits in the binary representation of c are the elements of the stack. Our abstraction function is:

$$\begin{aligned} \phi(1) &=_{\text{DF}} \langle\rangle \\ \phi(c) &=_{\text{DF}} \langle c \bmod 2\rangle \text{ ++ } \phi(\lfloor c/2 \rfloor) \end{aligned}$$

Note that ϕ is undefined for $c = 0$. If we apply this abstraction function to the abstract program, using the methods defined above, we immediately derive the following concrete program:

var $\langle c := 1, d := 0\rangle$:
while $n > 0$ **do** $n := n-1$; $c := 2.c+1$ **od**;
do do if $c = 1$ **then exit**(2) **fi**;
$d := c \bmod 2$; $c := \lfloor c/2 \rfloor$;
if $d = 1$ **then** $c := \lfloor c/2 \rfloor$; $x := g(n+1,x)$;
$c := 2.c$; **exit**
else $n := n+1$ **fi od**;
while $n > 0$
do $n := n-1$; $c := 2.c+1$ **od od end**

The **while** loop sets c to $c.2^n + 2^n - 1$ and sets n to zero. We will absorb the assignment to c into the loop. If we also expand the statement **if** $d = 1 \ldots$ backwards, we can remove local variable d:

var $\langle c := 2^{n+1} - 1\rangle$:
do $n := 0$;
do if $c = 1$ **then exit**(2) **fi**;
if odd?(c)
then $c := \lfloor c/2 \rfloor$; $x := g(n+1,x)$; $c := 2.c$;
$c := c.2^{n+1} - 1$; **exit**
else $c := c/2$; $n := n+1$ **fi od od end**

When c is odd, the statements $c := \lfloor c/2 \rfloor$; $c := 2.c$ are equivalent to $c := c-1$. When c is even we will repeatedly increment n and divide c by 2 until it becomes odd. This suggests applying entire loop unrolling to the inner loop:

do $n := 0$;
  do if $c = 1$ **then exit**(2) **fi**;
    while even?(c) **do** $c := c/2$; $n := n+1$ **od**;
    if $c \neq 1$ **then** $c := c-1$; $x := g(n+1,x)$;
      $c := 2^n.(c+1)-1$**exit fi od od**

If $c = 1$ in the second **if** statement, then on the next iteration, both loops will exit. So the inner loop is a dummy loop which can be removed. The **while** loop is equivalent to the statement: $n := n + \mathsf{ntz}(c)$; $c := c/2^{\mathsf{ntz}(c)}$ where the function $\mathsf{ntz}(c)$ returns the number of trailing zeros in the binary expansion of c. Since n is initially zero, this is equivalent to: $n := \mathsf{ntz}(c)$; $c := c/2^n$. If the result is not 1 (i.e. if c had more than a single 1 digit in the expansion) then assignments:

$$n := \mathsf{ntz}(c);\ c := c/2^n;\ c := c-1;\ c := 2^n.(c+1)-1$$

simplify to $n := \mathsf{ntz}(c)$; $c := c-1$, so we can remove the variable n and replace it by $\mathsf{ntz}(c)$:

do if $c = 1$ **then exit fi**;
  if $c/2^{\mathsf{ntz}(c)} \neq 1$
    then $x := g(\mathsf{ntz}(c)+1,x)$; $c := c-1$
    else $c := c/2^{\mathsf{ntz}(c)}$; **exit od od**

The test $c = 1$ is redundant, since if $c = 1$ then $\mathsf{ntz}(c) = 0$, so the loop will exit anyway without changing c. So the loop is equivalent to a **while** loop:

while $c/2^{\mathsf{ntz}(c)} \neq 1$ **do**
  $x := g(\mathsf{ntz}(c)+1,x)$; $c := c-1$ **od**

Finally, we can represent c by the integer i with abstraction function $c = i + 2^{n+1}$, since c starts out greater than 2^{n+1}, and the loop terminates as soon as $c = 2^{n+1}$, and for $i > 0$ we have $\mathsf{ntz}(c) = \mathsf{ntz}(i)$:

var $\langle i := 2^n - 1\rangle$:
while $i > 0$ **do**
  $x := g(\mathsf{ntz}(c)+1,x)$; $c := c-1$ **od end**

While this iterative form for the original recursion has been known for many years [1], the importance of the result is that the iterative form was derived directly from the recursive program simply by the application of a generic recursion removal transformation (which introduces a stack), a change in data representation to convert the stack of bits to a single integer, followed by some generic optimisation transformations to simplify the result. This sequence of generic transformations has been applied to many different types of algorithm [4,6,7].

3. ABSTRACTION AND REFINEMENT

The previous section suggests a means to prove the connection between two different concrete data representations where there is no suitable abstraction function to connect them:

(1) Find an appropriate abstract data type such that both representations are valid implementations of this data type; then

(2) Determine suitable abstraction functions and prove that the corresponding diagrams commute.

For example: suppose we have a program which uses a stack implemented as a fixed size array with a stack pointer. We want to re-write the program to use a linked list: this will remove the fixed limitation on the size of the stack (although it will still be limited by the amount of available memory) and will ensure that a stack containing a small number of elements will only consume a small amount of memory.

A suitable abstraction function for the array implementation is:

$$\psi(i,A) = \begin{cases} \langle\rangle & \text{if } i = 0 \\ \langle A[i]\rangle \mathbin{+\!\!+} \psi(i-1,A) & \text{when } i > 0 \end{cases}$$

where A is the array, and i is the stack pointer.

Pushing an element x onto the array is implemented as:

$$\mathsf{push\_A}(i,A,x) = \begin{cases} \mathsf{Error} & \text{if } i = N \\ i := i+1;\ A[i] := x & \text{otherwise} \end{cases}$$

where N is the maximum size of the array.

The relationship between the two implementations (array and linked list) is one of *abstraction and refinement*: we can *abstract* the array implementation to the stack abstract data type and then refine it to the linked list implementation.

The key to the usefulness of the abstraction and refinement process is the definition of the abstract data type: in general, *any* two programs are related via abstraction and refinement: **abort** is an abstraction of any program and any program is a refinement of **abort**. In practice, the abstraction and refinement relation is restricted by the abstract specification for the program: any abstraction step which gives a result which is *more* abstract than the specification can no longer guarantee that the final refinement is an implementation of the specification. With this restricted form of abstraction and refinement, the relation between the two versions of the program (i.e. the programs which use two different data representations) is that they are *both* refinements of the abstract specification, using two different abstraction functions.

4. CONSTRUCTING A SUITABLE ABSTRACTION FUNCTION

The abstraction and refinement technique is straightforward to apply when an abstract specification and development history is available. In this case, the "abstraction" part of the process is already present, and the developer just needs to start at the appropriate level of abstraction and refine to a new program using the new data representation. In the *transformational programming* approach to software development [6,7] it is recommended that the initial abstract specification and the transformational derivation of executable code should be maintained, not just the final version of the code (which can be regenerated by executing the derivation process on the initial specification). However, most of the code in existence was not generated by this method: indeed most code was not generated by *any* formal method. Any existing specifications or documentation may well be out of date and cannot be relied upon. So the only accurate definition of the meaning of a program is the code itself.

This causes a problem when carrying out the abstraction and refinement process: it may not be obvious which variables and code sections correspond to the implementation of a particular abstract data type. This problem is exacerbated in assembler systems: the code to implement a particular data type may be scattered through the program (for efficiency reasons) and data may be re-used in different operations.

For example, a concrete operation to push an element onto a stack implemented as an array is:

$$i := i + 1;\ A[i] := x$$

However, we cannot guarantee that these two statements appear together: they may be separated by other code, or may appear in a different order:

$$A[i-1] := x;\ i := i + 1$$

If we do not know how many stacks are being used in the program, or which variables are implementing these stacks, then there is an initial analysis required before any abstraction and refinement process can be completed: without further analysis it is impossible to tell whether or not a particular variable increment statement such as $j := j + 1$ is part of a stack operation implemented using an array.

Given a program which we need to migrate to a new data model, the first step in the analysis is therefore to determine the abstract data type corresponding to the concrete data stored in each variable.

For simplicity we assume in this paper that each variable holds data of a single type throughout the execution of the program. In general, a data item may be re-used to hold different abstract values in different parts of the program. This is particularly important for assembler programs where any memory variable can store data of any type, and where registers and work areas may be used and re-used frequently as the program executes. These situations can be handled by computing the Static Single Assignment (SSA) form of the program and then analysing the SSA form to group the set of assignments and references to a particular variable into partitions which are independent in the sense that systematically renaming all the occurrences of the variable in one partition does not affect the function of the program (because an assignment in one partition cannot affect any reference in a different partition). After this renaming, each variable van be expected to hold data of a single type throughout the execution of the program. This renaming process is beyond the scope of this paper.

4.1 Assembler to COBOL Migration

IBM mainframe assembler includes the following native data types, these are the types for which there are native instructions operating on that type:

—16, 32 and 64 bit binary numbers in two's complement format.

—Character strings of any length (up to $2^{64} - 1$ bytes). Characters are usually stored as 8-bit bytes. Older programs typically use the EBCDIC encoding, more recent machines can also use ASCII encoding. The older move and compare instructions (`MVC` and `CLC` respectively) operate on strings of up to 256 bytes, the newer `MVCL` and `CLCL` instructions operate on strings up to up to 16,777,215 bytes, while the "extended" instructions (`MVCLE` and `CLCLE`) use the entire contents of a 64 bit general register for the length of the move or compare.

—Unicode character strings with two or four byte characters.

—Packed decimal numbers. A packed decimal data item consists of $2n-1$ digits packed into n bytes, one digit per nybble. The lowest order nybble indicates the sign: `0x0C` for a positive value, `0x0D` for a negative value and `0x0F` for an unsigned value. A packed decimal data item can be 1 to 16 bytes long (1 to 31 digits).

—Zoned decimal data: a string of 1 to 256 characters representing a number with one digit per character. The least significant digit may have a sign encoded in its high nybble: `0xCn` for a positive value, `0xDn` for a negative value and `0xFn` for an unsigned value, where `n` is the least significant digit. Note that in the EBCDIC encoding digits 0–9 are encoded as `0xF0` to `0xF9` respectively. The `PACK` instruction converts zoned decimal to packed, while the `UNPK` instruction converts packed decimal to zoned.

—A hexadecimal floating point value consisting of a sign bit, a seven bit exponent (-64 to 63 as a power of 16) and either a six hexadecimal digit fraction (for a 32 bit floating point number), or a 14 hexadecimal digit fraction (for a 64 bit floating point number). More modern IBM mainframes also have 128 bit floating point numbers, in addition to binary and decimal floating point formats.

—Bit operations can be applied to data from 1 to 256 bytes in length.

In addition to the above native data types, there are also data types specific to the application: such as dates represented in various formats, customer codes, location codes and so on.

It is possible to migrate from assembler to COBOL while preserving all data exactly as it is represented in the assembler. This option has been used successfully in major migration projects, and may well be the best option: especially when parts of the system are remaining in assembler and need to inter-operate with the migrated COBOL code.

However, some COBOL compilers (such as the IBM mainframe COBOL compiler) have yet to implement the "new" features in the ISO/IEC 1989:2002 standard, which was published in 2002, with a CS (Committee Draft) available in 1997. This standard added support for bit fields and bit operations, but current mainframe compilers do not have native support for these operations.

In our case study (see Section 6) we are migrating from assembler to IBM mainframe COBOL which does not have native bit operations. Therefore, the requirement from the customer is to convert the data representation for bit fields, packed hex digit fields and packed decimal digit fields to COBOL strings. There are also code fields which need to be expanded: for example, a one byte country code is expanded to a four character code, and various date formats of various sizes are expanded to a standard ISO 8601 ten character date field of the form YYYY-MM-DD.

In our case study, an assembler fullword (32 bit data definition) could contain any of the following data structures:

—A 32 bit binary number. In this case, the variable will be implemented in COBOL as a data item with type `S9(9) COMP` which is a signed 32 bit binary, so no change is needed;

—A 31 bit pointer. Note that mainframe assembler also has 24 bit pointers which are stored in a 32 bit data item, where the top eight bits can be used for additional data. The initial IBM System/360 was released in 1964 and included a range of machines with different capabilities. The low-end models (such as the Model 40 which could be rented for a mere $20,000 a month) had an eight bit data bus: so it took four memory accesses

to load a 32 bit word. At the time, 24 bits seemed sufficient to store an address (16MB of RAM should be enough for anyone), so a 24 bit address mode was defined: programs in this mode could execute faster, since loading an address required only three memory accesses. Nearly 50 years later, modern 64 bit z/OS systems are still compatible with the 24 bit address mode in order to cater for legacy software.

—A string of four characters. This will be implemented in COBOL with the type `X(4)`, so the length of the COBOL data is the same as the length of the assembler data even though the types are different;

—A string of eight decimal digits. In the assembler, each byte contains two digits. In COBOL, there is no simple way to extract the low order or high order four bits from a character, so we want to implement this data item as a string of eight digits with type `9(8)`;

—A "packed decimal" data item: this will have seven decimal digits plus a four bit sign field in the lowest four bits. The sign is usually one of: `0x0C` for a positive value, `0x0D` for a negative value and `0x0F` for an unsigned value. Packed decimals are a native COBOL data format with type `S9(7) COMP-3`. The length and format of the COBOL data is the same as the assembler in this case..

—A string of eight hexadecimal digits. COBOL does not have a specific type for hex digits, so this will be implemented as a string of eight characters with type `X(8)`;

—A set of 32 bit flags. Standard mainframe COBOL does not have bit operations, other than as callable routines. These were added to the COBOL language in the COBOL 2002 ISO and ANSI standards, but IBM's mainframe COBOL compilers have yet to be updated to these standards. Traditionally, COBOL programmers implement a flag as a single character containing one of two values, such as "Y" or "N";

—A hexadecimal floating point value consisting of a sign bit, a seven bit exponent (-64 to 63 as a power of 16) and a six hexadecimal digit fraction. More modern IBM mainframes also have binary and decimal floating point formats.

Similarly, a halfword (16 bit data item) could be a number, or two characters, or four hex or decimal digits, or sixteen flag bits etc.

Our abstraction function needs to be defined in such a way that each data item is mapped to the appropriate abstract data type. However, the assembler data definition may simply state that variable `FOO` is a fullword, or a string of four characters, with nothing in the definition to indicate which of the above types of data will be stored in this variable. Even for packed decimals, we cannot rely on the assembler programmer declaring the data as a packed decimal: there is no type checking in assembler and nothing to stop the programmer from declaring data as one type and using it as another type.

So, in order to construct a suitable abstraction function automatically, we need additional information.

There are three main sources of information which can be used to determine the data type of a variable:

—Documentation for record layouts: these usually have documentation, most internal data does not. Note that although the record layout may be documented, different programs and even different modules within the same program may use non-standard field names;

—Documentation for utility module parameters: the case study system includes a number of utility modules which take parameters of known types. For example, one module is a

utility which takes two parameters: a five byte record consisting of five one byte fields (code, function, state, country and exp2) and a 47 byte record with seven fields. Whenever a call to this module is encountered, we can deduce the types of the parameters;

—Examine the operations applied to the variable: for example, if a bitwise AND or OR operation is applied with a constant parameter of `0xF0` or `0x0F` then the data is likely to be either packed decimal digits or hex digits. A packed decimal operation or comparison can only be applied to decimal data. A Pack instruction converts packed decimal data to zoned decimal and an Unpack instruction does the reverse, and so on;

Using this initial information, we can deduce information about unknown data items via *type inferencing*:

—If variable FOO is copied from BAR, then we can infer that FOO and BAR have the same data type: at least, from this point onwards in the program (see below);

—If variable FOO is compared against variable BAR then they have the same data type.

By analysing the operations carried out on data items (eg bit manipulation, packed decimal operations and comparisons, binary numeric operations, assignments of hex digit flag values and so on), we can infer the type of each data item. This type inference can then be propagated through the program: for example, if two data items are added via a packed decimal addition operation, then we know that both data items are packed decimals.

To carry out the type inferencing analysis we first construct the transitive closure of all simple assignments and comparisons to put the data names into a set of equivalence classes: we may assume that all data items in a class are of the same data type. This process is described in more detail in the discussion of the case study (Section 6).

File records may not be declared as such in the assembler code, so it may not be clear which data definitions describe known record structures. However, it is possible to determine from the source code and JCL (Job Control Language) instructions which files are used by each program, and therefore which record structures are potentially in use within each module.

We have implemented a process which compares the sequence of data definitions in the assembler against the set of possible record layouts. If the match is sufficiently close, then we can assume that the set of definitions will be applied to the data in the corresponding record layout. From the documentation for this record layout, we can determine the data type for each field in the record layout.

5. POINTER HANDLING

The main barrier to totally automated type inferencing is that data may be addressed via a pointer. Suppose an assembler instruction copies four bytes from the address in `R3` to the address in `R4`. Without any further information about the data pointed at, it is impossible to determine whether the four bytes should be left as four bytes, expanded to eight bytes or 32 bytes, or perhaps a ten byte date field, or some other combination (the four byte data area might consists of two or more fields of different types).

COBOL has native language features for pointers and pointer arithmetic, but these are not commonly used by COBOL programmers. So the assembler to COBOL migration engine includes a number of specialised transformations for tracking and eliminating pointers where possible. For example, when an assembler module processes an array of data

records, the usual assembler implementation is to use a register as a pointer which is incremented to process each record in sequence. The migration engine can detect this code and automatically convert the data representation to an array of records, replacing the register pointer by an index into the array. More general examples of pointer arithmetic will need human intervention to resolve the pointer.

6. CASE STUDY

Our case study consists of an assembler system currently in production in a large American insurance company. The system consists of over 3,000 programs and 8,991 assembler modules (many of which are used in more than one program).

Each assembler module is translated into low-level WSL and then transformed to restructure and simplify the WSL. See [3,5] for a description of the migration process. The data translation operation is applied before translation to COBOL and consists of the following stages:

(1) Equivalence Classes: Partition all data items into a equivalence classes as follows:
   (a) Process each statement which copies from one variable to another variable and ensure that both variables are placed in the same equivalence class. Create a new class if needed, or merge two existing classes;
   (b) Process all relations (less than, greater than, equal, etc.) and ensure that the operands are in the same class;
   (c) Process all packed decimal operations and ensure that the operands are in the same class;

(2) Type Inferencing: Examine the operations on all data items to determine the type of data:
   (a) An assignment which moves a hex value other than `0x40` (an EBCDIC space), `0x00` or `0xFF` indicates that the target contains hex digits or packed decimal digits
   (b) An assignment which moves a hex string indicates that the target contains hex digits;
   (c) An assignment which moves a string other than a hex string indicates that the target contains string data;
   (d) A bit operation with a hex value of `0x0F` or `0xF0` indicates that the target and other parameter contains hex digits or packed decimal digits;
   (e) A packed decimal operation shows that the data contains packed decimals: unless it is a pack or unpack instruction applied to a single byte. It is extremely rare to find a single byte packed decimal (since this can only contain values the -9 to +9), but quite common for the `PACK` and `UNPK` instructions to be applied to a single byte in order to reverse the nybbles in the byte. In the latter case, we may assume that the byte contains two hex digits;
   (f) If an unpack operation is applied with a length one greater than the defined length of the source field, then we may assume that the source field actually contain packed decimal digits with no sign nybble;
   (g) Finally, check for calls to utility modules which take parameters of known types;

   The order of the above operations ensures that the more accurate information overrides less accurate information. For example, if a field is known to contain packed decimal digits this overrides any previous determination that the field contains packed hex

digits (since the latter determination cannot usually distinguish between packed hex and packed decimal digits);

(3) Data Conversion: The final stage is to translate operations on the old data to the corresponding operation on the new data. For example, extracting a nybble from a byte which contains hex or packed digits translates to extracting the appropriate character from a string. Substring operations which have a known length and span over several data fields need to be modified to take into account the changed lengths of the data fields. Pack and unpack operations translate to simple assignments, since the packed field has already been converted to the corresponding unpacked field. As each operation is updated, the data fields are renamed so that we know which operations have been processed. Calls to known utility modules are also converted (see below).

The method for translating utility module calls is a three stage process:

(1) Determine the type, offset and length for each field in each parameter.

(2) Apply any data translations to the parameters to convert to the new data model. Use dataflow analysis to propagate known data types through the code. This is the same process that is applied to file records as they are converted to the new data model.

(3) Apply pattern matching and replacement to the calls to replace them with the equivalent new code or calls to new utilities. For example: after translating the parameters, a call to a simple type translation utility which expands packed hex or decimal digits to a string is replaced by a simple assignment (since after data translation, both the source and destination contain string data). This pattern matching and replacement will also deal with re-ordering parameters, adding new parameters etc.

Three typical utility functions will now be described:

—Module `U01` takes two parameters: a source and a length. It will set a return code in register `R15` depending on whether the input data field consists of valid packed decimal digits (i.e. no nybble in the input data contains any hex digit A–F).
When the translation engine sees a call to this utility it knows that the first parameter contains packed data. This will be expanded to a string, after which the utility call can be removed and replaced by equivalent WSL code:

if numeric?(source$[1\,..\,2*\text{length}]$)
  then $r_{15} := 0$
  else $r_{15} := r$ **fi**

The test for numeric characters is a native operation in COBOL. Note that the length parameter is doubled since the packed string has been translated to a character string which is twice as long.
Further transformations can merge this **if** statement with a subsequent test of r_{15} to eliminate the register assignment.

—Module `U02` takes 3 parameters: input, length and output. Each byte of the input hexfield is expanded to two bytes in the output.
On finding a call to `U02`, we know that the first parameter is a hexfield whose length is given by the second parameter, while the third parameter is a character field twice as long.

After translating the parameters, and any related data, a pattern match and replacement operation converts the utility call to a simple assignment.

—Module `U03` takes a single parameter which should be a four byte data area. It returns with the current date stored in the parameter as eight packed digits in the form YYYYMMDD.
This will be translated to a call to the new COBOL date module, which returns an ISO 8601 date as a ten byte field in the form YYYY-MM-DD.

Any known information about data types (eg as defined in the documentation) is provided to the transformation process in the form of a text file, called the *master translation file*, in the following format:

```
Asm-Name  New-Type  Type  COBOL-Name
```

The fields are space-separated to make the file human readable.

—`Asm-Name` The assembler data name. The translation engine can look up the declared assembler type and length of the data from the assembler name.

—`New-Type` Shows the type and length of the COBOL data which the , for example `X(10)` for a ten byte string.

—`Type` This field indicates the type of translation (if any) required. The possible values include:
  —`obsolete` This field will not be needed in the new system, so has no COBOL name;
  —`code` This one byte field is a country code which will expand to a four character field;
  —`string` This field is a string which will be unchanged;
  —`packed` A packed decimal field which is also unchanged, since COBOL has native support for packed decimal data;
  —`digits` A packed digit string which will expand to a character string twice as long;
  —`hex` A packed hex digit string which also expands to a character string;
  —`DDMMCCYY` A date field in the given format. All date fields will be converted to ISO 8601 standard date ten byte date fields in the form YYYY-MM-DD.

—`COBOL-Name` This will typically include a suffix: when type inferencing determines that a new assembler variable is equivalent to one already in the table, then the new variable will be allocated a new COBOL name consisting of the assembler name plus the suffix from the COBOL name of the related variable.

This information is updated as the transformation progresses and a new file written at the end of the process which includes all information.

After checking the draft master translation file, and adding new data names, and checking the output of the pointer dereferencing analysis, this program will apply the data translations given in the master translation file to the module, generating new code which uses the new data model.

7. CONCLUSION

In this paper we have outlined a method for translating a program from one data representation to a different data representation by a process of abstraction and refinement. A major application of this method is in migrating from assembler to COBOL where it is desired to use native COBOL 74 data representations wherever possible: eliminating packed hex

and decimal digits, bit operations, non-standard date fields and so on. A case study involving 8,991 assembler modules demonstrates that the method can be used with commercial assembler systems.

Complete automation of the process is not always possible, due to cases where data is accessed via pointers and it cannot be determined from the module source code alone what type of data is being addressed. Many pointer references can be resolved by static analysis, with the remainder requiring some manual intervention.

ACKNOWLEDGMENTS

This research was funded by Software Migrations Ltd.

References

[1] H. Partsch & P. Pepper, "A Family of Rules for Recursion Removal," *Inform. Process. Lett.* 5 #6 (1976), 174–177.

[2] M. Ward, "Recursion Removal/Introduction by Formal Transformation: An Aid to Program Development and Program Comprehension," *Comput. J.* 42 #8 (1999), 650–673, ⟨http://www.cse.dmu.ac.uk/∼mward/martin/papers/recursion-t.ps.gz⟩ doi:10.1093/comjnl/42.8.650.

[3] M. Ward, "Mass Migration from Assembler to C—A Case Study," Technical Report, 1999.

[4] M. Ward, "Derivation of Data Intensive Algorithms by Formal Transformation," *IEEE Trans. Software Eng.* 22 #9 (Sept., 1996), 665–686, ⟨http://www.cse.dmu.ac.uk/∼mward/martin/papers/sw-alg.ps.gz⟩ doi:doi.ieeecomputersociety.org/10.1109/32.541437.

[5] Martin Ward, "Pigs from Sausages? Reengineering from Assembler to C via FermaT Transformations," *Science of Computer Programming, Special Issue on Program Transformation* 52 #1–3 (2004), 213–255, ⟨http://www.cse.dmu.ac.uk/∼mward/martin/papers/migration-t.ps.gz⟩ doi:dx.doi.org/10.1016/j.scico.2004.03.007.

[6] Martin Ward & Hussein Zedan, "Transformational Programming and the Derivation of Algorithms," *25th International Symposium on Computer and Information Sciences, 22nd–24th September*, London (2010).

[7] Martin Ward & Hussein Zedan, "Deriving a Slicing Algorithm via FermaT Transformations," *IEEE Trans. Software Eng.*, IEEE computer Society Digital Library (Jan., 2010), ⟨http://www.cse.dmu.ac.uk/∼mward/martin/papers/derivation2-a4-t.pdf⟩ doi:doi.ieeecomputersociety.org/10.1109/TSE.2010.13.

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Virtual Education Space

S. Stoyanov, V. Valkanova, A. Stoyanova-Doycheva, E. Doychev, V. Valkanov, K. Gramova
Plovdiv University, Bulgaria

Distributed eLearning Centre (DeLC) developed in the Faculty of Mathematics and Informatics aims at delivery of electronic education services and teaching content, personalized and customized for each individual user. Virtual Education Space (VES) is a successor to DeLC. In this paper, general characteristic and the architecture of VES are presented. Problems related to the implementation and modelling of the space are considered as well.

Categories and Subject Descriptors: []:

General Terms: [], []
Additional Key Words and Phrases: eLearning, Intelligent Agents, BDI Architecture, Personal Assistant, Virtual Education Space, DeLC

1. INTRODUCTION

In recent years the interest towards electronic education has been growing stronger. As a result of that many universities have developed and implemented their own systems for electronic and long-distance education. In line with this trend a Distributed eLearning Centre (DeLC) project was implemented in the Faculty of Mathematics and Informatics aiming at the development of an infrastructure for context-aware delivery of electronic education services and teaching content, personalized and customized for each individual user [1], [2].

DeLC is a reference architecture, supporting a reactive, proactive and personalized provision of education services and electronic content. The DeLC architecture is modeled as a network which consists of separate nodes, called eLearning Nodes. Nodes model real units (laboratories, departments, faculties, colleges, and universities), which offer a complete or partial educational cycle. Each eLearning Node is an autonomous host of a set of electronic services. The configuration of the network edges is such as to enable the access, incorporation, use and integration of electronic services located on the different eLearning Nodes. The eLearning Nodes can be isolated or integrated in more complex virtual structures, called clusters. Remote eService activation and integration is possible only within

Correspondence address: Plovdiv University, Plovdiv, Bulgaria

a cluster. In the network model we can easily create new clusters, reorganize or remove existing clusters (the reorganization is done on a virtual level, it does not affect the real organization). For example, the reorganization of an existing cluster can be made not by removing a node but by denying the access to the offered by it services. The reorganization does not disturb the function of other nodes (as nodes are autonomous self-sufficient educational units providing one or more integral educational services).

Wired and wireless access to the eLearning services and teaching contend have been implemented in DeLC. The current DeLC infrastructure consists of two separate education clusters [3]. The first one, known as MyDeLC, delivers educational services and teaching content through an educational portal. The second one provides mobile access to services and content over an extended local network called InfoStations.

VES is developed as a successor to DeLC accounting for two important tendencies in the development of Internet and Web. The broad usage of the Internet and its steady transformation into a network of objects [4], as well as the globalization of cyberspace, are a foundation for the rapid development of cyber-physical social systems which will lead to essential technological, economical and sociological consequences in the following years. The phrase cyber-physical systems is used to specify a tight integration and coordination between computational and material resources where a compact integration between calculation, communication and control exists as well as interaction with the environment in which they are situated [5]. For a number of applicational areas it is necessary to measure the presence of the human and social dimensions within those spaces. Have we reached the point where the social and human dynamics become an indelible part of the cyber-physical space so that the inclusion of the term "social" is completely reasonable. One logical consequence is the notion of Pervasive Intelligent Spaces, where humans and objects interact intelligently among themselves in a way known as anywhere-anytime-anyhow. The spaces become intelligent when they are capable of monitoring what is happening inside them, have the ability to model their own behavior and to operate on the basis of their own decisions as well as interact with the inhabiting communities. Apparently, besides residents, these spaces require the buildup of a suitable information infrastructure.

Those types of spaces can introduce new approaches and scenarios for solving complex problems in the field of electronic education. An important tendency in electronic education is that it is based on the integrated character of the highly technological world that people live and study in. A basic priority is the development of education spaces via integrating different technologies which engage the students and increase the interest towards the learning process in ways that were heretofore impossible; create new opportunities for education and teaching, improve and broaden the interaction with local and global communities [6]. Education spaces, both physical and virtual, are planning environments where different forms of integrated education are performed. They relate the school, home and educational community by increasing and supporting a flexible education outside the boundaries of the school buildings and outside the usual school days. Those spaces can also direct the making of strategic decisions in schools, government and educational institutions.

A second important tendency that appears with the idea of transforming DeLC into a virtual education space is the creation of semantic web. The semantic web is a development of the current syntactic web where the infrastructure delivers a model of machine-

comprehensible data. The data is stored separately but can easily be integrated if necessary. The idea for semantic web was first introduced by Tim Berners-Lee et al in [7] [8]. Ideas for the usage of semantic web in the electronic education are presented in [9]. At the current moment using the capabilities of semantic web in electronic education is the subject of strong scientific interest.

The first ideas for the virtual education space (VES) are presented in [10]. A detailed characteristic of the space is given in [11]. In this publication we make a general characteristic and present the architecture of the Virtual Education Space. We also discuss problems related to the implementation and modelling of the space.

2. VES OVERVIEW

2.1 Basic characteristics of the VES

VES is an intelligent space. In accordance with [12] [13] an intelligent space is a environment that can continuously monitor what are happening in it, can communicate with their inhabitants and neighbourhoods, can make related inference and decisions and act on these decisions. In comparison to DeLC, an intelligent education space will support more effectively the process of blended learning, integrating electronic forms of education with the real learning process.

VES is context-aware. According to [14] context is all information which can be used to characterize the situation of an identity. By "identity" we can designate a man, a place or an object which are viewed as meaningful for the interaction between user and application, which includes themselves. In accordance to the definition of context, Dei defines that a system is context-dependent if it uses contexts to deliver significant information and/or services, and the importance depends on the user's tasks [15] [14]. In our case context-dependency is the ability of a system to find, identify and interpret the changes (events) in its environment and depending on their nature to undertake compensating actions. The main compensating actions (attributes for context-dependency) are personalization and adaptation. Personalization is the system's ability to adapt to individual features, desires, intentions, goals of the users. Adaptation is the system's ability to adapt to the remaining context features such as area of knowledge, school subject, types of devices used by the end-users.

VES is scenario-oriented. From user's point of view, the space is a set of separate e-learning services and educational scenarios provided for the use through education portal DeLC or personal assistants. Scenarios are implemented by corresponding workflows rendering an account of the environment's state. Thus it is possible to take into account various temporal characteristics (duration, repetition, frequency, start, end) of the educational process or events (planned or accidental) which can impede or alter the running of the current educational scenario. To deal with emergencies (earthquake, flood, fire) there are defined emergency scenarios which are executed with the highest priority.

VES is a controlled infrastructure. Access to the space's information resources is only possible through the so-called "entry points." The personal assistants operate as typical entry points while the education portal of DeLC is a specialized entry point; a user has to be in possession of a personal assistant or to use the portal to be able to work in the space.

2.2 VES Architecture

The VES architecture contains different types of components. Assistants play an important role in the space. Three types of assistants are supported in the space (1). The personal assistants have to perform two main functions providing the needed "entry points" of the space (Fig.1, first orbit). First, they operate as an interface between their owners and the space and if necessary, carry out activities related to personalization and adaptation. Secondly, they interact with other assistants in the space in order to start and control the execution of the generated plans. In certain cases they operate operate as an intermediary for activation of scenarios or services. The personal assistants will be usually deployed over users' mobile devices. The specialized assistants are usually located on the server nodes of the VES, known as operatives (Fig.1, third orbit). They support the execution of the plans generated by the personal assistants; therefore they implement suitable interfaces to the available electronic services and data repositories. Operatives serve two subspaces, known as DiLibs-Subspace and Admin-Subspace respectively. Guards (Fig.1, second orbit) are special assistants which are responsible for safety and the efficient execution of the plans in the space. These are usually intelligent devices that react to various physical quantities in the environment, e.g. smoke, temperature, humidity. The guards act as an interface between the physical and the virtual world in the space.

2.3 Assistants as Intelligent Agents

VES is "populated" only by active components known as assistants. Each assistant has to play a role in accordance with its delegated responsibilities. The responsibilities (tasks) can be implemented or delivered by external electronic services. The electronic services themselves cannot be separate operational components of the space because they are suitable for implementation of business functionality but are static without having the properties to be context-aware and intelligent. The assistants, implemented as intelligent software agents, constitute the kernel of the space. The assistants are active context-aware intelligent components that support the planning, organization and implementation of the educational process. The basic features of the intelligent assistants are:

—Autonomy - they operate without the direct intervention of humans or other agents, they have control over own actions and internal states;

—Reactivity - they perceive their environment maintaining continuous contact with this environment and respond to changes that occur in their environment;

—Proactivity - they not only act in response to their environment. They are able to be proactive exhibiting a goal-driven behavior; "

—Sociality - they are able to interact and cooperate with other assistants via some communication language (e.g. ACL [16]).

Furthermore, the space assistants are implementing as limited rational agents which means the following:

—They have only partly control over the space;

—Operate with limited capacity (resources) on the planning, forecasting, selection and execution of actions;

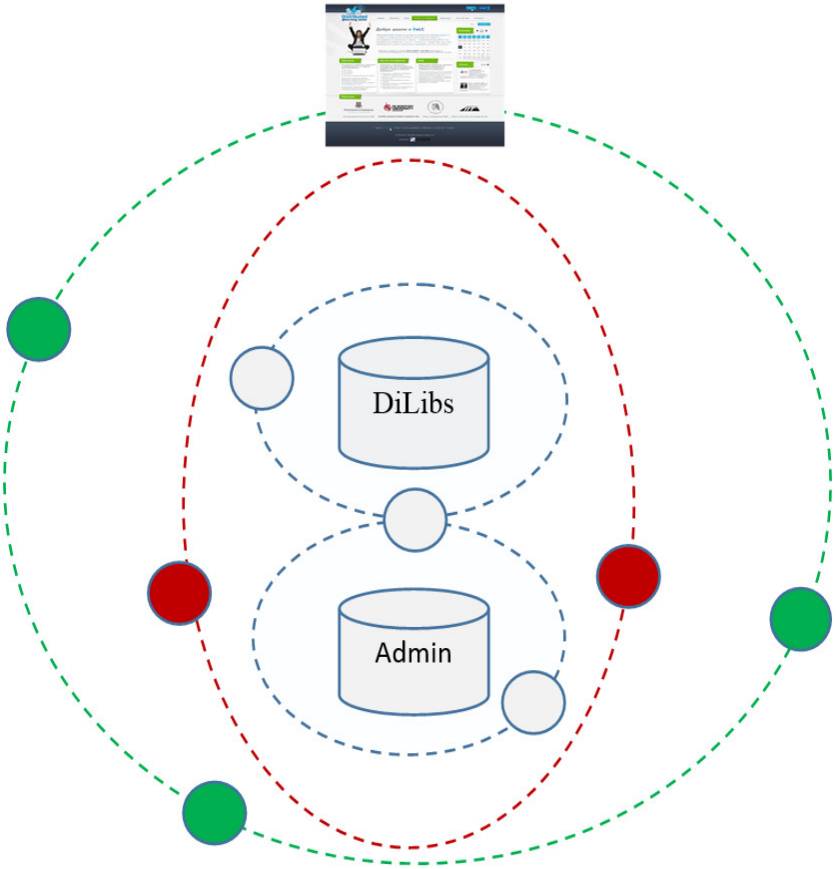


Fig. 1: VES Architecture

—In accordance with the task completed they be able to assess their behavior and based on these evaluations to measure the effectiveness of their actions.

Significantly in the "bounded rationality" is finding meaningful solutions for efficient use of available resources. The scope of the action of a rational agent can be defined as the optimal success according to its current knowledge and skills, limited resources and limited time. A model known as "practical considerations" can be applied for decision-making by rational agents. A rational agent holds a practical consideration performing two steps:

—Deliberation - at this stage, the agent decides what to do (what purpose it wants to achieve) using its mental states;

—Planning (means-ends reasoning) - at this stage, the agent decides how to meet the goal.

Operation of a rational agent is presented by help of "sense-think-act" life cycle where deliberation and planning is fitted with sensors and effectors. BDI (Belief-Desire-Intention) architecture [17] presented a human-like model for bounded rationality is usually used for

development of rational agents. In the deliberation step three mental states of an agent play an important role:

—Beliefs represent perceptions of the agents about their environment;

—Desires represent agents' wishes or tasks that have not been converted into intentions;

—Intentions are committed desires.

In the means-ends step an agent outlines the process for addressing how to achieve a goal using existing resources (also known as planning). The planning module of the agent takes as input a structure consisting of:

—Goal, i.e. the current intention of the agent;

—Current state of the environment, i.e. the beliefs of the agent;

—Possible actions of the agent.

The result is an action plan for achievement the goal.

3. PERSONAL ASSISTANTS

3.1 Sense-Think-Act Cycle

The personal assistants operating in the space are implementing as intelligent agents with bounded rationality whose operation cycle is given in Fig. 2. During the deliberation a personal assistant tries to determine its current goal (intention) with the help of the three mental states (beliefs, desires, intention).

3.2 Events

The unified presentation of the mental states and relations between them will ease the development of a general model of an agent's deliberation. In the suggested architecture for a personal assistant the events perform this function.
An event is a basic concept within the VES. However, a universal definition has not been reached, as multiple theories exist concerning events; e.g. in computing, an event is an action or occurrence detected by the program that may be handled by the program [18]; in philosophy, events are objects in time or instantiations properties in objects [19]; in probability theory, an event is a set of outcomes of an experiment to which a probability is assigned [20]; an event in the Unified Modeling Language (UML) is a notable occurrence at a particular point in time [21]. In the space, events take place as in the real world as well as in the virtual one bridging the both. However, all these events have to be represented in the virtual world (assistants) to make them amenable to usage from the assistants. An event can be presented as an atomic identity which can only be identified but not attributed, or the event can be presented as an identity which can be attributed by allowing the identification and working with the different attributes. In the space we accepted the model given in [22]. The presented model is a six-faceted model that aims at establishing a common foundation for a wide diversity of applications and addresses several elementary aspects of event description.
These aspects are briefly characterized as follows [22]:

—Structural - events are a modeling concept applicabale at many different levels of abstraction. Events can be combined in many different ways to define other events, which

```
repeat
   deliberation {
        current_belief ← sense(environment);
        beliefs ← update(beliefs, current_belief);
        current_intention ← generate_goal(PC, beliefs);
   }
   means-ends {
        switch (current_intention) {
         case 1: activate_edu_scenario();
         case 2: activate_service();
         case 3: update(PC);
         /* more cases
         default: run_edu_scenario();
   }
```

Fig. 2: PA's sense-think-act cycle

can also be related to other events, thus creating complex structures and hierarchies of events [23];

—Informational - an event model should provide information about the events that occur (e.g. event type) and should support a taxonomical organization of event types. The way for creating complex events and hierarchies is determined by the area of application;

—Temporal - the temporal characteristics of the events have to be taken into account because the events are inherently related to the concept of time;

—Spatial - a common event model should also show location awareness and support different ways of capturing the spatial aspect in an event's description;

—Causal - in many applications the users are interested in the chain of events;

—Experiential - various applications offer users engaging ways of exploring and experiencing a course of events to let them gain insights into how the events evolved.

3.3 PA's Deliberation Phase

In our deliberation model the three mental states (beliefs, desires, intentions) are presented as events. An event is defined at the level of the applicable area and is loaded with the semantics of terms from that area; thus, events are holding a lecture, participating in a seminar, holding an examination, exercises. Those events can be characterized through different attributes. In the deliberation phase a central role is played by desires, presented as a personal calendar of the user (PC ABC.2.). Thus for instance, the personal calendar of a student would include all activities that he has to perform in order to successfully complete the school year. Those can include attending lectures, participating in seminars, taking examinations. All those activities are viewed as potential events which can occur in the future. The role of beliefs is to determine definitively the actual access to the personal calendar. The goal of the actual access is to identify a desire (or a number of desires) from the calendar, a candidate for becoming the agent's actual goal. Afterwards, this desire is transformed into an actual goal (generate_goal() in Fig.2.). In accordance to the actual beliefs there are two types of desires: ones that can be transformed into intentions and ones who only remain desires (e.g. an absence from a lecture, seminar or an examination). In our model there are two types of basic beliefs; they are the current date and current time. Independently from the type of the event which was perceived by the agent, they play an important role in the realization of the actual access to PC. Agent's deliberation phase is completed with generation of current intention.

3.4 PA's Planing Phase

After the actual intention has been determined during the deliberation phase, the PA has to make a plan for its achievement. In our model the planning is performed in dependence with the type of the goal (Fig. 2). Usually, the PA requires the execution of a certain educational scenario (run_edu_scenario()) by generating a query to the operative agents of the digital libraries. In certain cases (see p. 5.1.), the activation of electronic services or educational scenarios is achieved through the PA's intermediation (activate_service(), activate_edu_scenario()). There are instances where a PC update is required (update(PC)); such an example is given in p. 5.3.

4. OPERATIVES

Operatives are the second large group of active components in the space. They are usually situated on the space's servers. Operatives (realized as intelligent agents) are not suitable software components for delivering business-functionality. The functionality is realized as electronic services which in their nature are static and cannot operate as separate components in the space. For that reason a suitable interface operative is supplied for each service, which makes the respective service an active component by allowing it to interact with other components in the space. In accordance with the functions that the operatives perform, the space can be viewed as containing two subspaces called DiLib-Subspace and Admin-Subspace.

4.1 DiLibs-Subspace

The main goal of the operatives functioning in the DiLib-Subspace, is to secure the execution of the educational scenarios which are directly related to the education process. When an educational scenario has to be realized, the necessary operatives secure the suitable work stream. The operatives of that subspace realize interfaces to three components

supplied to the space by DeLC - SCORM 2004 Engine, Test Engine and Event Engine. The operatives' environment in that subspace is mainly comprised of digital libraries. The digital libraries are a specialized storage where mainly educational content is stored. Using the SCORM 2004 standards (for presenting educational content) and QTI 2.1. require certain structure of the digital libraries. In them they store also specifications for the educational scenarios. The digital libraries are created and actualized mainly with the help of specialized editors delivered to the users by the digital interface. When necessary, their content can be dynamically actualized by the serving operatives.

4.2 Admin-Subspace

Admin-Subspace secures all activities related to the organization, control and documentation of the education process. In the administrated database is stored all the necessary useful information for planning, organizing, protocoling and documenting the educational process such as school plans, programs and schedules, protocols from examinations, gradebooks. One example for organization of the school process is the preparation of the students' personal assistants. What is significant here is the creation of individual student calendars (desires of the personal assistants serving them). The individual calendars are generated automatically by the school curriculum for a school term or a school year. From the school curriculum certain events can be derived which are related to the student's immediate learning activity such as lectures, seminars, examinations. Afterwards, the calendar can be completed with events of extracurricular character such as a birthday party, meeting friends, hobbies. Two operatives play an important role in that subspace - the electronic gradebook and the teacher's notebook.

4.3 Interaction between the two subspaces

The two subspaces interact intensively for securing blended learning. An example for such an interaction is the scenario for protocoling the students' electronic testing (Fig. 3). The student's personal assistant (PAS), interacting with the teacher's personal assistant (PAT), determines for execution an educational scenario for electronic testing. Catalog Operative (CO), operating within the DiLibs-Subspace, analyzes the scenario and determines the necessary work flow, i.e. the Interface Operative (IO) and Content Access Operative (CAO) agents serving the scenario's execution. Since the activation of the Test Engine is realized through the client layer of the space, the PAS is used for that goal. The results from the test are written in the student's book of the tested student which causes a change in the environment and the activation of the Student Book Operative (SBO). Depending on the results (e.g. a failed test), the SBO informs the PAT which in turn requires from the PAS an actualization of the student's PC (requesting a date for re-taking the examination and on confirmation, storing it in the PC).

4.4 Operatives' Environment

Context-Awareness and intelligent behavior cannot be achieved only by the presence of intelligent agents. These have to be also supported in the agents' environment, i.e. intelligently structured data is also required. Under "intelligent" we understand suitably structured data which can be stored separately and when necessary (executing complex queries), easily integrated [24]. The environment of the assistants operating in the DiLIbs-Subspace, is built in relation to the requirements of the standards SCORM 2004 and QTI

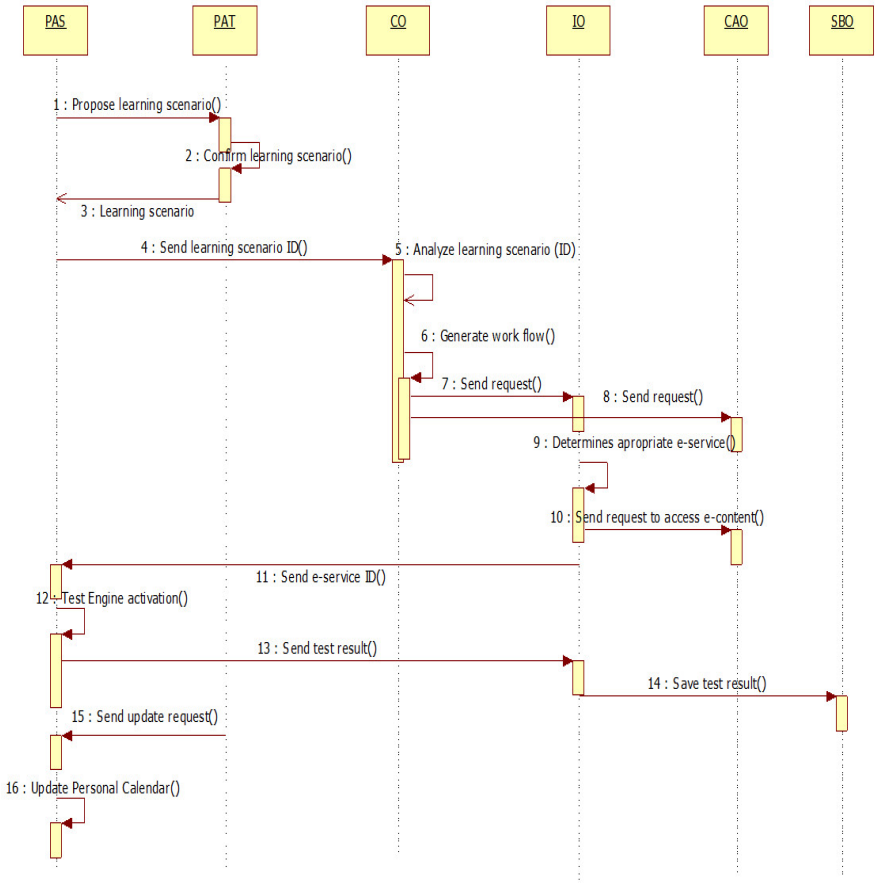


Fig. 3: DiLibs - Admin Interaction

2.1. for structuring the necessary repositories. The administrative information in the Admin-Subspace is stored in relational databases. A flaw of these repositories and data bases is the absence of semantics. For that reason in the assistants' environment different educational ontologies are developed. Using semantic technologies we intend to achieve the following benefits:

—Education focused on the student - the teaching material, possibly from different authors, can be related to commonly agreed ontologies. Personalized courses can be developed through semantic queires. The teaching material can be sought and derived from the context of actual problems in accordance with the decisions of the learner;

—Flexible access - the knowledge can be accessible in any order that the learner desires. The circumstances will be determined by suitable semantic annotations. Support for non-linear access;

—Integration - An integrated platform for the business processes of the organizations. The educational activities can be integrated in those processes.

The ontologies are also an effective mechanism for realizing a shared comprehension. In electronic education it is suitable to build the following storages for knowledge (ontologies) [25]:

—Ontology of the teaching content - containing basic concepts of the area in which the education is performed, and the relations between them;

—Pedagogical ontologies - for solving pedagogical problems related to the comprehension of the teaching material. For instance, the teaching material can be classified as: lessons, exercises, illustration examples, solutions for problems with a high degree of difficulty;

—Structural ontologies - define the logical structure of the teaching material and typical knowledge of that type include hierarchical and navigational relations.

5. VES IMPLEMENTATION

The space is built in the following stages:

—Transforming DeLC;

—Developing the space's assistants;

—Experimenting and modeling.

5.1 DeLC Transformation

DeLC environment is service-oriented and multi-layered, consisting of three logical layers: user interface, e-services and digital libraries. The user interface supports the connection between the users and the portal. Through it the users can register in the system and create their own personalized educational environment. The user interface visualizes and provides access for the user to services, depending on their role, assigned during the registration. Three components called 'engines' are located in the middle layer that are transparent for the users. Using the information, contained in the meta-objects, they can effectively support the activation, execution and finalization of the eLearning services. SCORM 2004 Engine is implemented for delivering an interpreter of the electronic content, developed in accordance with the SCORM 2004 standard. The Test Engine assists in performing electronic testing using the DeLC environment. The Event Engine supports a model for event management, enabling the users to see and create events and also be notified for them in advance. The events in the system reflect important moments for the users, such as a lecture, examination, test, national holiday, birthday, etc. One event is characterized by attributes, such as a name, start and end date and time, details, and information if it is a recurring one, as well as rules for its recurrence. The Event Engine supports yearly, monthly and weekly recurring. The third layer contains electronic content in the form of repositories, known as digital libraries. Test Engine is the most-often used service in the real teaching process in the Faculty of Mathematics and Informatics in the Plovdiv University. For that reason we started the transformation of DeLC from this component. The main goal of the represented architecture (Fig.4) is implementing an Test Engine as a part of Virtual Education Space. The access to the functionalities of the engine is provided by web services for managing, assessing and analyzing digital assessment content (Web Services Layer). These services are in direct communication with a multi-agent content and assessment management system (Assessment Agent System) that is responsible for providing the digital information resources requested by users via the web services from the service layer of the Test Engine. The Assessment Agent System manages the digital content by using the provided

REST web services for database access to acquire it and cooperative work between the existing agents for its processing. The digital assessment content that the engine manages is designed as a fully compliant with the QTI 2.1 final specification [26]. Defining the architecture and its provided services in such a way facilitates a wide range of opportunities for integrating the proposed Test Engine in the space.

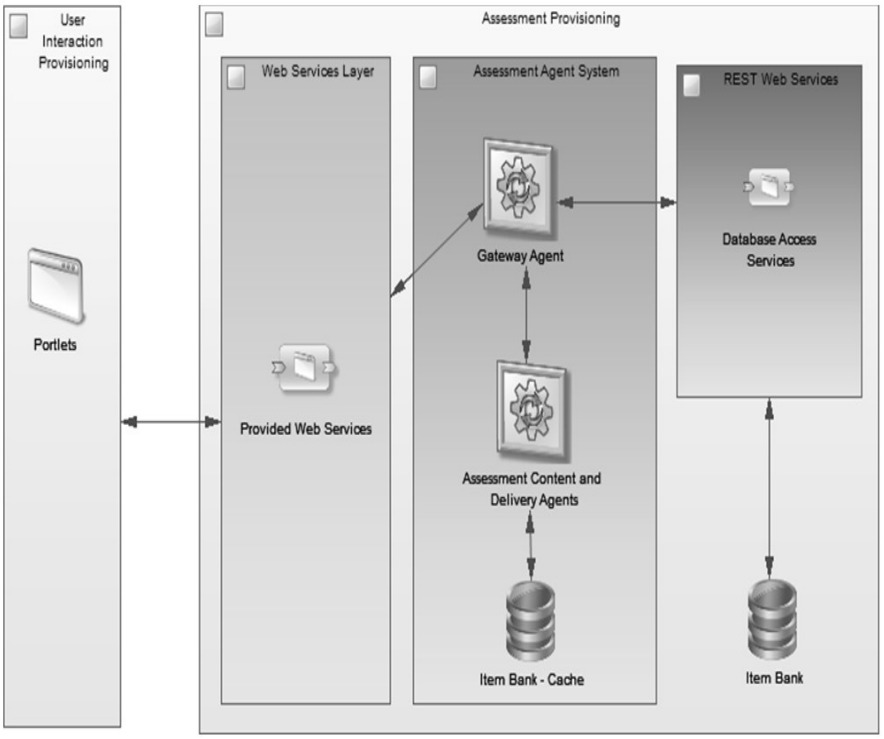


Fig. 4: Test Engine Architecture

The Service Layer of the represented architecture is a set of the provided web services that act as a mediator between the User Interaction Provisioning modules (portlets) and the Assessment Agent System. Due to the fact that the assessment content implementation is fully compliant with the information model defined by the QTI specification, the set of functionalities implemented as services that can be performed over and with the data are well defined and can be classified in three main groups - Delivery Services responsible for delivering the assessment content, Reporting Services for outcome data management and Storage Services for basic data transfer in and out of the eTesting system. The core of the Assessment Provisioning implementation is the Assessment agent System. It is the actual eTesting realization as a multi-agent system that provides the management and implementation of the digital assessment content and processes in an intelligent and adaptive fashion. The service layer acts as a user accessible wrapper of this core that

makes the processes of assessment and content management transparent to the users. The two basic aspects of information management concerning eTesting - assessment content and assessment processes management - distinguish two groups of agents in the assessment multi-agent system to perform the specific operations defined by the concrete aspect of management - Content Management Agents (CMA) and Assessment Delivery Agents (ADA). The key concept that makes it possible for the assessment agent system to communicate with the provided web services in the proposed eTesting system is the adoption of the Web Services Integration Gateway (WSIG) architecture [27] that defines one agent (Gateway Agent - GA) that mediates the communication between externally provided to the multi-agent system web services (Service Layer) and all other agents on the platform that wish to exchange data with these services (CMA, ADA). The eTesting functionalities are implemented as goals set by the GA when a service request from the Service Layer is received or autonomously generated by the GA or other agents. These goals are achieved by the deliberate actions of a single or a group of CMA or ADA as a result of their collaboration. This approach for implementing the eTesting functionalities provides flexible mechanisms for adding new functionalities to the eTesting management system that enrich the provided business logic in both functional diversity and complexity aspects. Accessing the digital content in the main storage (the Item Bank) via communication with the provided REST API enhances data caching in the Item Bank - Cache ensuring fast content management and manipulation. The component that ensures decoupling between the implementation of the assessment management logic and the digital data management is the REST Web Services Layer. It provides a REST API [28] for database access for acquiring and manipulating the needed digital assessment content. Using the provided REST mechanisms for caching and data delivery this component ensures a full quick access to all the needed information while leaving the assessment processes logic implementation intact.

5.2 Assistants Development

The first stage aims towards building the kernel of the space which includes the transformed environment of DeLC, which was discussed in the previous paragraph. SCORM 2004 Engine, the new version of Test Engine and Event Engine make the server part of that kernel. The client part includes only one entry point - the DeLC portal. The goal of the second stage is expanding the kernel with the following new components:

—Personal Assistants and Operatives;

—Game-Oriented Learning;

—eLearning Ontologies.

Game-Oriented Learning and eLearning Ontologies are examined in two publications [29] [30]. Here we will look at several aspects related to the development of PA. Since PA is individual, it cannot be programmed from the very beginning for each user. We need a generic architecture from which instances of individual PA can be configured, whose creation is a serious challenge. There are several options with regard to a generic PA's architecture:

—The agent is entirely located on the server - the advantages of this approach are that there will be no need to migrate the agent and/or its states between the different mobile devices which the user would potentially use; high efficiency in processing complex tasks or such associated with processing large amounts of data - only the results will be

sent to the mobile device. The disadvantages are the increased amount of traffic between the client device and the server (even for the performance of simple tasks), which in turn can involve potential security problems and jeopardize the transfer of information;

—The agent is entirely located on the client device - this approach has several drawbacks: such a system would take a substantial part of the relatively limited resources of the mobile devices; migration of the agent states across mobile devices will be necessary, which is a considerably complex and difficult task;

—The agent is distributed between the client device and the server side - this approach allows an efficient distribution of the tasks that the agent has to perform as well as if there is a need of some kind of agent migration - it will be in a minimum extent. This architecture also significantly reduces the traffic between the client and the server and allows greater flexibility and scalability of the application.

For that reason we pay attention to the distributed architecture, which is further determined by JADE-LEAP (Lightweight Extensible Agent Platform) selected technology for implementation of the personal assistants. This is an extension to the basic JADE specifically designed for mobile devices with limited resources [31]. The choice of this technology is motivated by many reasons, some of which are the following:

—The platform is developed entirely in Java, which coincides with the language used for the development of VES and the Android operating system;

—Facilitates the development of multi-agent systems via an intermediate layer that is compatible with the specifications of FIPA [32];

—Provides tools to facilitate debugging and deployment and there is a transparent communication mechanism between the agents through ACL messages;

—JADE supports the use of many additional libraries, one of which provides the important to us BDI architecture, through which we can implement the mental states of intelligent agents, namely autonomy, social ability, reactivity, proactivity and rationality.

In addition to JADE-LEAP to ensure dealing with the mental states of the agents we are integrating JADEX environment [33]. The JADEX reasoning engine follows the BDI model and facilitates easy intelligent agent construction with sound software engineering foundations. It allows agents' programming in XML and Java and can be deployed on different kinds of middleware such as JADE. The platform allows mixing of agents with different architecture in one application. JADEX provides also a framework for developing software agents running on the Android platform. The personal assistants, responsible for interaction with users should provide a convenient and intuitive interface. This is a reason why we have chosen the Android platform. Furthermore, at the present, Android is the most popular mobile platform in the world and supports a great number of different mobile devices. In addition, the Android provides a very powerful development framework for building applications for mobile devices. It automatically adapts the user interface to look its best on every device, while giving the developer as much control as he wants over the whole user interface on different device types [34]. There are convenient tools for developers which offer a full Java IDE with advanced features for developing, debugging and packaging Android applications. The environment offers an emulator of virtual devices that emulates any hardware configuration. The platform uses Java and is open source, which allows working with multiple, external, written by third-party libraries. The Android philosophy is to support an extremely high level of user customization.

5.3 VES Modeling

During the third stage we predict the development of models of the space and experimenting with them. With the help of those models we aim at:

—Checking and studying in-depth the expected behavior of the assistants and the interactions among them;

—Studying and understanding the temporal dependencies in the space;

—Supporting the formalization of structures in the space.

Our goal is to develop a united technological environment for modeling which includes three formal systems (CCA, CS-Flow, Tempura), which will support both the modeling of the entire space and its separate components. Context-awareness requires applications to be able to adapt themselves to the environment in which they are being used such as user, location, nearby people and devices, and user's social situations. The Calculus of Context-aware Ambients (CCA) [35] has been proposed to model context-aware systems. The Calculus of Context-aware Ambients (CCA) is a process calculus based on the notion of ambients. CCA is used to model ambients in terms of process, location and capability. In [36] are demonstrated the capabilities of the calculus for modeling the mLecture, mTest and mTutorial services presented by DeLC. The context-aware process management in the Virtual Education Space is a significant problem to be solved. We believe a scenario-oriented management is appropriate where various education scenarios can be activated depending on the specific situation in the space. At the same time we are looking for opportunities for formal specification of these scenarios. Contex Secure Flow (short CS-Flow) is a formal interpretable language for presenting of context-aware workflow [37]. Recently, we are going to examine the language use in VES [11].

The most important characteristic of the scenarios is that they are time-dependent, therefore we need a proper formalism to describe them and an interpreting mechanism to correspond with our system. A good approach for developing the required interpreting mechanism is to look for a formal notation which allows on one hand to specify the scenarios with their temporal characteristics and on the other hand allows the existence of a possibility for a program interpretation of those specifications. Usually formal logic-based models are suitable in such situations. Apparently, in this case a formal notation using first order logic is inapplicable, according to time-critical specifics of the system. Thus, the formal notation Interval Temporal Logic (ITL) was chosen for the following reasons [38]:

—ITL is a time-dependent logic allowing the presentation and control of linear and parallel processes. A basic characteristic of this logic is considering time as a discrete sequence of points in time called intervals. Losing the concept of time as an endless notion we can describe different processes in a way similar to the operation of modern computers and software systems;

—For ITL there is an interpreting mechanism and its program realization called Tempura.

In [39] is presented the AjTempura, the new version of the Tempura interpreter which is agent-oriented and written in Java. This interpreter can work in a united technological environment for modeling of the space.

6. CONCLUSION

In this paper, the Virtual Education Space that is an intelligent, context-aware, scenario-oriented and controlled infrastructure is presented. Now, our efforts are directed towards

the implementation of personal assistants. A huge challenge is how do we achieve personalization? Therefore, we are going to develop a detailed generic assistant's architecture that can be adapt to specific users.

REFERENCES

[1] S. e. a. Stoyanov, "From cbt to e-learning," Journal Information Technologies and Control, 3(4), 2-10, (2005).

[2] S. e. a. Stoyanov, "An approach for the development of infostation-based elearning architectures," Comptes Rendus de lÁcademie Bulgare des Sciences., 61(9), 1189-1198, 2008.

[3] E. Doychev, Environment for Provision of eLearning Services. PhD thesis.

[4] A. Kevin, "That ïnternet of things, in the real world things matter than ideas," RFID Journal,.

[5] F. Y. Wang, "The emergence of intelligent enterprises, from cps to cpss," IEEE Intelligent Systems, pp. 85-88, July/August 2010.

[6] M. Consortium, "Learning spaces framework.," Australia - New Zealand: MCEETYA Consortium, 2008.

[7] O. L. T. Berners Lee, J. Handler, "The semantic web," Scientific American, vol. 284, pp. 34-43, May 2001.

[8] T. Berners-Lee, "What the semantic web can represent," W3 org., Scientific report 2000.

[9] S. R. L. Stojanovic, S. Staab, "elearning based on the sematic web," WebNet, 2001.

[10] I. P. D. Orozova, S. Stoyanov, "Virtual education space," International Conference, Free University of Burgas, 14-15 June, 2013, 153-159, ISBN 978-954-9370-99-7.

[11] V. Valkanova, "Researching the virtual learning space in the secondary school," PhD Thesis, Sofia, 2014, ISBN 978-954-322-768-6.

[12] B. L. et. al., "Intelligent spaces: An overview," IEEE International Conference on Vehicular Electronics and Safety, 13-15 Dec. 2007, Beijing, 978-1-4244-1266-2.

[13] F.-Y. Wang, "Driving into the future with its," IEEE Intelligent System, Vol.21, No.3, 2006, pp.94-95.

[14] A. K. Dey, "Understanding and using context," Personal and Ubiquitous Computing Journal, Vol.5, No.1, pp.4-7. 2001.

[15] G. A. A.K. Dey, "Towards a better understanding of context and context-awareness," In: Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness, New York, ACM Press, 2000.

[16] "Agent communication language specifications," http://www.fipa.org/repository/aclspecs.html.

[17] M. G. A.S. Rao, "Bdi agents: from theory to practice," In: Proceedings of the 1st International Conference on Multi-Agent Systems, San Francisco, CA, 1995, 312-319.

[18] "http://en.wikipedia.org/wiki/Event_(computing).,"

[19] "http://en.wikipedia.org/wiki/Event_(philosophy).,"

[20] L.-G. Alberto, "Probability, statistics and random processes for electrical engineering," Upper Saddle River, NJ: Pearson, 2008.

[21] "http://en.wikipedia.org/wiki/Event_(UML).,"

[22] J. R. Westermann U, "Toward a common event model for multimedia applications.," IEEE MultiMedia. 14, 19-29., 2007.

[23] J. R. Rafatirad S, Gupta A, "Event composition operators: Eco.," In EiMM 2009: Proc. 1st ACM Int. Workshop on Events in Multimedia, pp. 65-72. New York, NY: ACM., 2009.

[24] J. H. D. Allemang, "Semantic web for the working ontologist," Elsevier, 2011, ISBN: 978-0-12-385965-5.

[25] "A semantic web primer," The MIT Press, 2004.

[26] "Ims question & test interoperability overview," Final v2.1, 31 August 2012, http://www.imsglobal.org/question/qtiv2p1/imsqti_oviewv2p1.html.

[27] G. D. Bellifemine F., Caire G., "Develping multi-agent systems with jade," John Wiley & Sons, Ltd, February 2007.

[28] "http://rest.elkstein.org/2008/02/what-is-rest.html, rest.elkstein.org.,"

[29] V. et. al., "Game-oriented learning in virtual education space," This issue.

[30] A. S.-D. et. al., "Digital library in virtual education space," This issue.

[31] "Jade-leap,"

[31] "Foundation for intelligent physical agents," http://www.fipa.org.

[32] "Jadex agents," http://jadex-agents.informatik.uni-hamburg.de/.

[34] "Android for developers," http://developer.android.com/about/index.html.

[35] H. Zedan F. Siewe and A. Cau., "The calculus of context-aware ambients.," Journal of Computer and System Sciences.

[36] M. H. Al-Sammarraie, "Policy-based approach for context-aware systems," PhD thesis, Software Technology Research Laboratory, De Montfort University, Leicester, UK.

[37] H. Zedan, "Cs-flow. computational model - lunguistic support," Internal Report, STRL, De Montfort University., (2012).

[38] B. Moszkowski, "Executing temporal logic programs," Cambridge University Press, Cambridge., (1986).

[39] V. Valkanov, "Context-aware management of eservice," PhD Thesis, Sofia, 2013.

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Semantically Enhanced Search

A. Rahuma

De Montfort University, Leicester, U.K.

In multimedia databases, data are images, audio, video, texts, etc. Research interests in these types of databases have increased in the last decade or so, especially with the advent of the Internet and Semantic Web. Fundamental research issues vary from unified data modelling, retrieval of data items and dynamic nature of updates.
The work presented here builds on findings in Semantic Web and retrieval techniques and explores novel tagging methods for identifying data items. Tagging systems have become popular which enable the users to add tags to Internet resources such as images, video and audio to make them more manageable. Collaborative tagging is concerned with the relationship between people and resources.
Most of these resources have metadata in machine processable format and enable users to use free- text keywords as search techniques. The limitation with such techniques includes polysemy (one word and different meaning), synonymy (different words and one meaning), different lexical forms (singular, plural, and conjugated words) and misspelling errors or alternate spellings. We introduce semantic characterization of web resources that describes the structure and organization of tagging, aiming to extend the existing Multimedia Query using similarity measures to cater for collaborative tagging. In addition, we discuss the semantic difficulties of tagging systems, suggesting improvements in their accuracies.
The scope of our work is to increase the:

—accuracy and confidence of multimedia tagging systems.

—similarity measures of images by integrating varieties of measures.

To address the first shortcoming, we use the WordNet based on a tagging system for social sharing and retrieval of images as a semantic lingual ontology resource. For the second shortcoming we use the similarity measures in different ways to recognise the multimedia tagging system.
To increase the accuracy of the tagging system we have performed different experiments on many images using similarity measures and various techniques from VoI (Value of Information).
The findings have shown the linkage/integration between similarity measures and that VoI improves searches and helps/guides a tagger in choosing the most adequate of tags.

Categories and Subject Descriptors: []:

General Terms: H.3: INFORMATION STORAGE AND RETRIEVAL:[General, Content Analysis and Indexing, Information Storage, Information Search and Retrieval, Online Information Services, Library Automation, Digital Libraries], H.5: INFORMATION INTERFACES AND PRESENTATION: [Multimedia Information Systems Hypertext/Hypermedia]

Additional Key Words and Phrases: Multimedia, Semantic Web, Similarity measures, Tagging Systems, Collaborative tagging

Correspondence address: Software Technology Research Laboratory, De Montfort University, Leicester, UK.

1. INTRODUCTION

The availability of internet-enabled devices, such as mobile phones, desktops, laptops, tablets, mini tablets, PDAs, set-top boxes and smart TVs, has facilitated the access to upload and modify content, images, video and audio media. Here are some statistics which show

—An estimated 7 Petabytes is the amount of photo content added to Facebook every day;
—By the middle of 2011, an estimated 100 billion photos were hosted;
—During 2012, 300 million new photos were added every day;
—In 2012, 4.5 million photos uploaded to Flickr each day;
—In 2011, 6 billion photos were hosted by them;
—5 billion is the total number of photos uploaded to Instagram since its start Sept 2012
—58 photos are uploaded every second.

Similar statistics can be found for other types of media - audio and textual. Therefore, the Internet has become a universal source of rich learning resources. We call this Universal Library that almost renders obsolete the traditional libraries (albeit physical or virtual).

Nonetheless, we should observe the following

—Whilst we can imposed structure on traditional libraries, the Universal Library is, on the whole, unstructured and un-organised;
—Each object in the Universal Library is tagged (including empty tags).
—The un-structured nature of the Universal Library gives its popularity and should be maintained and preserved.
—The growth of a Traditional library is limited, bounded and finite in nature, the Universal Library is almost unbounded and unlimited.
—We can argue that, from an educational theorist view point, the globalisation and the scale of the Universal Library should greatly benefit the learning process.

The inherent structure in the traditional libraries, gives it the advantages of precise searching. The inherent difficulty of the Universal Library is in its imprecise search. Therefore, the key challenge is *how can the accuracy and efficiency of search and retrieval be improved without too many restrictions?* In what follows, we will attempt to shed some light on such a challenge.

2. TAGS AND THEIR CHALLENGES

A tag is a word chosen by the subject to identify the object. Such presents many challenges, including:

2.0.0.1 *The influence of the users' culture.* : Ethnicity and cultural differences guide perception and cognition differently. For example, an analysis of image tags created by European American and Chinese participants concluded that whereas Westerners focus more on foreground objects, the Easterners have a more holistic way of viewing images early on. This was discovered through the analysis of tag assignment order. For Easterners,

the specificity of tags increased from holistic scene description to individual objects. On the other hand, the tags given by the Westerners focused on individual objects first and then on overall scene content.

2.0.0.2 *The influence of Motivation.* : Motivations, probably, forms a major influence on the usability of tags for all purposes. Tags that arise from the need of future retrieval and contribution, particularly for the benefit of external audience, are likely to be visually more relevant compared to tags used for personal references. Images that are annotated and shared within special interest groups are very likely to be specifically annotated and heavily monitored. They would also be heavily influenced by the motivation of the interest group.

2.0.0.3 *The Users' Domain knowledge.* : Some users who tag their images with non-understandable words, characters, personal references or numeric symbols, can only be thought of as doing so because they have a praticular knowledge about the domain that caused them to save and annotate the images in the first place. Such tags have no use or meaning to the wider audience, and should be filtered out, so as not to affect usage statistics.

2.0.0.4 *The issue of Semantic loss.* : An annotator in folksonomies is not obliged to associate all relevant tags with an image, leading to semantic loss in the textual descriptions. The batch-tag option provided by most photo sharing sites adds to this problem by allowing users to annotate an entire collection of photos with a set of common tags. Even if such tags are potentially useful to provide a broad personal context, they cannot be used to identify image-level differences, thus leading to semantic loss. One consequence of this fact is that the absence of a tag from an image description cannot be used to confirm the absence of the concept in that image. Hence, such images cannot be directly used as negative examples for training.

2.0.0.5 *The issue of Vocabulary.* : The spontaneous choice of words to describe the same content varies among different people, and the probability of two users using the same term is very little. Known as the vocabulary problem, this issue is often cited as a common characteristic of folksonomic annotations. The different word choices introduce problems of polysemy (one word with multiple meanings), synonymy (different words with similar meanings) and basic level variation (use of general versus specialized terms to refer to the same concept).

2.0.0.6 *Operational Challenges.* : In addition to these challenges, *collaborative tagging* imposes its own addition challenge. An *object* (or a resource) in a collaborative tagging environment, is *continually* tagged:

—with 0 or more tags (in some system, up to 75 tags are allowed per an object ,)

—by more than one *subject* (user),

—using different languages or notations (i.e. abbreviations - *lol, omg, etc.*, and that

—a tag can be removed at any time by its creator.

3. SIMILARITY MEASURES

The similarity between two images can be characterised as follows [5].

—The similarity between two images A and B is related to their commonality. The more commonality of attributes they share, the more similar the two images are.

—The similarity between two images A and B is related to the differences between them. The more differences their attributes have, the less similar the two images are.

—The maximum similarity between two images A and B can only be reached when A and B are identical, no matter how much commonality they share.

There are many similarity measures that can be employed to deal with the many challenges that collaborative tagging presents. Jaccard coefficient [2] can be used to work out the similarity between images based on user's tags. The coefficients are used to normalise the co-occurrence between two tags. The Jaccard coefficient, sometimes referred to as the "Jaccard similarity coefficient", can be defined as a *statistic* used for comparing the similarity and diversity of sample sets. That is, given two objects, X_1 and X_2, each with n binary attributes, the Jaccard coefficient is a useful measure of the overlap that X_1 and X_2 share with their attributes. Each attribute of X_1 and X_2 can either be 0 or 1.

$$\sigma(X_1, X_2) = \frac{|X_1 \cap X_2|}{|X_1 \cup X_2|}$$

For example if we consider the following attributes for a fruit: *Sphere, sweet, sour* and *crunchy*. Then, an *Apple* (X_1) and a *Banana* is represented as

$$Apple = \{1,1,1,1\} \; and \; |\, Apple \,| = 4$$
$$Banana = \{0,1,0,0\} \; and \; |\, Banana \,| = 4$$

Here we have
$Apple \cup Banana = \{1,0\}$ with $|\, Apple \cup Banana \,| = 2$, and
$Apple \cap Banana = \{1\}$ with $|\, Apple \cap Banana \,| = 1$.

$$\sigma(Apple,\ Banana) = \frac{|Apple \cap Banana|}{|Apple \cup Banana|} = 0.5$$

The resulting similarity between two images ranges from 0 meaning the images are exactly opposite, to 1 meaning the two images are exactly the same, with 0 usually indicating independence, and in-between values indicating intermediate similarity or dissimilarity.

For example, if we search for a photo of a Barking Dog using Flickr, Figure 1:

We can add a 3-field structure to the tag, namely, object, action and background. This would be then compared with similar words stored in the WordNet database. By applying

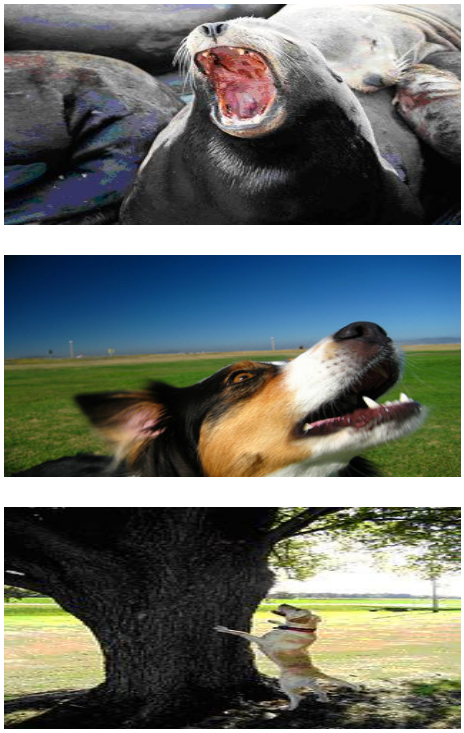


Fig. 1: Inaccurate search of barking dog

Jaccard similarity measure over the returned list of words , will give back a similarity score between 0 and 1.
Applying this to our barking dog yields a much better results, in Figure 2.

Fig. 2: Examples of Accurate search for a barking dog

4. THE FRAMEWORK

Our model of image ranking [5] can further be detailed by the following chart, Figure 3, where web users tag images with semantically related words, such as "Jelly Bean" together with "Android". Within a large photo sharing social website containing numerous independent users, such as Flickr, the semantic relationship can be captured and utilised. However, this method alone is not sufficient to all the relationships between the tags such as "window" in the photo of a "house". The photos containing both "house" and particular style of "window" may be tagged as "house" only. Such an issue can be solved by applying tag visual correlation to measure the tags visual similarity. These two methods of correlations only use the relation between tags, which can be combined in the Rankboost framework [7] [1], which in turn uses the order of instances rather than the absolute distance.

The tag recommendation process can be explained by an example, where a selected photo with user-defined tags and an ordered list of candidate tags is derived for each of the user-defined tags, based on tag co-occurrence. The lists of candidate tags are then used as input for tag aggregation and ranking, which ultimately produces the ranked list of recommended tags. For example, the photo of SacrÃ©-Coeur Figure(4) may have two user-defined tags, namely SacrÃ©-Coeur and Paris. Using Tag Co-occurrence, a list of co-occurring tags (church, architecture, montmarte, seine, Europe, travel and night) is derived 4. They have some tags in common, such as France and Paris. After aggregation and ranking four tags are recommended: *Paris*, *Church*, *Architecture* and *France*. The actual number of tags being recommended should, of course, depend on the relevancy of the tags, as we will see in the example case of using the 'value of tags'.

Tag co-occurrence is the pillar that the tag recommendation approach is built upon, and as a consequence, only works reliably when a large quantity of supporting data can be captured and accessed [6]. Fortunately, the amount of user-generated content that is created by Flickr users, satisfies this demand and provides the collective knowledge base that is needed to make tag recommendation systems work in practice. There exists various methods to calculate co-occurrence coefficients between two tags. The co-occurrence between two tags is defined as the number of photos, in our collection, where both tags are used in the same annotation.

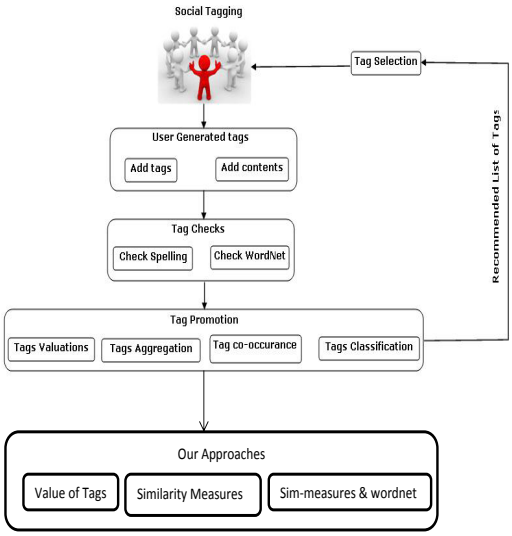


Fig. 3: General Approach

Using the raw tag co-occurrence for computing the quality of the relationship between two tags is not very meaningful, as these values do not take the frequency of the individual tags into account. Therefore it is common to normalise the co-occurrence count with the overall frequency of the tags. There are essentially two different normalisation methods: symmetric and asymmetric.

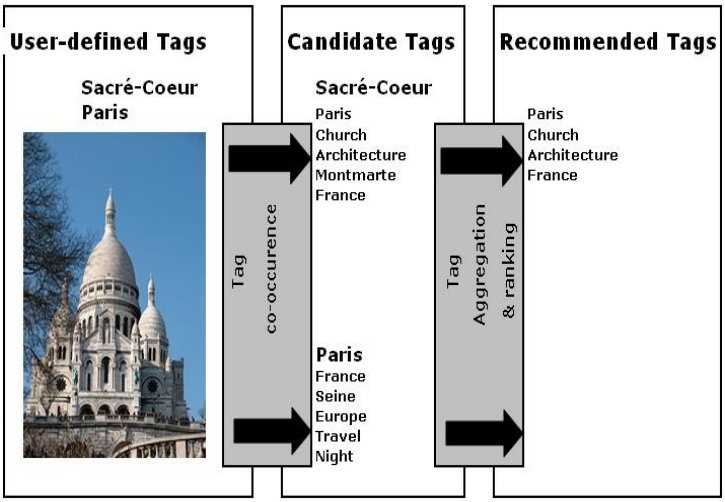


Fig. 4: The tag recommendation process

Symmetric measures:. We use the Jaccard coefficient, introduced in chapter 2, to normalise the co-occurrence of two tags t_i and t_j by calculating:

$$J(t_i,t_j) := \frac{t_i \bigcap t_j}{t_i \bigcup t_j}$$

The coefficient takes the number of intersections between the two tags, divided by the union of the two tags. The Jaccard coefficient is known to be useful to measure the similarity between two objects or sets. In general, we can use symmetric measures, like Jaccard, to deduce whether two tags have a similar meaning.

Alternatively, tag co-occurrence can be normalised using the frequency of one of the tags. We can use the equation:

The equation captures how often the tag t_t co-occurs with tag t_j normalised by the total frequency of tag t_i. This can be interpreted as the probability of a photo being annotated with tag t_j given that it was annotated with tag t_i. Many other variations of asymmetric co-occurrence measure have been proposed in the literature before to build tag (or term) hierarchies.

To illustrate the difference between symmetric and asymmetric co-occurrence measures consider the tag Eiffel Tower. For the symmetric measure we find that the most co-occurring tags are (in order): Tour Eiffel, Eiffel, Seine, La Tour Eiffel and Paris. When using the asymmetric measure the most co-occurring tags are (in order): Paris, France, Tour Eiffel, Eiffel and Europe.

It shows that the Jaccard symmetric coefficient is good at identifying equivalent tags, like Tour Eiffel, Eiffel, and La Tour Eiffel, or picking up a close by landmark such as the Seine. Based on this observation, it is more likely that asymmetric tag co-occurrence will provide a more suitable diversity of candidate tags than its symmetric opponent.

The next step in the process of tag aggregation is to merge the known lists of candidate tags for each of the user-defined tags, into a single ranking. There are two aggregation methods, based on voting (a strategy that computes a score for each candidate tag) and summing (a strategy that takes the union of all candidate tag lists) [6] that can be used along with a re-ranking procedure (where tags are arranged in their order of high relatedness [3]) that promotes candidate tags containing certain properties and significance values.

To achieve this, we use three different types of tags:

—User-defined tags U refer to the set of tags that the user assigned to a photo.
—Candidate tags C_u is the ranked list with the top most co-occurring tags, for a user-defined tag $u \varepsilon U$. We denote C to refer to the union of all candidate tags for each user-defined tag $u \varepsilon U$.
—Recommended tags R is the ranked list of the most relevant tags produced by the tag recommendation system.

For a given set of candidate tags (C) a tag aggregation step is needed to produce the final list of recommended tags (R), whenever there is more than one user-defined tag. In this section, we define two aggregation strategies. One strategy is based on voting (a strategy that computes a score for each candidate tag), and does not take the co-occurrence values of the candidate tags into account, while the summing strategy (which takes the union of all candidate tag lists) [6] uses the co-occurrence values to produce the final ranking. In both cases, we apply the strategy to the top co-occurring (or highly related) tags in the list.

Another method of increasing the accuracy of image tags is to expand the three fields used to four fields (or parameters), namely 'primary object', 'secondary object', 'action' and 'primary colour'. However, in this case, each potential tag information received will first be assessed for its value. This is also referred to as Value of Information or Value of tags.

As an example of the implementation of the 4 fields method, consider the search for a photo of a red sky at a lake. In normal circumstances, such a search may return the non-relevant image Figure(5), which shows a lake with red flowers but without the red sky at dusk.

However, our enhanced method of tagging would allow users to enter extra object names to further identify the tag. In this case, the primary object would be 'lake', the secondary object would be 'sky', the action would be 'dusk', or more precisely, 'sunset', and finally the colour would be 'red'.

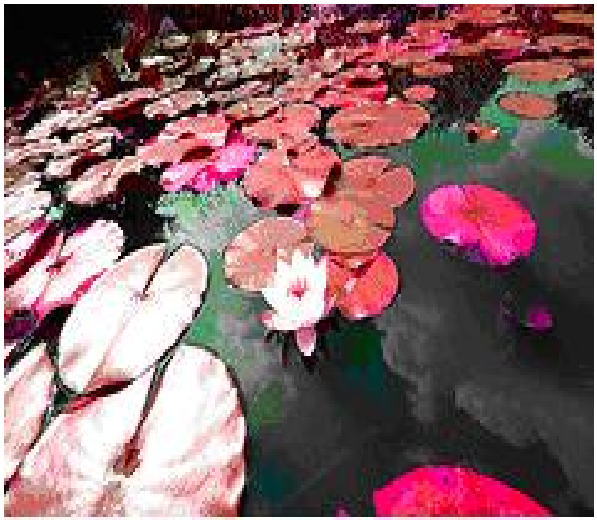


Fig. 5: Lake with red flowers

Fig. 6: Lake with red sky at dusk

In this method, before the WordNet database is queried to check if the specified words stored in the tags and returned by the search do exist, each tag returned is 'valued' against a set of pre-defined criteria. Examples of this criteria are:

—Popularity: What is the size of the tag on the Flickr tag cloud, i.e. how many times has the tag been voted for?

—Topicality: Is the tag suitable for the search topic? As an example, consider a search for an image of the city of London. The returned tags may represent London City or the novelist Jack London. In this case, the results are compared to the categories on WordNet, where London city belongs to 'noun.location'. This category is ranked higher (as it has more tags per photo) than the London Novelist category 'noun.person'

—Uniqueness: Is the tag of the photo unique and unambiguous? For example, a photo of a 'car' which is also tagged 'car' is unique and can only refer to a car, irrespective of its type.

—Redundancy: Are there too many irrelevant and redundant tags? For example, a search for a photo of a cat that returns 'cat', 'feline', 'tabby', 'fluff', 'jinx' (for a photo of a black cat) and 'cuddles' is, obviously, plagued by too many redundant tags, when 'cat' or 'feline' would suffice.

—Simplicity: How simple is a photo tag? For example, a photo of a Teapot that is tagged 'Teapot for brewing Darjeeling tea' may be too complex for search engines, as well as tag rankings algorithms (and the word Darjeeling may also be classified as spam). Ideally, the photo should be tagged as, simply, 'Teapot'.

—Spelling: Misspelled tags should, obviously, be excluded from the list of returned tags.

—Recency: For this assessment, tags are ranked by age, such that an image that has several possible tags, which were created over a long period of time, would rank the most recent tags higher than the oldest ones.

The returned list of tags is deemed to be much more accurate in terms of the search query, and this can be used to more accurately return the image in Figure(6), which represents exactly the criteria being searched for i.e. a lake with a red sky.

There are many other tag criteria that can be used to assess returned tag values. However, the criteria of topicality and relevance is of more importance as it answers the question "What are users tagging?" This criteria is mapped to WordNet categories, which are used to bind tags to the category with the highest ranking. Figure 7 shows the distribution of Flickr tags over the most common WordNet categories, which can be used to assess and classify tags. When focussing on the set of classified tags, we find that locations are tagged most frequent (28%); followed by artefacts or objects (16%), people or groups (13%), actions or events (9%), and, finally, time (7%).

From this information, we can conclude that users do not only tag the visual contents of the photo, but to a large extent provide a broader context in which the photo was taken, such as, location, time, and actions.

Another criteria that would be used to rank photo tags, is the classification of tags as defined in Table I, which looks at classes of photos with one tag, photos with 2-3 tags, 4-6 tags, and more than 6 tags, respectively. The table can be used to compare voting strategies (i.e. photos with a high number of user tags) against summation strategies (photos with

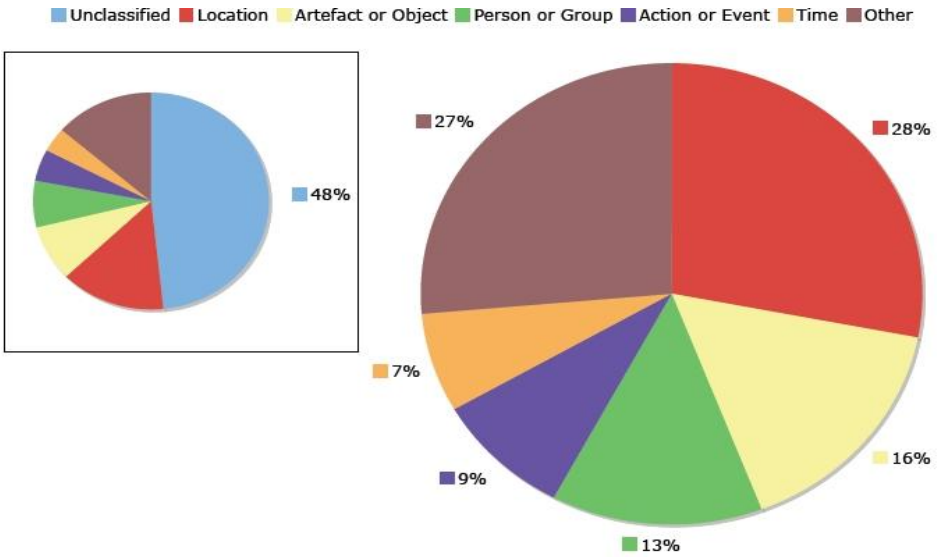


Fig. 7: Flickr's tags Most frequent WordNet categories

aggregated tags). Experiments have shown that the increases in the accuracy of tag ranking is proportional to the number of a photo's user-defined tags [4]. This indicated that only 13 % of all tagged photos have a higher degree of accuracy as they contain more than six tags. The high number of tags will serve as an input into the Jaccard measure of co-occurrence, as we will discuss in the Example section.

	Tags per Photo	**Photo%**
Class I	1	31 %
Class II	2 - 3	33 %
Class III	4 - 6	23 %
Class IV	> 6	13 %

Table I: **Definition of photo-tag classes and the percentage of photos in each class**

Finally, below we will introduce an example that details how to increase the accuracy of tagging by employing n-dimension of resources, i.e. as many of the above methods as possible.

4.0.0.7 *Summary.*. Our research work revolves around the improvements of image tagging, and for this reason, we have opted to combine many of the methods discussed in the previous sections. Users will be able to enter tags based on two searchable objects, as well as the photos action and background. This will significantly enhance the value added to the photo tags.

Once the user defined tags are saved with the photos, the returned list of tags, from a search query, will be enhanced by comparing it against a set pre-defined values (or criteria). Such an action would serve to filter out many irrelevant results. The returned list would be further enhanced by promoting the tags via the use of tag classes that utilise voting strategies.

The final filtered list of tags would then be used as recommended tags for users to choose from, as this would reduce the introduction of irrelevant tags that can be entered due to misspellings, inaccurate descriptions and attempted spamming. Users would then select one or more tags from this pre-defined list, without the ability to enter free text.

Once we get a large number of photos that have been tagged by a promoted and recommended set of tags, the set of results returned by a search query would be highly accurate. This would allow us to accurately compare similarity measures between photos using the Jaccard coefficient.

5. EXAMPLE

The example we will use is a photo of Big Ben's tower. Initially, we allow users to enter their tags into the four fields described in the previous sections; namely primary object, secondary object, action and colour. However, before the tags can be added, we use the Jaccard method to calculate co-occurrence coefficients. Both normalisation methods; symmetric and asymmetric will be used for the calculations.

For the primary object, we use the symmetric measure to find the most co-occurring tags which returns (in order): *Big Ben, Big Ben Tower, Westminster, Thames, London and England*. These recommendations will be offered to the users to populate the primary object field. Next, we use the asymmetric measure to calculate the most co-occurring tags for the secondary object which returns (in order): *London, England, Clock, Tower, Westminster, Architecture and Europe*. It is more likely that asymmetric tag co-occurrence will provide a more suitable diversity of candidate tags than its symmetric opponent. Therefore, it is more useful for returning the secondary object's recommended list. Similarly, the co-occurring tags for action would return: *Travel, Tour, Visit and Book*. Finally, the colours returned are: *Blue, Black and White*.

In Figure 8, the list of tags produced by the symmetric and asymmetric measures for each of the four fields are further aggregated to produce the final list of recommended tags. We use two aggregation strategies. One strategy is based on voting, and does not take the co-occurrence values of the candidate tags into account, while the summing strategy uses the co-occurrence values to produce the final ranking. In both cases, we applied the strategy to the top co-occurring tags in the list.

The voting strategy computes a score for each candidate tag, where a vote for that candidate is cast. A list of recommended tags is obtained by sorting the candidate tags on the number of votes. The summing strategy also takes the union of all candidate tag lists, and sums over the co-occurrence values of the tags.

Figure 7 showed that users do not only tag the visual contents of the photo, but to a large extent provide a broader context in which the photo was taken, such as, location, time, and

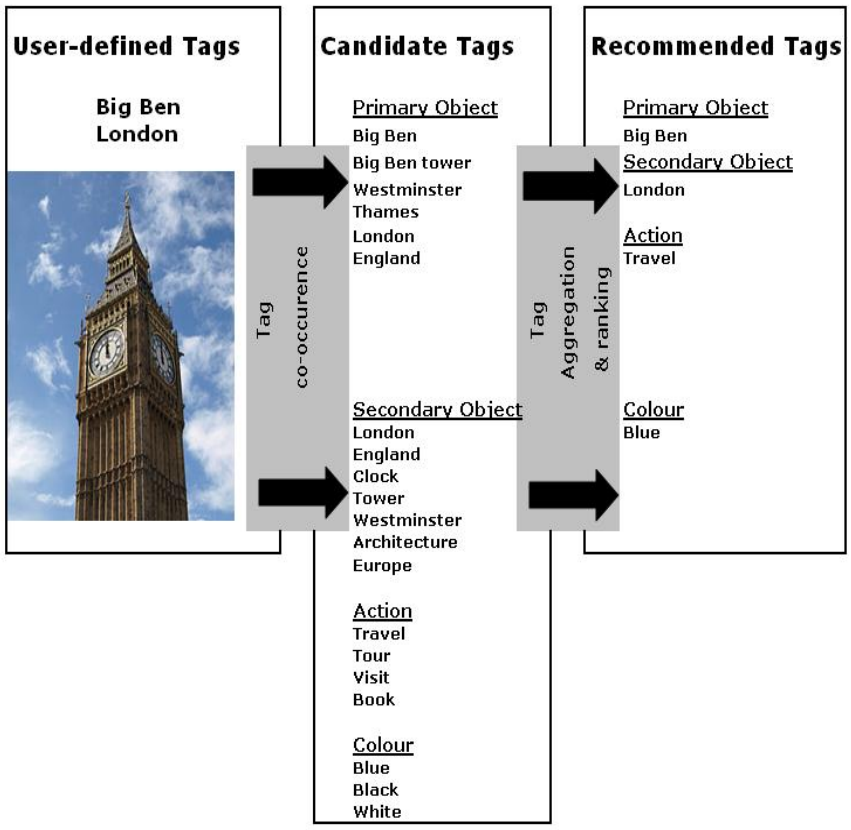


Fig. 8: Big Ben's Tag Recommendation

actions. The tags being recommended, by our above strategy, and accepted by our users can now be analysed based on vote aggregation (summing) and promotion (voting). At first, we can see that the largest (most frequent) category in the Figure is 'Unclassified' at 48%.

WordNet	Acceptance ratio %
Unclassified	39 %
Location	71 %
Artifact or Object	61 %
Person or Group	33 %
Action or Event	51 %
Time	46 %
Other	53 %

Table II: **Acceptance ratio of tags of different WordNet categories**

However, when voting is taken into account, where users select one or more of the tags recommended by our strategies, we can deduce that there exists a gap between user-defined and accepted tags for those tags which can not be classified using WordNet.

Table II shows the acceptance ratio for different WordNet categories. In the Table we can see that locations, artifacts, and objects have a relatively high acceptance ratio. However, people, groups and unclassified tags (tags that do not appear in WordNet) have relatively low acceptance ratio. We conclude that our system is particularly good at recommending additional location, artifact, and object tags.

REFERENCES

[1] Y.; R. Iyer; R.E. Schapire; Freund and Y. Singer. An efficient boosting algorithm for combining preferences. In Proc. of the 15th Intl. Conference on Machine Learning., 1998.

[2] P. Jaccard. otude comparative de la distribuition florale dans une portion des alpes et des jura.

[3] Seongjae Lee; and Soosun Cho. Web image retrieval re-ranking with wikipedia semantics. In International Journal of Multimedia and Ubiquitous Engineering., 2012.

[4] Angus E.; Thelwall M.; and Stuart D. General patterns of tag usage among university groups in flickr. Online Information Review, 2008.

[5] A. Rahuma. Semantically-enhanced image tagging system. Ph.D. Thesis, De Montfort University, Leicester (UK), 2013.

[6] BÃ**P**$_r$kur SigurbjÃ**P**$_r$nsson; and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. Yahoo! Research., 2008.

[7] L. Wu; L. J. Yang; N. H. Yu; and X. S. Hua. Learning to tag. In Proceeding of ACM International World Wide Web Conference., 2009.

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Digital Library in Virtual Education Space

Stoyanova-Doycheva, E. Doychev, S.Stoyanov

Plovdiv University, Bulgaria

In this paper is presented the common architecture of Digital Library in Virtual Education Space and its supporting operatives. Also here is shown an example for electronic testing of students by using the digital library for question generation.

Categories and Subject Descriptors: []:

General Terms: [], []

Additional Key Words and Phrases: eLearning, Virtual Education Space, Digital Library, Question Generation, Ontology.

1. INTRODUCTION

The Distributed eLearning Centre (DeLC) implemented in the Faculty of Mathematics and Informatics aims at providing context-aware delivery of electronic education services and teaching content, personalized and customized for each individual user [1,2]. DeLC is a reference architecture, supporting a reactive, proactive and personalized access to education services and electronic content. The DeLC architecture is modeled as a network which consists of separate nodes, called eLearning Nodes. Nodes model real units (laboratories, departments, faculties, colleges and universities), which offer a complete or partial educational cycle. Each eLearning Node is an autonomous host of a set of electronic services. The configuration of the network edges is such as to enable the access, incorporation, use and integration of electronic services located on the different eLearning Nodes. The eLearning Nodes can be isolated or integrated in more complex virtual structures, called clusters. Remote eService activation and integration is possible only within a cluster. In the network model we can easily create new clusters, reorganize or remove existing clusters (the reorganization is done on a virtual level, it does not affect the real organization). Wired and wireless access to the eLearning services and teaching content has been implemented in DeLC. The current DeLC infrastructure consists of two separate education clusters [3]. The first one, known as MyDeLC, delivers educational services and teaching

Correspondence address: Plovdiv University, Plovdiv, Bulgaria

content through an educational portal. The second one provides mobile access to services and content over an extended local network called InfoStations. DeLC has been used in the real educational process in the faculty during the last three years. Considering the trends of progressive transformation of Internet into Internet of Things and the emergence of Semantic Web [5, 6] we started developing Virtual Education Space (VES) as an evolution of the DeLC project. VES is an intelligent, context-aware, scenario-oriented and controlled infrastructure maintained by various assistants which are implemented as intelligent agents [7]. Three types of assistants are supported in the space [8]. The personal assistants have to perform two main functions providing the needed "entry points" of the space. First, they operate as an interface between their owners and the space and if necessary, carry out activities related to personalization and adaptation. Secondly, they interact with other assistants in the space in order to start and control the execution of the generated plans. In certain cases they operate as an intermediary for the activation of scenarios or services. The personal assistants will usually be deployed over the users' mobile devices. The specialized assistants are usually located on the server nodes of the VES, known as operatives. They support the execution of the plans generated by the personal assistants; therefore they implement suitable interfaces to the available electronic services and data repositories. Operatives serve two subspaces, known as DiLibs-Subspace and Admin-Subspace respectively. Guards are special assistants which are responsible for the safety and efficient execution of the plans in the space. These are usually intelligent devices that react to various physical quantities in the environment, e.g. smoke, temperature, humidity. The guards act as an interface between the physical and the virtual world in the space. From a functional point of view the space (VES) consists of two parts - DiLib-Subspace and Admin-Subspace. In this paper is presented the common architecture of DiLib-Subspace and its supporting operatives. Also here is shown an example for electronic testing of students by using of the digital library for question generation.

2. DIGITAL LIBRARY OVERVIEW

The e-resources are the main source for an e-learning educational system. Because of this it is very important to organize them in an appropriate structure - to make them easily accessible for the participants in the educational process. According to the structure of the Virtual Education Space [8], we propose the architecture of the digital library as shown in Fig. 1. The digital libraries in VES consist of repositories that contain different types of e-resources - ontologies, Sharable Content Objects (SCO elements), e-packages in SCORM 2004 [9], data bases with test questions, and statistics for the students. These e-resources are accessible for the following library operatives:

—Questioner Operatives - they work and access ontology repository. They use ontology to create and assess tests automatically (these operatives will be discussed below);

—Content Operative - works with repositories of SCO elements and ontologies. It creates a structure of the e-learning content, as it uses concreate ontologies and concrete requirements defined by the teacher. After that it looks for appropriate SCO elements in the repository;

—SCO Operative - works with repository of SCO elements and searches SCO elements in a specific topic. This service will help the teachers to create lectures from SCO elements;

—SCORM e-Package Operative - works with e-packages in SCORM 2004 repository and searches for e-packages created in specific topics;

—Test Operative - works with a test questions databases repository. Its task is to create exam tests in concrete discipline with specific pattern or difficulty;

—Statistic Operative - works with Statistic databases repository. This operative accesses all statistic databases in the repository and can add, edit and search different information. This information can be delivered to different participants in the educational process - other operatives in the digital library, teachers or students.

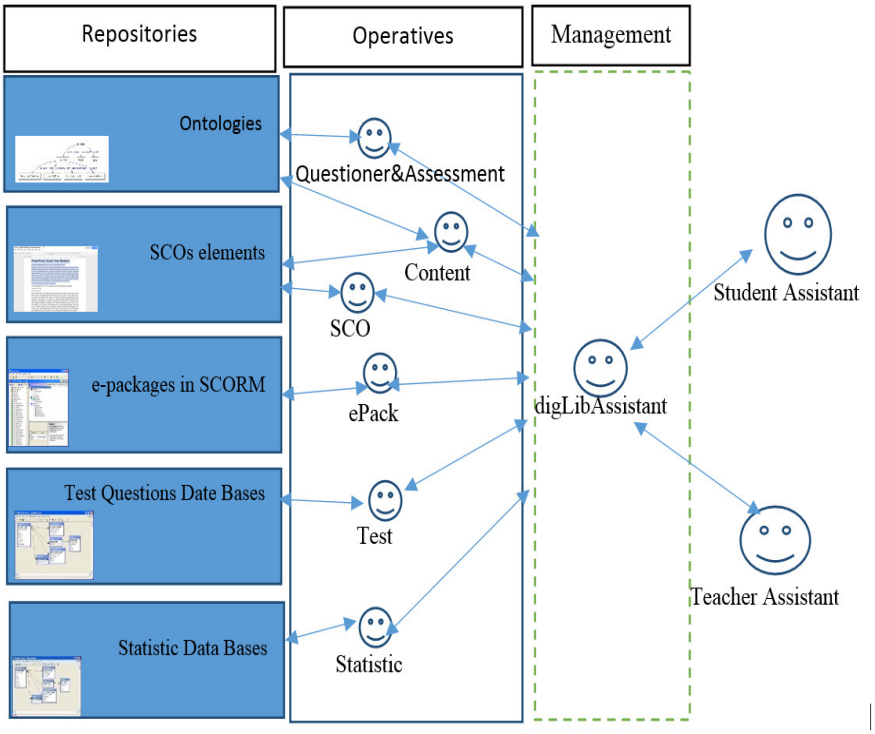


Fig. 1: DiLib Architecture

The management of the digital library in VES is the responsibility of an operative, known as digLibAssistant, which establishes the connection between personal assistants of students and teachers and the operatives that work in the digital library. The proposed digital library has a three-level architecture (Fig. 1):

—Repositories with different types of e-resources;

—Operatives that serve the digital library, as they work with a concrete type of e-resources and for certain purposes, can communicate with each other;

—The high level of the digital library architecture is the management - digLibAssistant - intelligent assistant, which communicates with the environment in VES and accepts requests to the digital library. After that the digLibAsistant distributes requests to the

appropriate operatives in the digital library. They fulfil the requests and return a response back to the client. The additional responsibility of the digLibAssistent is to generate operatives with concrete functionality in case there is are no such ones or they are busy. The operatives in the library that don't have tasks in a defined time destroy themselves. In this way, the digital library will have just as many operatives as needed.

Below will be discussed the functionality and implementation of Questioner Operatives, consisting of two operatives that generate and assess tests by using this ontology. 3 Questioner Operatives

The Questioner Operatives' main goal is to support e-testing in the software engineering education by using ontologies. The proposed test environment can work with different ontologies in different areas, but our prototype works with UML ontology. The purpose of the test environment is to provide a way for students to check their knowledge in UML. Also it can be used by teachers during exams. The main function of the environment is to create tests about UML from the UML ontology, present them to students in an appropriate way and check the answers given by the users. The test contains a number of questions, which are designed to be a short answer questions. All of the questions in one test are different. The functionality of the test environment is shown step by step in Fig. 2 using an activity diagram. When the work of the environment starts, a test begins. The first question is generated and displayed for the user. For every question the user has a defined time for answering. The user should write his/her answer before the time is up. The checking is accomplished when he/she switches to the next question or when the time is up.

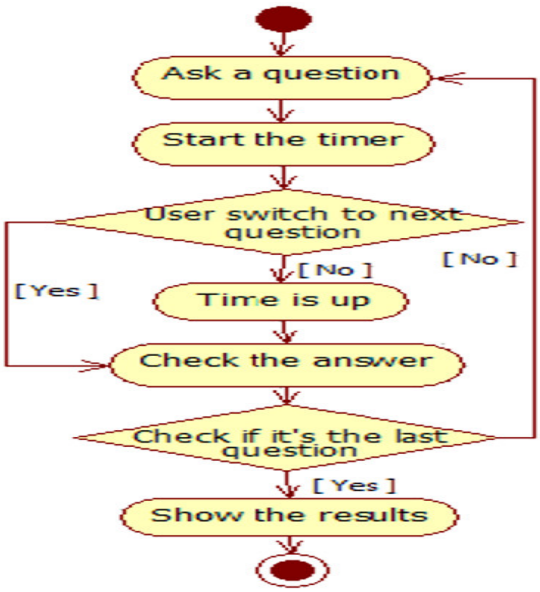


Fig. 2: Activity Diagram of test environment functionality

When answering a question is finished, the system checks if that was the last question. If it's not, a new question starts and the whole procedure is repeated. When the answer of the last question is checked, the results are shown to the user. Then he/she can see how many of his/her answers are correct and how many are the wrong ones. Also, it is displayed which of the questions have incorrect answers so students can analyze their knowledge and catch up on the information they have missed.

2.1 Architecture

The two operatives in the environment are named Questioner Operative (QO) and Assessment Operative (AO) respectively. Each of them has its own specific tasks, but they have similar architecture. Both have sensors and effectors. They are used by the operatives to accomplish interaction with their environment. Via the sensors and effectors QO and AO access the GUI, i.e. the operatives' environment. The Questioner Operative deals with the test in general. It generates random questions, which compose the UML test. Using its sensors, the Questioner Operative determines when to start or end the UML test. The sensors help the operative to decide whether to proceed with the next question. The effectors of QO are responsible for the influence on the GUI caused by the actions of the operative. By means of the effectors the operative provides the questions to the user interface and makes it to display them. In this way QO changes the GUI by showing the results at the end of the test. One of the most important behaviors of the QO is the Question Generator. Its function is to compose random questions about UML. To fulfill this, it uses the knowledge base and in particular, the axioms which construct it. The Questioner Operative's Local Control manages all of the operative's tasks and behaviours. The QO's actions depend on sensors and effectors, their collaboration and the information they gather. As a result, the Local Control regulates all of these. Here it is decided what behaviour/effector to be executed according to the information, taken by sensors or other behaviours in the operative. The second operative in the environment is the Assessment Operative. Its role is to check the users' answers and to keep records of the results during the test. By the sensors, AO establishes when the answer is ready for checking, i.e. whether the user has written an answer. The effectors here don't do a lot of work, but their function is important for the behaviour of the whole environment. The only action here is to raise a flag, which shows to the GUI that the answer is checked. This operative has a behaviour called an Answer Processor. It analyzes the answer and its task is to make it in a form, which is appropriate and easier to check its truth later. The most significant task of the AO is reviewing the answer given by the user to the question, which is asked by QO. This is achieved with the actions of the Assessment service, implemented in the Assessment Operative. The operative uses the UML ontology, where the checking algorithm confirms or denies answers. The Assessment Operative has its Local Control. It deals with coordinating the sensors and effectors of the operative and its tasks in general. Each action in the operative is based on the information, which is received from the sensors or extracted from the knowledge base. The knowledge base, used by the two operatives, is an indispensable part of our intelligent environment. Actually, this knowledge base is an ontology, which represents UML and the significant rules, building the language. Classes show the elements in UML while properties present the relations between the elements. Both the classes and the properties together construct axioms, which are a kind of principles for the UML. On these principles, in axiom forms, is based the UML test in the test environment. The ontology approach is chosen because it's a better way to present the information, so that it can be processed in a semantic means.

2.2 Communication

Although the Questioner Operative and the Assessment Operative have their own specific tasks, they need to communicate during the test to perform and coordinate their actions. In this way they can each require some performance from the other one or send information. This communication is implemented in accordance with the FIPA specifications, using JADE. QO and AO send asynchronous messages to each other (Fig. 3). In this way they accomplish the communication, when it is needed.

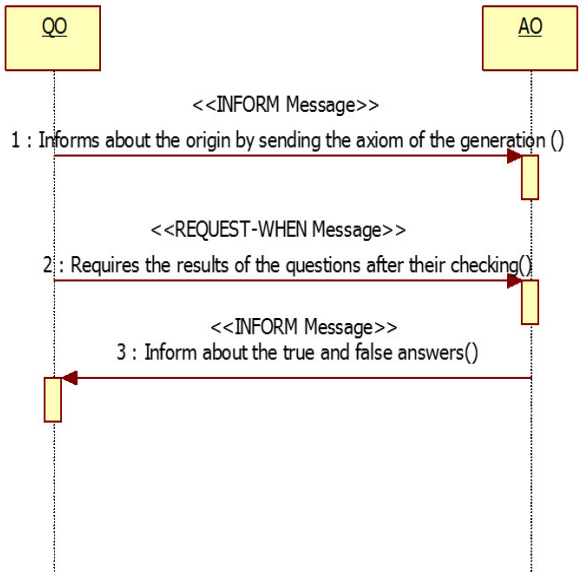


Fig. 3: Communication between QO and AO

When the Questioner Operative finishes the generation of the upcoming question, it sends an inform message to the Assessment Operative. In this way AO receives information about the question, which is necessary for checking the user's answer. The content of the message is the string of the axiom or the expression from which the question is generated. At the end of the test, after all of the questions have been asked, QO sends another message to AO because it needs information about the user's answers. Its kind is a "request-when" message. Through this the Questioner Operative requires from the Assessment Operative to send it the results of the test when they are processed. After receiving the aforementioned message and checking all of the user's answers, AO reports the score from the test to QO. This is achieved by sending an inform message, which contains information about the count of the true responses and the wrong ones. Furthermore it notifies which of the questions are answered incorrectly.

2.3 Implementation

The development environment that was chosen for developing TE is Eclipse [10]. The realized project is a simple Java desktop application. We used Java with Swing API for implementing the GUI of the environment and the two operatives are realized with the JADE framework [11]. It is powerful and convenient because it facilitates the implementation of multi-agent systems through a middleware, which complies with the FIPA specifications. The knowledge base is realized as ontology of UML. It is a good way to present the information in the UML domain, so it can be easily processed by the operatives. Protégé-OWL [12] is the used editor for creating our ontology. It is free and the OWL format, approved by W3C, is appropriate for the purpose. Another advantage is OWL API - a Java library for processing ontologies. The knowledge base is an ontology in the UML domain. Its aim is to provide knowledge for the UML elements - which are they and what relations they have between each other. This is done by creating classes for each UML element. Their semantic meaning is shown by adding properties, which relate classes, and their constraints to present the details of the elements. All of the classes in the ontology are organized in a hierarchy. In this way the type of an element can easily be determined by finding the element in the opposite position which is its superclass. There are five classes, which are the main classes on the first level of the hierarchy (Fig. 4).

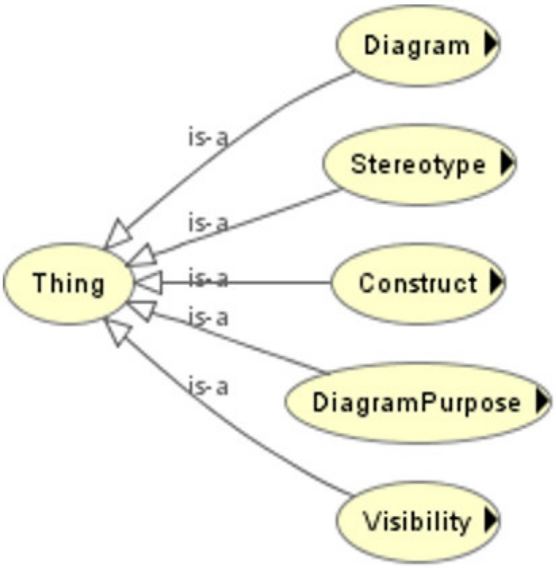


Fig. 4: Main Classes

All the other UML elements in the form of classes are subclasses of one of these. For example, the class UseCaseDiagram is a subclass of Diagram, so it can be concluded that the used case diagram is a kind of diagram. The created properties in the ontology and the existential and universal constraints establish the relations between classes. Thus we

can say how a UML element could be connected to another one or what its function is. The classes, their hierarchy, properties and constraints form the axioms of the ontology. These axioms are rules about UML domain. They are the key of semantic processing of the knowledge by the operatives. Our ontology contains over 850 axioms, so that offers a variety of questions that can be asked by TE. The most interesting and important behavior of the QO operative is QuestionGenerationService. It implements the logic for composing the questions, using the knowledge from our ontology. Also, this behavior approves for displaying only questions that weren't asked before. So every asked question is saved in a list and new questions are generated until the current one is not in that list. Then the question is visualized. The generation of a question is based on axioms in the knowledge base. The first step of the algorithm (Fig.

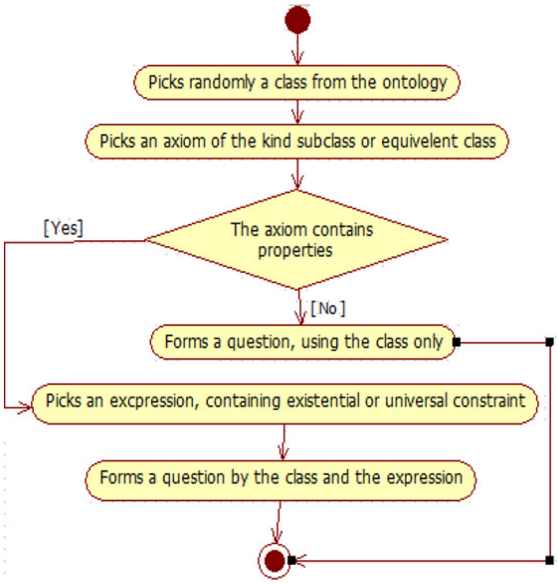


Fig. 5: Question Generation Algorithm

Then axioms are selected with reference to this class, which present a subclass or an equivalent class, and one of them is chosen, again randomly. The other types of axioms such as disjoint classes are excluded because they are not suitable for our test and also it will be more difficult to process them for the purpose. At this point the algorithm is divided into two branches according to the axiom. There is a verification that establishes if it is composed by expressions with properties. The result of this process determines which of the two methods for generating a question will be used. The first approach is a generation by only the name of the class. It is executed when the result of this analysis shows that the axiom is not formed by properties. Because of this and because of the axiom's type we can conclude that it represents which super class it is or which is the equivalent class of the one that was picked. So semantically this is the kind of the UML element, represented by its

class. For this reason the string of the question is composed by the name of the class and "is". For example, if the picked class is InteractionDiagram, the question will be "InteractionDiagram is". The user has to finish the sentence with the correct UML element. In this case that is a behavioral diagram because the interaction diagram is a kind of behavioral diagram. If there are properties in the chosen axiom the question is formed differently. The expressions in the axiom are extracted and one of them is picked. It must be of the ObjectSomeValuesFrom or ObjectAllValuesFrom type or, in other words, it must contain only existential or universal constraints. These are the most suitable types for generating our questions. If the expression has other expressions in its structure, they can be only of the ObjectUnionOf type. Other types could be used too but it will complicate the process too much and they are not included in the generation for the present. When an expression is picked, the string of the question is formed by the name of the class and the property from the expression. Thus the answer will be one or more UML elements, represented by classes, which are related to the picked class by the property. For example, UseCase is chosen to be the class and the picked expression says that it can be related by the property hasRelationship with the classes BidirectionalAssociation or Include or Extend. The formed question will be UseCase hasRelationship and the user must finish the sentence. A true answer will be at least one of the three UML elements, represented by the classes in the expression. When generating such questions with more than one answer, the operative gives a message to the user how to divide the different elements. This happens when the question is composed by an expression, containing the ObjectUnionOf type in itself. After every question string is defined, the used UML element - axiom or expression from axiom, is saved in the object, which holds the content of the message for AO. An example for a question generation from the concrete axiom in the ontology is given in Fig. ref(6).

Fig. 6: Example for Question Generation Algorithm

QO chooses randomly a class from the ontology. It refers to the concept that will be questioned. From the axioms of this class the operative picks one that will be used for the generation of the question. For example, the chosen class is ComponentDiagram and its axiom is "SubClassOf <#ComponentDiagram> ObjectAllValuesFrom(<#hasConstruct> ObjectUnionOf(<#Artifact> <#Class> <#Component> <#Dependency> <#Interface> <#Realization> <#Usage>)))". It shows the UML constructs, which could be added in the component diagram. The question in this case will be formed by taking the name of the class and the name of the object property, which is contained in the axiom. So we will have "ComponentDiagram hasConstruct" and the user is expected to complete the sentence with the correct constructs. Since the ComponentDiagram class is connected with an object union of classes, the constructs, which are represented by them, are the correct ones. So the answer in this case can contain multiple constructs. It should consist of one or more of the elements from the object union expression to be assigned as a correct answer. If there is a construct, which doesn't belong to this set, the answer is wrong. It is important that the user separates every single construct by a comma. This symbol is used by AO to "understand" where the end of the current element is and where is the start of the new one. The most important behavior for AO is CheckService. It is activated when the question and the answer are received. The algorithm for checking (Fig. 7) is based on the knowledge base and it has several branches for establishing the truth of the answer. When the operative is ready with the conclusion for the answer, a new string (true or false) is put in the vector, holding the results and checking the ends.

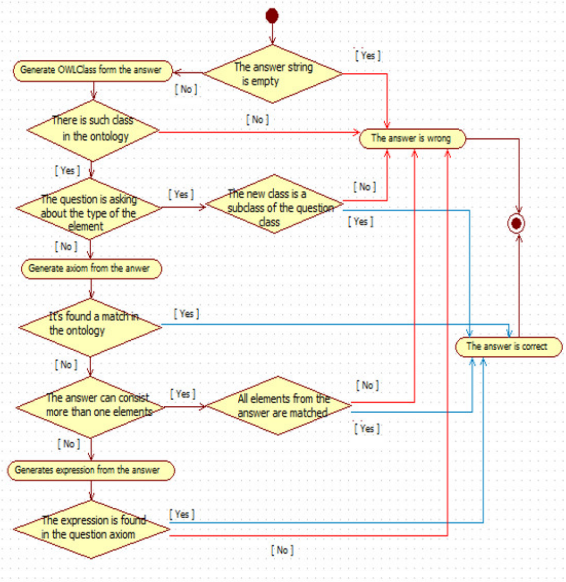


Fig. 7: The algorithm for making a conclusion for the truth of the answer

In the beginning only the first element is taken from the vector, which keeps the answer because the first case will depend on it. If it is an empty string or a relevant class is not found in the ontology, the answer is wrong and the checking is over. For the last condition an object of the OWLDataFactory type is used to create an OWLClass object by the string of the answer. After that a few preparatory steps are made. The string of the question is obtained from the GUI. As we know, the first word in it is the name of the class, from which the question is generated. Hence, it is extracted in the OWLClass object. To check the answer, the type of the question has to be defined. If its last three symbols are "is", it asks about the UML element. Consequently, the answer should match with one of the super or equivalent classes of the class from the question. If this happens the answer is true. If there is not a match, it is considered to be false. In both cases the checking ends. For the above situation, only the class is used, from which the question is generated. But if this is not the case, we will use the axiom that was sent by QO. There is are three options available. The first option is to construct an axiom using the string of the question from the GUI and the string of the user's answer. It will be compared to the real axiom and according to this the operative will make a conclusion about the answer. In order to fulfil this, we have to divide this case to several subcases, because of the different types of axioms. Subsequently, an axiom is created, which is of the EquivelentClasses or SubClassOf type. This axiom has an expression in itself, which is ObjectAllValuesFrom or ObjectSomeValuesFrom. The types depend on the information, which was sent by QO. Therefore, if this axiom is equal or is part of the string, taken by QO, the answer is true. If this condition is not satisfied, but the constructed axiom is contained by the ontology, the answer is also considered to be true. If a match is not found again, the operative passes to the next case, where the received axiom contains an ObjectUnionOf expression. We assume that the user is has written more than one UML element in the answer. So here is the place where are used all of the elements of the vector, holding the answers. In the implemented logic it is enough if all of the elements from the answer are true. It's not mandatory for all elements from the axiom to be written by the user. Consequently, the operative just checks if every answer element is part of the ObjectUnionOf expression. If they are, the answer is true, if they aren't, then it's false. The last case shows implementing the logic, when none of the upper cases was executed. It extracts an OWLClassExpression object from the answer axiom and compares it with the sent expression. The result of this checking is the conclusion for the truth of the answer.

3. CONCLUSION

The two operatives described in this paper - Questioner and Assessment, work together to create and assess tests automatically by using a specialized ontology. For example, we test our Test Environment to work with two ontologies - the UML ontology and the Software Engineering ontology. Currently the rest of the operatives, shown in Fig. 1 (Content, Test, etc), are in process of development in order to build a fully functional digital library. Such digital libraries in VES will allow the usage of more flexible and personalized learning scenarios.

REFERENCES

[1] S. e. a. Stoyanov, "From cbt to e-learning," Journal Information Technologies and Control, 3(4), 2-10, (2005).

[2] S. e. a. Stoyanov, "An approach for the development of infostation-based elearning architectures," Comptes Rendus de lÁcademie Bulgare des Sciences., 61(9), 1189-1198, 2008.

[3] E. Doychev, Environment for Provision of eLearning Services. PhD thesis.

[4] A. Kevin, "That ïnternet of things, in the real world things matter than ideas," RFID Journal,.
[5] F. Y. Wang, "The emergence of intelligent enterprises, from cps to cpss," IEEE Intelligent Systems, pp. 85-88, July/August 2010.
[6] M. Consortium, "Learning spaces framework.," Australia - New Zealand: MCEETYA Consortium, 2008.
[7] O. L. T. Berners Lee, J. Handler, "The semantic web," Scientific American, vol. 284, pp. 34-43, May 2001.
[8] T. Berners-Lee, "What the semantic web can represent," W3 org., Scientific report 2000.
[9] S. R. L. Stojanovic, S. Staab, "elearning based on the sematic web," WebNet, 2001.
[10] I. P. D. Orozova, S. Stoyanov, "Virtual education space," International Conference, Free University of Burgas, 14-15 June, 2013, 153-159, ISBN 978-954-9370-99-7.
[11] V. Valkanova, "Researching the virtual learning space in the secondary school," PhD Thesis, Sofia, 2014, ISBN 978-954-322-768-6.
[12] B. L. et. al., "Intelligent spaces: An overview," IEEE International Conference on Vehicular Electronics and Safety, 13-15 Dec. 2007, Beijing, 978-1-4244-1266-2.
[13] F.-Y. Wang, "Driving into the future with its," IEEE Intelligent System, Vol.21, No.3, 2006, pp.94-95.
[14] A. K. Dey, "Understanding and using context," Personal and Ubiquitous Computing Journal, Vol.5, No.1, pp.4-7. 2001.
[15] G. A. A.K. Dey, "Towards a better understanding of context and context-awareness," In: Proceedings of the Workshop on the What, Who, Where, When and How of Context-Awareness, New York, ACM Press, 2000.
[16] "Agent communication language specifications," http://www.fipa.org/repository/aclspecs.html.
[17] M. G. A.S. Rao, "Bdi agents: from theory to practice," In: Proceedings of the 1st International Conference on Multi-Agent Systems, San Francisco, CA, 1995, 312-319.
[18] "http://en.wikipedia.org/wiki/Event_(computing).,"
[19] "http://en.wikipedia.org/wiki/Event_(philosophy).,"
[20] L.-G. Alberto, "Probability, statistics and random processes for electrical engineering," Upper Saddle River, NJ: Pearson, 2008.
[21] "http://en.wikipedia.org/wiki/Event_(UML).,"
[22] J. R. Westermann U, "Toward a common event model for multimedia applications.," IEEE MultiMedia. 14, 19-29., 2007.
[23] J. R. Rafatirad S, Gupta A, "Event composition operators: Eco.," In EiMM 2009: Proc. 1st ACM Int. Workshop on Events in Multimedia, pp. 65-72. New York, NY: ACM., 2009.
[24] J. H. D. Allemang, "Semantic web for the working ontologist," Elsevier, 2011, ISBN: 978-0-12-385965-5.
[25] "A semantic web primer," The MIT Press, 2004.
[26] "Ims question & test interoperability overview," Final v2.1, 31 August 2012, http://www.imsglobal.org/question/qtiv2p1/imsqti_oviewv2p1.html.
[27] G. D. Bellifemine F., Caire G., "Develping multi-agent systems with jade," John Wiley & Sons, Ltd, February 2007.
[28] "http://rest.elkstein.org/2008/02/what-is-rest.html, rest.elkstein.org.,"
[29] V. et. al., "Game-oriented learning in virtual education space," This issue.
[30] A. S.-D. et. al., "Digital library in virtual education space," This issue.
[31] "Jade-leap,"
[31] "Foundation for intelligent physical agents," http://www.fipa.org.
[32] "Jadex agents," http://jadex-agents.informatik.uni-hamburg.de/.
[34] "Android for developers," http://developer.android.com/about/index.html.
[35] H. Zedan F. Siewe and A. Cau., "The calculus of context-aware ambients.," Journal of Computer and System Sciences.
[36] M. H. Al-Sammarraie, "Policy-based approach for context-aware systems," PhD thesis, Software Technology Research Laboratory, De Montfort University, Leicester, UK.
[37] H. Zedan, "Cs-flow. computational model - lunguistic support," Internal Report, STRL, De Montfort University., (2012).
[38] B. Moszkowski, "Executing temporal logic programs," Cambridge University Press, Cambridge., (1986).
[39] V. Valkanov, "Context-aware management of eservice," PhD Thesis, Sofia, 2013.

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Domain-Specific "Idea-tion1": Real Possibility or Just Another Utopia?

Delin Jing, Hongji Yang
Centre for Creative Computing, Bath Spa University

keywords: Idea-tion, Creative Computing, Ideation, Idea Generation

1 Introduction

What we become as individuals, companies, and nations depends primarily on our ideas and our ability to realise them [1]. The basis of all innovation is a creative idea. Thus, it is obviously to see new ideas are very important in various fields. Can a machine ever create new ideas? Does machines creativity is fake and predictable? Based on these questions, this paper discusses the process and possibility of domain-specific Idea-tion on the perspectives of software engineering. Ideation, generally considered as human idea generation, is the creative process of generating, developing and communicating new ideas, where an idea is understood as a basic element of thought that can be either visual, concrete or abstract [2]. It comprises all stages of a thought cycle, from innovation, development, to actualisation [1]. Idea-tion is used in this paper, stands for automatic machines idea generation. Thus, it is distinguished from ideation because Idea-tion is more concentrating on creativity generated by machine while ideation is mainly happened on human mind. Since the beginnings of computer technology, researchers have speculated about the possibility of building smart machines that could compete with human intelligence. Hence, during the past 70+ years, there are many researches and developments in the domain of Artificial Intelligence (AI) to simulate human behaviour [3, 4]. However, until now, it does not exist a computer exhibit full AI, that is to simulate a central property of human – intelligence, because we do not fully understand human beings [4]. From this perspective, some [5, 6] argue that autonomous idea generation is a utopia because human mind is unable to be replicated based on its un-predictable feature. Indeed, people [79] are still working hard to figure out how human mind works including how new ideas are generated from mind.

Such as many neuroscientists are devoted to unveil the specific neural processes leading to creative thought. However, human replica is only one road to 'idea-tion'. Is there no other road to reach it? Besides, the discussion of 'idea-tion' gives rise to several philosophical issues:

- Can computer think or do they just calculate?
- Is creativity a human prerogative?
- Does creativity depend on the material that comprises the human brain, or can computer hardware replicate creativity?

To answer above questions, following contents of this section review related works that are contributing on ideation and computer assistant systems. Based on the state-of-art, it discusses challenging issues on the realisation of 'idea-tion'. Later in this paper, it explores possible method and approach with Creative Computing to prove that there should be ways to lead 'idea-tion' from utopia to reality.

1.1 Ideation State-of-Art

Most ideation research either implicitly or explicitly assumes Osborn's conjecture that if people generate more ideas, then they will produce more good ideas [10]. There are many idea generation methods proposed by Osborn including the well-known brainstorming. These methods have been applied in many works. Osborn [10, 11] reported evidence that people generate more good ideas in the second half of a brainstorming session than during the first half. Some studies have also reported that certain ideation protocols can elevate both idea quantity and idea quality [12]. However, another work reported no relationships between idea quality and idea quantity [13]. That is, previous ideation literatures were focusing on how to generate ideas [10, 11] and idea quantity and quality [12–15]from different point of view.

There are also earlier research addressed facets if this question, from an organisational [16] or creativity perspective [17], focusing on the dialectical process of knowledge creation [18] or individual attributes [17, 19]. These studies suggest the process of ideation is a practice of researching consensus [20, 21].

In the last two decades, there are many researches concentrate on system assisted team ideation using brainstorming and related techniques. For instance, Yuan and Chen [10] proposed an E-brainstorming to utilise electronic communication to replace verbal communications and thus eliminates problems such as production blocking in companies or organisations.

Most researches are considering good idea as one that is feasible to implement, that would attain a goal, and that would not create new unacceptable conditions [16]. Hence, they neglected creativity's importance to form a novel idea.

Talking about creativity, there are few works tried to apply creativity techniques to a variety of approaches for supporting a group ideation process such as [22]. These

techniques can be executed using a Group Support System (GSS), thus allowing the ideation process to be distributed across geographical distances [22].

So far, the closest to this research is from Jogalekar and Mangla [23]. It supported the fact that machine can generate ideas, however, it considers creativity is not a necessity in the process [23]. In contrast, our work considers creativity as a core feature in 'idea-tion'.

1.2 Challenging Issue

Based on above review, there is lack of thinking of idea's creativity on existing idea generation researches. Especially, there does not exist a work focusing on designing or developing a method, framework or approach for machine automatic idea generation to provide creative ideas.

To address above issue, this paper proposes a Creative Computing method with designed approach for domain-specific 'idea-tion'. As explained in the very beginning, 'idea-tion' is more rely on creativity to generate new ideas. Therefore, it requires a new computing approach that is different from traditional system, to provide creativity fundamentally and then to accommodate 'idea-tion' process.

The rest of this research starts at review human ideation and simulate it to machine's automatic ideation, refers to section 2. On the basis of automatic ideation process, section 3 discusses how computing needs to be changed for 'idea-tion', proposes a domain-specific 'idea-tion' method, represents designed approach with relevant techniques and shows designed inference rules to support creativity. A real project named Courses Recommendation System is shown as use case in the next section to implement proposed method, techniques and rules. In the end, there is a section to conclude works, contributions and meanings of this paper. On one hand, this paper proves that domain-specific 'idea-tion' is not a utopia but a real possibility. On the other hand, it implies that it is possible to realise computer's 'Idea-tion'.

2. HUMAN IDEATION AND MACHINE 'IDEA-TION'

Ideation is usually considered as the starting point of innovation [24], i.e. new products development [25] and innovation generation [26]. Creative ideation means having ideas while new ideas arise during cognitive tasks [27]. Cognitive tasks involve either convergent or divergent thinking. Performance of a convergent thinking task requires a single correct answer while these results are measured by speed and accuracy. Divergent thinking tasks, in which one quests for many possible answers to open-ended questions, are designed to investigate novel ideas.

2.1 Human Ideation Processes

An ideation session is a period of time wherein a person or a fixed set of people work together to generate ideas in an attempt to attain a goal [14]. An idea is defined as an

actionable object-verb phrase that is presented as a potential solution to the task at hand [14]. A general human ideation process can be overviewed as following figure.

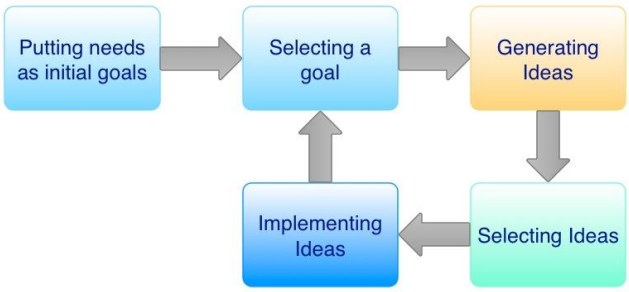


Figure 1. Human Ideation Process Overview

The whole process contains five phases: setting initial goals, selecting a goal, generating ideas, selecting ides and ideas implementing. It starts with putting various needs together as initial goals. Then a specific goal should be selected as the ideation's target. To achieve the selected goal, various ideas need to be generated and selected in the next two phases. In the end, selected ideas will be implemented to verify the selected goal can be achieved successfully. It is obviously to see that the kernel parts are the third and fourth phases, generating and selecting ideas, which are also our research scopes in this paper.

Jogalekar and Mangla's research [23] formulated the way, how the mind generates ideas as following bullet points. They also organised the working of mind in a diagram as figure 2.

- Mind accepts the information into patterns.
- It is a self-organising and self-maximising system.
- Culture helps in establishing Ideas. Culture is a set of routines followed, since generation.
- Education provides with tools to communicate these ideas and habit of learning and collecting facts and information, which is self-sorting to generate new ideas or anchor older ones.

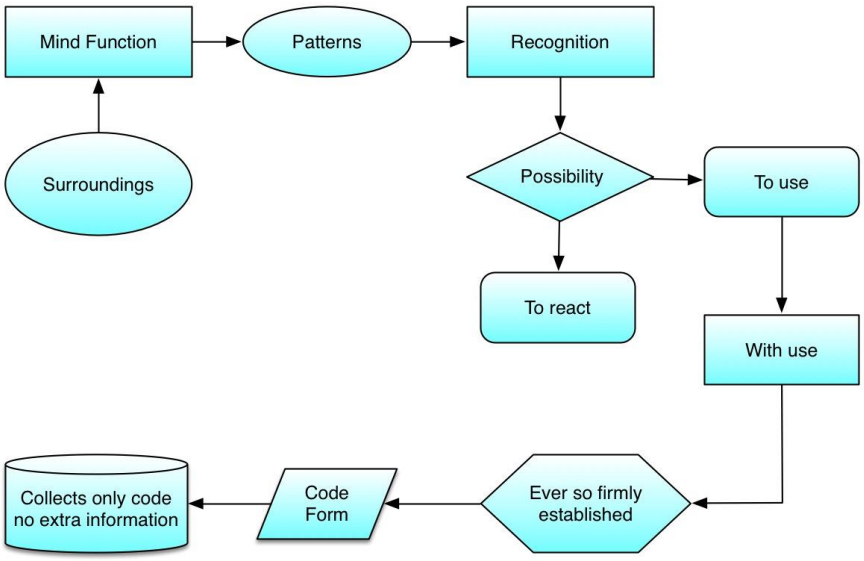


Figure 2. Working of Mind Proposed by Jogalekar and Mangla [23]

Surroundings feed in to the Mind function; they feed the elemental inputs to the brain, which it interprets as patterns of logical setups or random features. Then it operates on the patterns, which undergo the recognition phase where there are two possibilities as follows [23]:

- To react to the patterns; this can vary on a wider scale of probabilities of the decision making to reaction to be taken towards the pattern.
- Establish the patterns with use.

These patterns are then more firmly established and stored with the code form. The codes are virtual markers, which help in calling up the file or pointing to the data of pattern a more correct word to us here, which is to be read and referred to on need of thought process initiation.

Besides, some researches are working on applying brainstorming into engineering design creativity and ideation, such as Figure 3 from Goncher and his colleagues' work[28]. Because this figure is easy to understand, detailed explanations are omitted .

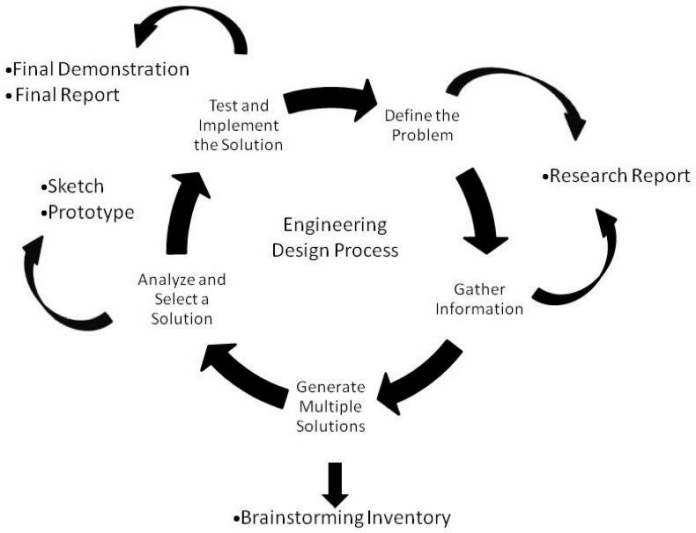


Figure 3. The Engineering Design Process and Corresponding Artifacts [28]

According to above information, in order to represent clearly, we summarised human idea generation process as figure 4 with explanations followed.

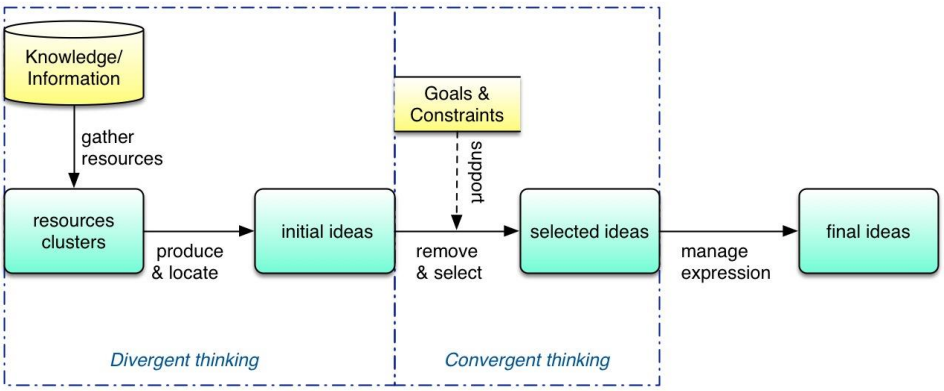


Figure 4. Human Idea Generation Process

Basically, as above diagram shows, there are four main steps to generate ideas under general circumstances,

Step 1: Gather various resources from personal knowledge and public information. Results of this step can be seen as a lot recourses clusters.

Step 2: Based on gathered resources, human produce new ideas or locate existing ideas. All of them are considered as initial ideas.

Step 3: In this step, initial ideas are reviewed according to goals and constraints. Useless and inappropriate ideas are removed; good ideas are remained as selected ideas.

Step 4: It manages expressions of select ideas to get better representations.

From cognitive perspective, step1 to step 2 are divergent thinking. Then it is convergent thinking to reduce number of ideas in step 3. Finally, there is an evolution step to formalise final ideas' expression.

The summarised process represents a process of individual's idea generation. For teamwork, it need to replica step 1 and 2 for each person to generate ideas from every member and merge everyone's ideas to form a big set of initial ideas. Discussion should be involved in each step's activities.

2.2 Machine Ideation Process

The human ideation is not only based on personal domain knowledge but also influenced by the effect factors including need, stress, urgency, expression, focus and clarity. Therefore, human ideation is not a stabilised process, which means it is unable to guaranty quality of generated ideas. For example, urgency usually equals limited time, which may lead to fewer ideas generated and no one is really suitable to achieve selected goal. Machine 'idea-tion' can avoid or reduce some of those unstable factors via computing's specialties, i.e., vast knowledge database, fast computing speed, un-emotional process, etc.

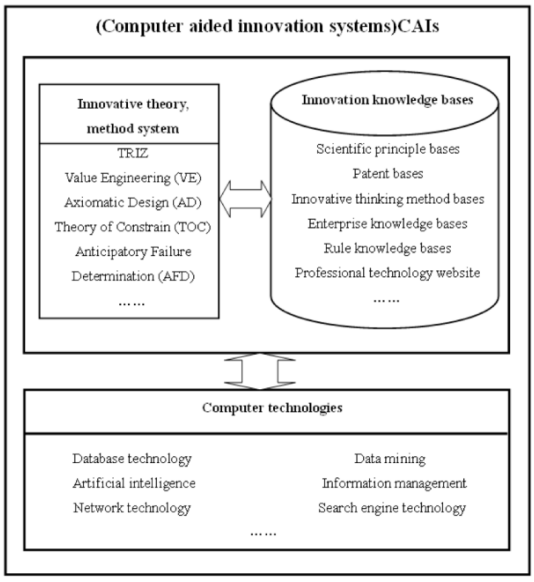


Figure 5. Structure of CAI System [29]

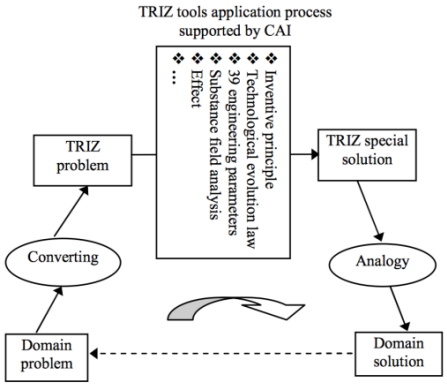


Figure 6. Solving Problem Supported by CAI [29]

Presently, there are researches worked on designing computer systems to support idea generation. For example, Jianhui Zhang and his colleagues [29] proposed a computer-aided innovation (CAI) system to assist new idea generation for product conceptual design, refers to Figure 5 and 6. However, this kind of works are more concentrating on solving problems than contributing on inspiring innovation through generated ideas.

According to last section's review of human ideation, using existing relevant works as basis, human idea generation process is simulated to machine's automatic 'idea-tion', which is represented as figure 7 with specified steps.

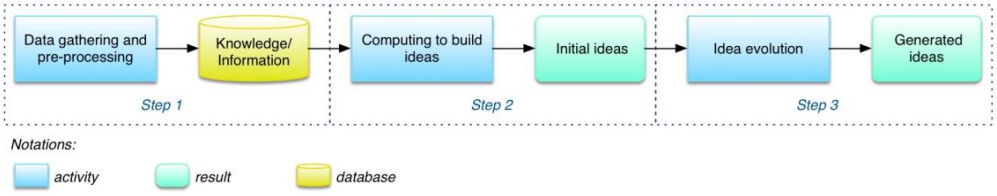


Figure 7. Machine 'Idea-tion' Process

Step 1: Data gathering and pre-processing. Determine the objective of task and select relevant data. And then text data is pre-processed and saved as relevant knowledge and information.

Step 2: Computing to build ideas. Based on gathered and processed knowledge and information, system computes following designed algorithms and rules to generate initial ideas to realise convergent thinking.

Step 3: Idea evolution. It is simulated step 3 and 4 of human idea generation process, which covers ideas reduction, selection and expression management.

This machine ideation process is a simulation of summarised individual idea generation (Figure 4). In contrast to human ideation, in this process, creativity does not rely on human being but generated via system computing. Next section will propose a Creative Computing method with relevant techniques employed in its approach to explain why machine automatic 'ideat-ion' is possible to be realised. In addition, there is a set of rules designed to support inference engine, which is the kernel part of proposed approach.

3. PROPOSED METHOD, TECHNIQUES AND RULES

3.1 Creative Idea and Creativity

Because creative ideas are different from those that normally arise, people often believe that such ideas require conditions dramatically different from the usual [30]. This view prompted the emergence of various idea-generating methods: brainstorming, synectics, lateral thinking, random simulation, and so on, all of which consist of withholding judgment and relying on analogies from other members in the group or on randomly selected analogies [29, 31]. This family of methods relies on the assumption that enhancing randomness, breaking rules and paradigms, and generating anarchy of thought can increase the idea's creativity and enhance probability of creative idea emergence [30].

Boden [32] says a creative idea is one which is novel, surprising, and valuable. Most important, creative ideas should be surprising because they go against out expectations [33]. According to Boden's definition [32, 33], an idea can be called "new" from two

perspectives: the objective (H-creative) and the subjective (P-creative) view [31]. They derive from two kind of creativity: *H-creativity* (short for historical creativity) and *P-creativity* (short for psychological creativity). *H-creativity* is fundamentally novel in respect to the whole of human history and *P-creativity* is the personal kind of creativity that is novel in respect to the individual mind [34, 35]. Hence, a H-creative idea must also be P-creativity. On the contrary, a P-creative idea may not be H-creativity, because, it only new to certain people but may be familiar with others. From the above discussion, the creative ideas expected from the proposed 'idea-tion' should belong to H-creativity. That is to say, the new ideas we concentrated on should be new to the world first. Obviously, H-creative ideas are very difficult to be generated by individual or a group, because it requires to be supported by extensive knowledge.

Apart from H-creative ideas and P-creative ideas, there are other catalogues of creative ideas. For example, Graham and Bachmann [1] classified new ideas into three kinds of ideas:

- *Derivative ideas*: a derivative idea involves taking something that already exits and changing it.
- *Symbiotic ideas*: A symbiotic method of idea creation is when multiple ideas are combined, using different elements of each to make a whole.
- *Revolutionary ideas*: a revolutionary idea breaks away from traditional thought and creates a brand new perspective.

Because they all can be H-creativity, these three types of new ideas are employed in this research. According to this classification, algorithms can be used to express different kinds of new ideas formally. We assume there are three existing ideas for certain problem "I_1", "I_2" and "I_3". Thus, above three kinds of ideas expressed as below formulas with examples,

- *Derivative ideas* I_x':
  $I_x' = f(I_x)$
  i.e., $I_1' = f(I_1)$
- *Symbiotic ideas* I_z:
  $I_z = f(I_x, I_y)$
  i.e., $I_4 = f(I_1, I_2)$
- *Revolutionary ideas* I_m:
  $I_m = ff(I_x)$
  i.e., $I_a = ff(I_3)$

Creativity is considered the ultimate human activity, a highly complex process, difficult to formalise and to control [30]. Some researchers hold that the creative thinking process is qualitatively different from "ordinary" day-to-day thinking [30, 34, 36], and involves a leap that cannot be formulated, analysed, or reconstructed [30]. Others adopt a reductionist view that creative products are the outcome of ordinary thinking, only quantitatively different from everyday thinking [29, 32, 33, 35].

3.2 Creative Computing Method and Approach

Considering above discussions on creative ideas and creativity, it requires a new computing method that is different from traditional system, to provide creativity fundamentally and then to accommodate 'idea-tion' process. Creative Computing is being discussed more widely in last few years, hoping to produce new, surprising and useful products [37]. It is mainly happening in software [38]. Creative Computing seeks to reconcile the objective precision of computer systems (mathesis) with the subjective ambiguity of human creativity (aethesis) [38]. One research objective in Creative Computing is to find the approach to get creativity and to realise it [37, 38]. Therefore, a Creative Computing method can be a way to realise 'idea-tion'.

In addition, according to Boden's work [33], creativity can be divided into three types as following descriptions and each type is a creative technique.

- *Combinational creativity*: it is to combine familiar ideas to make unfamiliar/new ideas. Typical combinational creativities includes bisociation, juxtapositioning of unrelated ideas, puns, conceptual blending, etc. [31]. Bisociation connects to incomparable contexts together; conceptual blending is to generate new ideas by integrating very different thoughts [37].
- *Exploratory creativity*: further research is conducted in one already existing conceptual space in order to find new route to reach a destination [37]. In another word, it is exploration of conceptual spaces to notice new things in old places [31].
- *Transformative creativity*: it means to create its own new exclusive conceptual space by transforming an existing conceptual space, so that the original problem can be transformed to new conceptual space and be solved there [37]. It makes new thoughts possible by transforming the conceptual space, by altering its own rules [31].

Considering classification of new ideas and three types of creativity, it is obviously to see they are perfectly matched. Combination creativity leads to symbiotic ideas, exploratory creativity matches derivative ideas, and revolutionary ideas are transformative creativity.

With these in mind, the required Creative Computing method should be a new computing approach that is different from traditional idea generation and problem solving system. It aims to build systems to combine entities, to transform one entity to another space and to explore entities in the solution space [37]. Hence, a new idea generation process can be drawn as Figure 8 as the proposed Creative Computing approach. The proposed method is called Creative Idea Generation in the following context.

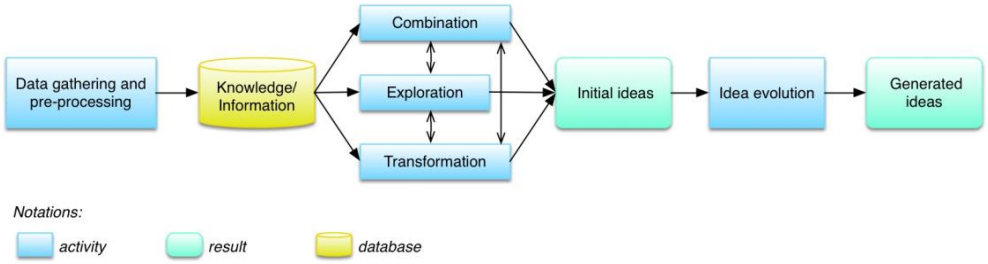


Figure 8. Creative Idea Generation Process

Combination, exploration and transformation are kernel activities of its computing step to generate initial ideas. Combination activity involves unfamiliar combinations of familiar knowledge and information. Exploration activity explores within an established conceptual space. This is more likely to arise from a thorough and persistent search of a well-understood space. Transformation activity deliberately transforms a conceptual space. It should involve the rejection of some of the constraints that define this space and some of the assumptions that define the problem itself. These three kinds of activities provide the basis of the techniques to compute resources and generate initial ideas. The results of one activity can be input of another activity to generate ideas through multi-activities. However, it is not necessary to implement all three kinds of activities. The practical realities of their application must be worked out in different applications and circumstances, usually on a case-by-case basis. In section 3.3, a set of rules is designed to support the proposed approach. These inference rules are fundamental reasoning logics employed description logics and the three kind of creative activities (combination, exploration and transformation).

3.3 Inference Rules

All reasoners use algorithms for inferring implicitly stated knowledge and, based on these algorithms, they also support some basic reasoning tasks, e.g., determining if a given individual is an instance of a given concept. Systems usually support the following standard reasoning tasks:

1) Sub-sumption Test: to determine whether one concept is more general than another. Sub-sumption tests are used to build and maintain a taxonomy of named concepts, called the concept hierarchy, and the process of building the concept hierarchy is called classification.
2) Satisfiability Test: to determine whether the constraints implied by the knowledge base are such that a concept is contradictory and thus its extension is empty in every model of the knowledge base.
3) Consistency Test: to determine whether a given knowledge base can have a model.
4) Retrieval: to retrieve all instances of a given concept or all pairs of individuals related via a given role.

However, this research requires more creative and complex queries. For example, a query that asks for a new research idea based on input keyword/keywords. The required new ideas cannot simply be searched from existing knowledge due to its creativity requirements. Therefore, an inference engine is proposed to support ideas generation. Figure 9 represents where the inference engine fits in an "idea-tion" system.

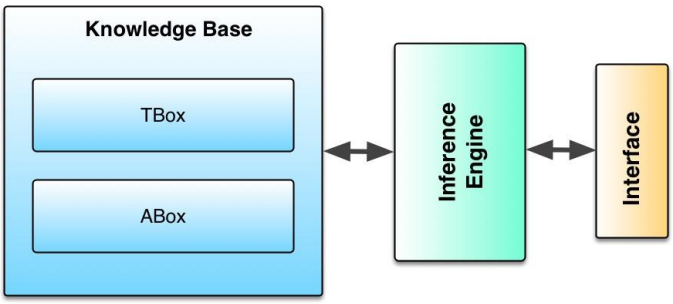


Figure 9. Inference System for "Idea-tion"

1) Preliminaries

This part defines the main DL and the reasoning problem considered in this research. The concept constructors it provides determine the expressivity of a particular DL. The informal naming convention for DLs describes the constructors that can be used: $\mathcal{E}$ (existential restrictions), $\mathcal{Q}$ (qualified number restrictions), $\mathcal{O}$ (nominal, objects), $\mathcal{I}$ (inverse roles), $\mathcal{H}$ (role hierarchies) and $\mathcal{S}$ is the abbreviation for $\mathcal{ALC}$ with transitive roles. $\mathcal{ALC}$ is a basic DL uses the DL fundamental constructors (refers to Table 1). It is not enough to describe the knowledge base this research required.

Therefore, it requires a specific DL in this research. $\mathcal{SHIQ}$ is the DL we propose to use in this research. Its syntax and semantics are represented in Table 1 and Table 2. Former table contains constructors of basic DL ALC. Special constructors for $\mathcal{SHIQ}$ are represented in later table that provides inverse roles and transitive roles.

Table 1. Syntax and semantics of ALC

Constructor	Syntax	Semantics
Atomic Concept	$\mathcal{A}$	$\mathcal{A}^I$
Atomic Relation	$\mathcal{R}$	$\mathcal{R}^I \subseteq \Delta^I \times \Delta^I$
Top	$\top$	Δ^I
Bottom	$\bot$	$\emptyset$

Intersection	$\mathcal{C} \sqcap \mathcal{D}$	$\mathcal{C}^I \cap \mathcal{D}^I$
Union	$\mathcal{C} \sqcup \mathcal{D}$	$\mathcal{C}^I \cup \mathcal{D}^I$
Negation	$\neg \mathcal{C}$	$\Delta^I \setminus \mathcal{C}^I$
Existential restriction	$\exists \mathcal{R}. \mathcal{C}$	$\{x \in \Delta^I \mid \exists y: (x, y) \in \mathcal{R}^I \wedge y \in \mathcal{C}^I\}$
Value restriction	$\forall \mathcal{R}. \mathcal{C}$	$\{x \in \Delta^I \mid \forall y: (x, y) \in \mathcal{R}^I \rightarrow y \in \mathcal{C}^I\}$
Axiom	$\mathcal{C} \sqsubseteq \mathcal{D}$	$\mathcal{C}^I \subseteq \mathcal{D}^I$

According to the needs of domain-specific knowledge base, more constructors are required. Therefore, $\mathcal{SHIQ}$ is the specific DL we choose which contains constructors in Table 2.

Table 2. Specific constructors for $\mathcal{SHIQ}$

Constructor	**Syntax**	**Semantics**
At-least restriction	$(\geq n \mathcal{R}. \mathcal{C})$	$\{x \in \Delta^I \mid \#\{y \in \Delta^I \mid (x, y) \in \mathcal{R}^I \wedge y \in \mathcal{C}^I\} \geq n\}$
At-most restriction	$(\leq n \mathcal{R}. \mathcal{C})$	$\{x \in \Delta^I \mid \#\{y \in \Delta^I \mid (x, y) \in \mathcal{R}^I \wedge y \in \mathcal{C}^I\} \leq n\}$
Inverse role	$\mathcal{R}^-$	$\{(x, y) \mid (y, x) \in \mathcal{R}^I\}$
Trans role	$\mathcal{R}_+$	$\mathcal{R}^I = (\mathcal{R}^I)^+$

In table 1 and table 2, $\mathcal{C}$ and $\mathcal{D}$ are concepts, and $\mathcal{R}$ is a role name. Specifically, it expresses instances/individuals, concepts and roles/relations as following,

- Set of individuals a, b, c …
- Set of atomic concepts (class names) $\mathcal{C}, \mathcal{D}$ …
- Set of role names $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \ldots, \mathcal{R}_n, \ldots$

A **TBox** is a set of statements of the form $\mathcal{C} \equiv \mathcal{D}$ or $\mathcal{C} \sqsubseteq \mathcal{D}$, where $\mathcal{C}$ and $\mathcal{D}$ are class expressions. It also contains $\mathcal{R}(\mathcal{C})$ or $\mathcal{R}(\mathcal{C}, \mathcal{D})$. They are called general inclusion axioms.

An **ABox** consists of statements of the form $\mathcal{C}(a)$ or $\mathcal{R}(a, b)$, where $\mathcal{C}$ is a class expression, $\mathcal{R}$is a role, and a, b are individuals.

2) Exploratory Inference Rules

Based on features of exploratory creativity, there is a set of rules defined to reason concepts and relations. The results are basis and kernel elements to form new ideas with other rules. The following rules assume inputs are concepts or instances existing in the knowledge base.

$$f_e(C) = C \sqcup R_i \leftarrow (\nexists R_i.C) \wedge (\exists R_i.D) \wedge (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset) \quad (1)$$

$$f_e(a) = C \sqcup R_i \leftarrow (a \in C) \wedge (\nexists R_i.C) \wedge (\exists R_i.D) \wedge (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset) \quad (2)$$

$$f_e(C) = \emptyset \leftarrow (\nexists R_i.C) \wedge (\nexists R_i.D) \wedge (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset) \quad (3)$$

$$f_e(a) = \emptyset \leftarrow (a \in C) \wedge (\nexists R_i.C) \wedge (\nexists R_i.D) \wedge (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset) \quad (4)$$

Rule (1) says, for input C, if the following conditions are met, the inference result is the concept C with relation R_i.
Conditions:

i. For concept C, relation R_i does not exist;
ii. For concept D, relation R_i exits;
iii. Concept D is not null and is not equals concept C; and
iv. Concept D is not sub-concept of C.

Rule (2) says, for input a, if a is an instance of concept C, the inference conditions and results are same as rule (1).

Rule (3) says, for input C, if the conditions of rule (1) cannot meet, then it inference result is null.

Rule (4) says, for input a, if a is an instance of concept C, when the conditions of rule (2) cannot meet, then it inference result is null.

Followings are exploratory inference rules for two and three inputs. These inputs can be concepts or mixture of instances and concepts. Rules of null results are omitted here. According to above rules, it is easy to understand the inference result is null if the corresponding rule's conditions cannot meet.

$$f_e(C,D) = f_e(a,D) = f_e(a,b) = f_e(C) \sqcup f_e(D) \sqcup R_i(C,D) \leftarrow \left(\exists! R_i(C,D)\right) \wedge (\exists R_i.\neg C) \vee (\exists R_i.\neg D), while\ a \in C\ and\ b \in D \quad (5)$$

Rule (5) means, for two valid inputs, if the following conditions are met, its reasoning result includes the concept C, D with relation $R_i(C, D)$.
Conditions:

i. For concept C and D, relation R_i that is connecting concepts C and D does not exist;
ii. Relation R_i exists in a concept while this concept is not C or D;
iii. If input contains one instance a, it is an instance of concept C; if inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D.

$$f_e(C, D, E) = f_e(a, D, E) = f_e(a, b, E) = f_e(a, b, c) = f_e(C, D) \sqcup f_e(C, E) \sqcup f_e(D, E) \leftarrow \left(\exists! R_i(C, D)\right) \wedge (\exists R_i. \neg C) \vee (\exists R_i. \neg D) \wedge \left(\exists! R_i(C, E)\right) \wedge (\exists R_i. \neg C) \vee (\exists R_i. \neg E) \wedge \left(\exists! R_i(D, E)\right) \wedge (\exists R_i. \neg D) \vee (\exists R_i. \neg E), while\ a \in C, b \in D\ and\ c \in E \quad (6)$$

Rule (6) represents, for three valid inputs, if the following conditions are met, its reasoning result is the combination of exploratory inference results of every two concepts.
Conditions:

i. If input contains one instance a, it is an instance of concept C; if inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D; if inputs are three instances a, b and c, a is an instance of concept C, b is an instance of concept D, and c is an instance of concept E.
ii. Rule (5) can be applied to concept C and D;
iii. Rule (5) can be applied to concept C and E; and
iv. Rule (5) can be applied to concept D and E.

Over all, rules (1) to (6) are designed based on exploratory creativity to assist reasoning in the proposed approach.

3) Transformative Inference Rules

Based on features of transformative creativity, there is a set of rules defined to reason concepts. The results are basis and kernel elements to form new ideas with other rules. The following rules assume inputs are concepts or instances existing in the knowledge base.

$$f_t(C) = f_t(a) = D \leftarrow (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset), while\ a \in C \quad (7)$$

Rule (7) represents, for an input concept C or instance a, if the following conditions are met, its reasoning result is another concept D.
Conditions:

i. Input D is not C;
ii. D is not a sub-concept of C;
iii. D is not a empty concept; and

iv. If input is an instance, $a \in C$.

$$f_t(C, D) = f_t(a, D) = f_t(a, b) = f_t(C) \sqcup f_t(D) \leftarrow \ f_t(D) \neq f_t(C), while\ a \in C\ and\ b \in D \quad (8)$$

Rule (8) represents, for two inputs, if the following conditions are met, its reasoning result is a union of $f_t(C)$ and $f_t(D)$. $f_t(C)$ and $f_t(D)$ refer to rule (7).
Conditions:

i. If input contains one instance a, it is an instance of concept C; if inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D.
ii. Rule (7) can be applied to concept C and D; and
iii. Results generated by $f_t(C)$ and $f_t(D)$ are different.

$$f_t(C, D, E) = f_t(a, D, E) = f_t(a, b, E) = f_t(a, b, c) = f_t(C) \sqcup f_t(D) \sqcup f_t(E) \leftarrow f_t(E) \neq f_t(D) \neq f_t(C), while\ a \in C, b \in D\ and\ c \in E \quad (9)$$

Rule (9) means, for three valid inputs, if the following conditions are met, its reasoning result is the combination of rule (7) applied to each concept.
Conditions:

i. If input contains one instance a, it is an instance of concept C; if inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D; if inputs are three instances a, b and c, a is an instance of concept C, b is an instance of concept D, and c is an instance of concept E.
ii. Rule (7) can be applied to concept C, D and E; and
iii. Results generated by $f_t(C)$, $f_t(D)$ and $f_t(E)$ are different with each other.

Rule (7), (8) and (9) are transformative inference rules for one, two and three inputs. These inputs can be concepts or mixture of instances and concepts. Rules of null results are omitted here. As designed exploratory inference rules, the inference result is null if the corresponding rule's conditions cannot meet.

4) Combinational Inference Rules

According to combinational creativity, a set of rules is defined to reason combination of different concepts. The results are basis and kernel elements to form new ideas with other rules. The following rules assume inputs are concepts or instances existing in the knowledge base.

$$f_c(C) = f_c(a) = C \sqcup D \leftarrow \ (D \in \neg C) \wedge (D \not\subseteq C) \wedge (D \neq \emptyset), while\ a \in C \quad (10)$$

Rule (10) means, while the input is a concept C or an instance a, if the following conditions are met, its reasoning result is the union of two concepts $C \sqcup D$.
Conditions:

i. Concept D is a negation of concept C; that is concept D can be any concept in the KB's TBox except concept C;
ii. Concept D is not a sub-concept of C;
iii. Concept D is not a null; and
iv. If input is an instance a, it is an instance of concept C.

$f_c(C, D) = f_c(a, D) = f_c(a, b) = C \sqcup D \sqcup E, \leftarrow \big(E \in (\neg C \vee \neg D)\big) \wedge \big(E \nsubseteq (C \wedge D)\big) \wedge (E \neq \emptyset), while\ C \neq D, a \in C\ and\ b \in D$
(11)

Rule (11) says, for two valid inputs, if the following conditions are met, its reasoning result is the union of three concepts $C \sqcup D \sqcup E$.
Conditions:

i. Concept E is a negation of concept C and negation of concept D; that is concept E can be any concept in the KB's TBox except concept C and D;
ii. Concept E is not a sub-concept of C and D;
iii. Concept D is not a null;
iv. Concept C is not equals to concept D; and
v. If inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D

$f_c(C, D, E) = f_c(a, D, E) = f_c(a, b, E) = f_c(a, b, c) = C \sqcup D \sqcup E \sqcup F \leftarrow \big(F \in (\neg C \vee \neg D \vee \neg E)\big) \wedge \big(F \nsubseteq (C \wedge D \wedge E)\big) \wedge (F \neq \emptyset), while\ C \neq D \neq E, a \in C, b \in D\ and\ c \in E$ (12)

Rule (12) means, for three valid inputs, if the following conditions are met, its reasoning result is the combination of four concepts $C \sqcup D \sqcup E \sqcup F$.
Conditions:

i. If input contains one instance a, it is an instance of concept C; if inputs are two instances a and b, a is an instance of concept C, and b is an instance of concept D; if inputs are three instances a, b and c, a is an instance of concept C, b is an instance of concept D, and c is an instance of concept E;
ii. Concept F is a negation of concepts C, D and E; that is concept F can be any concept in the KB's TBox except concept C, D and E;
iii. Concept F is not a sub-concept of C, D and E;
iv. Concept F is not a null; and
v. C, D and E are not equals to each other.

Rule (10), (11) and (12) are combinational inference rules for one, two and three inputs. These inputs can be concepts or mixture of instances and concepts. Rules of null results are omitted here. As designed other inference rules, the inference result is null if the corresponding rule's conditions cannot meet.

5) Weight Calculation Algorithms and Inference Rules

In the existing research results, the less appears means more valuable as a new idea. Therefore, we propose adding a property named "weight" $\mathcal{W}$ for each concept. Also, algorithms are designed to calculate the value of $\mathcal{W}$; rules are defined to measure degree of innovation for generated ideas. Obviously, the smaller the value of $\mathcal{W}$ is, the more worthy of recommendation.

To get the value of $\mathcal{W}$, it is necessary to have a property N for each concept and property to calculate the number of its appearing. The value of N is a natural number and should be equal to or greater than 1. Algorithms designed to calculate value of $\mathcal{W}$ are listed and explained below.

Weight calculation algorithms:

Algorithm 1: Concept weight calculation

$$\mathcal{W}.\mathcal{C} = \frac{N.\mathcal{C}}{N_{max}.\mathcal{D}}, \mathcal{D} \in (\neg\mathcal{C} \sqcup \mathcal{C}) \quad (13)$$

The above rule says weight $\mathcal{W}$ of concept $\mathcal{C}$ is the ratio of $N.\mathcal{C}$ and $N_{max}.\mathcal{D}$.

Algorithm 2: Relation weight calculation

$$\mathcal{W}.\mathcal{R}(\mathcal{C}) = \frac{N.\mathcal{R}(\mathcal{C})}{N_{max}.\mathcal{R}(\mathcal{D})}, \mathcal{D} \in (\neg\mathcal{C} \sqcup \mathcal{C}) \quad (14)$$

The above rule says weight $\mathcal{W}$ of relation $\mathcal{R}$ for concept $\mathcal{C}$ is the ratio of $N.\mathcal{R}(\mathcal{C})$ and $N_{max}.\mathcal{R}(\mathcal{D})$, while $\mathcal{D}$ can be any concept in the KB.

Inference rules based on weight values:

$$F_r(C) = C \sqcup R_i.C \leftarrow \exists R_i.C : \mathcal{W}.R_i(C) \in [0,0.4] \quad (15)$$

while $i \geq 1$ and i is a natural number.

Rule (15) means, for concept C, if there exists a relation R_i while this relation's weight is in [0, 0.4], it inferences concept C with this particular role R_i is a valuable research idea candidate.

$F_r(C) = \emptyset \leftarrow \forall R_i.C: \mathcal{W}.R_i(C) \notin [0,0.4]$
(16)
while $i \geq 1$ and i is a natural number.

Rule (16) represents, for concept C, if the weight's values of all relation R_i are not in [0, 0.4], it inferences there is no valuable research idea.

$F_r(C,D) = C \sqcup D \sqcup R_i(C,D) \leftarrow \exists R_i(C,D): \mathcal{W}.R_i(C,D) \in [0,0.4]$
(17)
while $i \geq 1$ and i is a natural number.

Rule (17) means, for concepts C and D, if there exists a relation R_i between C and D while this relation's weight is valued in [0, 0.4], it inferences concept C and D with this particular role R_i is a valuable research idea candidate.

$F_r(C,D) = \emptyset \leftarrow \exists! R_i(C,D) \vee (\forall R_i(C,D): \mathcal{W}.R_i(C,D) \notin [0,0.4])$
(18)
while $i \geq 1$ and i is a natural number.

Rule (18) represents, for concepts C and D, if there does not exist a relation between C and D; or if weights of all relations between C and D are not valued in [0, 0.4], it inferences there is no valuable research idea based on measure of weight.

$F_r(C,D,E) = C \sqcup D \sqcup E \sqcup R_i(C,D) \sqcup R_i(D,E) \leftarrow \exists R_i(C,D): \mathcal{W}.R_i(C,D) \in [0,0.4] \wedge$
$\exists R_j(D,E): \mathcal{W}.R_j(D,E) \in [0,0.4]$
(19)
While $i \geq 1, j \geq 1$, i and j are natural number.

Rule (19) means, for three concepts C, D and E, if there exists relation R_i between C and D and relation R_j between D and E while these two relations' weights are valued in [0, 0.4], it inferences a valuable research idea candidate that is concept C, D and E with these two particular roles/relations R_i and R_j.

$F_r(C,D,E) = \emptyset \leftarrow \exists! R_i(C,D) \vee \exists! R_j(D,E) \vee (\forall R_i(C,D): \mathcal{W}.R_i(C,D) \notin [0,0.4]) \vee$
$(\forall R_j(D,E): \mathcal{W}.R_j(D,E) \notin [0,0.4])$
(20)
while $i \geq 1$ and i is a natural number.

Rule (20) says, for three concepts C, D and E, if one or more of following conditions are met, it inferences there is no valuable research idea based on measure of weight.

Conditions:

i. If there does not exist a relation R_i between C and D;
ii. If there does not exist a relation R_j between D and E;
iii. If weights of all relations between C and D are not valued in [0, 0.4]; or
iv. If weights of all relations between D and E are not valued in [0, 0.4].

6) Query Answering

To generate a set of idea candidates, a set of rules is designed for query answering. User's inputs and system's function are query; Created ideas are answer to the query. The following are query-answering rules for generating of ideas candidates I_s.

If input is $C\ or\ x \in C$, $I_s = F_r(C) \sqcup f_e(C) \sqcup F_r(f_t(C)) \sqcup f_e(f_t(C)) \sqcup F_r(f_c(C)) \sqcup f_e(f_c(C))$

(21)

Above rule shows, when input is a concept C or an instance x that belongs to C, generated idea candidates I_s are sum of inference results of $F_r(C)$, $f_e(C)$, $F_r\big(f_t(C)\big)$, $f_e(f_t(C))$, $F_r(f_c(C))$, and $f_e(f_c(C))$. $F_r(C)$ is inference based on weight value, details refers to rule (15). $f_e(C)$ generates ideas according to exploratory inference, refers to rule (1)-(4). $F_r\big(f_t(C)\big)$ applies rules (15) and (16) to the results generated by transformative inference rule (7). $f_e(f_t(C))$ is to apply exploratory inference rules (1) – (4) to results generated by transformative inference rule (7). $F_r(f_c(C))$ is applying rules (15) and (16) to the results generated by combinational inference rule (10). $f_e(f_c(C))$ generates idea candidates by applying exploratory inference rules (1-4) into results produced from combinational inference rule (10).

If inputs are $C\ and\ D$, or $x \in C\ and\ D$, or $x \in C\ and\ y \in D$, $I_s = F_r(C) \sqcup F_r(D) \sqcup F_r(C,D) \sqcup f_e(C,D) \sqcup F_r\big(f_t(C,D)\big) \sqcup f_e\big(f_t(C,D)\big) \sqcup F_r\big(f_c(C,D)\big) \sqcup f_e(f_c(C,D))$

(22)

Rule (22) means, for two valid inputs, its generated idea candidates, represented as I_s, are sum of inference results of $F_r(C)$, $F_r(D)$, $F_r(C,D)$, $f_e(C,D)$, $F_r\big(f_t(C,D)\big)$, $f_e\big(f_t(C,D)\big)$, $F_r\big(f_c(C,D)\big)$, and $f_e(f_c(C,D))$. There are ideas generated directly from inferences based on weights, exploratory creativity, transformative creativity and combinational creativity. In addition, there are ideas generated by combining different kind of inference rules.

If inputs are $C, D\ and\ E$, or $x \in C, D\ and\ E$, or $x \in C, y \in D\ and\ z \in E$,

$$I_s = F_r(C) \sqcup F_r(D) \sqcup F_r(E) \sqcup F_r(C,D) \sqcup F_r(C,E) \sqcup F_r(D,E) \sqcup F_r(C,D,E) \sqcup f_e(C,D,E)$$

$$\sqcup F_r\big(f_t(C,D,E)\big) \sqcup f_e\big(f_t(C,D,E)\big) \sqcup F_r\big(f_c(C,D,E)\big) \sqcup f_e(f_c(C,D,E)) \quad (23)$$

Rule (23) means, for three valid inputs, its generated idea candidates, represented as I_s, are sum of inference results of $F_r(C)$, $F_r(D)$, $F_r(E)$, $F_r(C,D)$, $F_r(C,E)$, $F_r(D,E)$, $F_r(C,D,E)$, $f_e(C,D,E)$, $F_r\big(f_t(C,D,E)\big)$, $f_e\big(f_t(C,D,E)\big)$, $F_r\big(f_c(C,D,E)\big)$, and $f_e(f_c(C,D,E))$. Some ideas are generated directly from inferences based on weights, exploratory creativity, transformative creativity and combinational creativity. Meanwhile, it generates other ideas by combining of different kinds of inference rules.

Above rules (21), (22) and (23) defined how idea candidates are generated under different circumstances. Specifically, different inference rules are applied and combined to support results' creativity and innovation.

7) Inference Example

This part aims to illustrate the proposed rules for the inference engine using a simple example. It is assumed there is a KB's Tbox shown as Figure 10, which contains concepts, i.e., A, A_1, etc., and roles/relations, i.e., R_1, R_2 and so on.

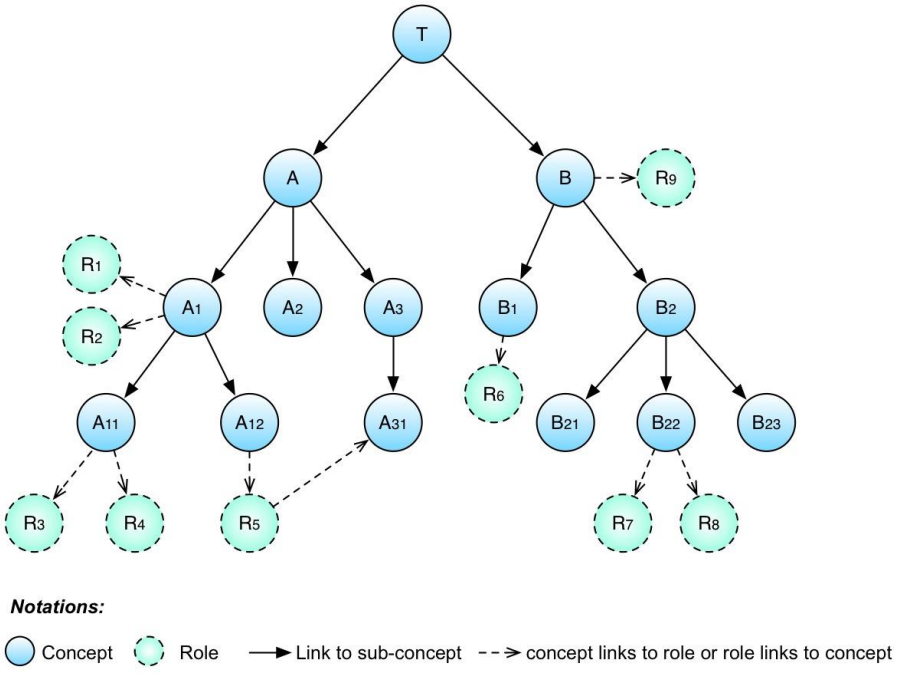


Figure 8. Example KB's TBox

The example KB is assumed $W(R_2) \in [0,0.4], W(R_3) \in [0,0.4]$, $W(R_8) \in [0,0.4]$. If input is A_1, idea candidate I_x will be generated based on proposed inference and querying answer rules. Specifically, there are 23 idea candidates generated for this example as listed below,

$I_1 = F_r(A_1) = R_2.A_1 \sqcup A_1$

$I_2 = f_e(A_1) = A_1 \sqcup R_6.A_1$

$I_3 = f_e(A_1) = A_1 \sqcup R_7.A_1$

$I_4 = f_e(A_1) = A_1 \sqcup R_8.A_1$

$I_5 = f_e(A_1) = A_1 \sqcup R_9.A_1$

$I_6 = F_r(f_t(A_1)) = R_3.A_{11} \sqcup A_{11}$

$I_7 = F_r(f_t(A_1)) = R_8.B_{12} \sqcup B_{12}$

$I_{8-12} = f_e(f_t(A_1)) = R_i.A_{11} \sqcup A_{11}$, While $i = 5,6,7,8,9.$

$I_{13} = f_e(f_c(A_1)) = R_5(A_1, A_2) \sqcup A_1 \sqcup A_2$

$I_{14} = f_e(f_c(A_1)) = R_5(A_1, A_3) \sqcup A_1 \sqcup A_3$

$I_{15} = f_e(f_c(A_1)) = R_5(A_1, A_{31}) \sqcup A_1 \sqcup A_{31}$

$I_{16} = f_e(f_c(A_1)) = R_5(A_1, B) \sqcup A_1 \sqcup B$

$I_{17-19} = f_e(f_c(A_1)) = R_5(A_1, B_j) \sqcup A_1 \sqcup B_j, while\ j = 1,2,3$

$I_{20-22} = f_e(f_c(A_1)) = R_5(A_1, B_k) \sqcup A_1 \sqcup B_k, while\ k = 21,22,23$

Ideas candidates= ΣI_{1-22}

To realise the proposed inference engine, it requires a way to transform designed algorithms and rules into executable codes. JESS is an acronym for Java Expert System Shell, which is a rule engine and scripting environment written entirely in Java language by Ernest Friedman-Hill [39]. It was originally inspired by the CLIPS expert system shell, but has grown into a complete, distinct Java-influenced environment of its own. JESS can be used to build Java applets and applications with the capacity to "reason" using knowledge supplied in the form of declarative rules. Therefore, it is able to translate proposed rules to realise the inference engine.

For instance, Table 3 shows JESS expression of rules (15), that is $F_r(C)$, applied in this example.

Table 3. JESS Code Example

Rule F_r translated into Jess code
(deftemplate A_1 "" (declare (ordered TRUE)))
(deftemplate A_{11} "" (declare (ordered TRUE)))
(deftemplate R_1 "" (declare (ordered TRUE)))
(deftemplate R_2"" (declare (ordered TRUE)))
(deftemplate R_3 "" (declare (ordered TRUE)))
(deftemplate R_4 "" (declare (ordered TRUE)))
(deffacts initialFacts
(male bill)
(female jane)
(female sally)
(parent bill sally)
(parent jane sally)
)
(defrule F_r
(concept ?X)
(role ?R_j.X)
(i.R∈ [0, 0.4] ?R_j)
=>
(printout t crlf "A study of "?R_j "to enhance"?X crlf)
)
(reset)
(facts)
(run)
(printout t crlf)

4. CASE STUDY: CRS

This section presents a project named Courses Recommendation System (CRS), to demonstrate and prove that proposed method is feasible to realise domain specific "ideation". CRS aims to create rich software for students to plot long-term futures. Specifically, its objective is to design and develop a software application that will allow users to explore possible future paths in education and career by navigating very large datasets.

4.1 Requirements of Creativity

The criteria used include subjects of study and personal interests, covered by following requirements. Requirements related to UI and other aspects are omitted here, so that it can show its need for creativity clearly.

R1: User should be shown suitable options based on his/her qualifications and interests so that he/she can choose the right course.
R2: User can enjoy the experience of using a feature that will guide he/she towards suitable courses/careers so that he/she can have fun while learning.
R3: User can see which course leads to specific careers so that he/she understands the implications of his/her choice.
R4: User's qualifications, experiences and interests will influence the options given to him/her so that the user can receive personalised recommendation results.

From full requirements of CRS, it is an 'idea-tion' process whilst the generated ideas should be recommendation of university or collage courses and corresponding reasons for the nominations. According to above requirements, it is not a traditional courses recommendation application but a new system that generates creativity results to provide users (students) surprising choices with understandable reasons. Thus, this research focuses upon the method to support the listed requirements; the proposed method is a creative recommendation engine with following contributions.

1) **A set of reasoning rules** based on proposed inference rules. This aims to achieve requirements *R1* and *R2*.
2) **A set of advanced reasoning rules** combined Creative Computing techniques, Description Logics and fuzzy description logic to support further creativity. For example, if a user provides his/her interest as a word 'sleeping', these rules help to investigate potential meanings on this "meaningless" answer rather than ignore it.
3) **A set of abstraction algorithms** based on abstraction techniques to get the proper information on courses and careers. There are some abstraction algorithms designed in our previous research [40] can be utilised. Useful information will be abstracted from raw data, whilst irrelevant data are bypassed. It is for the user requirement *R3*.
4) **A set of rules on weighting personal factors** (i.e. qualification, experiences, interests, etc.) to support courses inference process. These rules can be seen as a subset of reasoning rules in 1). It supporting to achieve user requirement *R4*.

4.2 Designed Approach and Prototype

Figure 9 is an overview approach of CRS. It starts with an UI showing some questions to answer. After user manually answered some or all questions, the creative recommendation engine is activated at backstage to generate nominations and reasons. Then the generated results should be shown to user. It can be an iterative process. If user is not satisfied with generated results, he/she can go back to start display and go through

this process again. User can try as many times as he/she want until there is a result he/she thinks good. If user is satisfied, he/she can choose a course from the recommended set and apply it on corresponding university website.

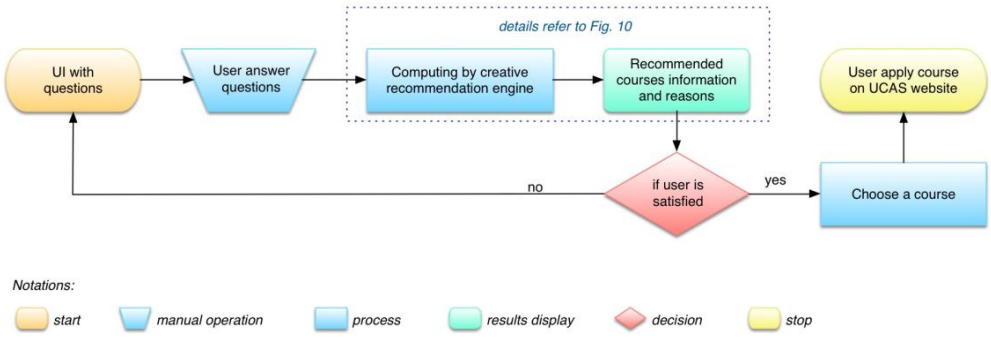


Figure 9. Overall Approach of CRS

The creative idea generation approach for CRS is designed as Figure 10 based on proposed method and approach in last section. It is detailed expression of creative recommendation engine from Figure 9, which focuses on presenting the role of creativity techniques and the whole approach as a Creative Computing application.

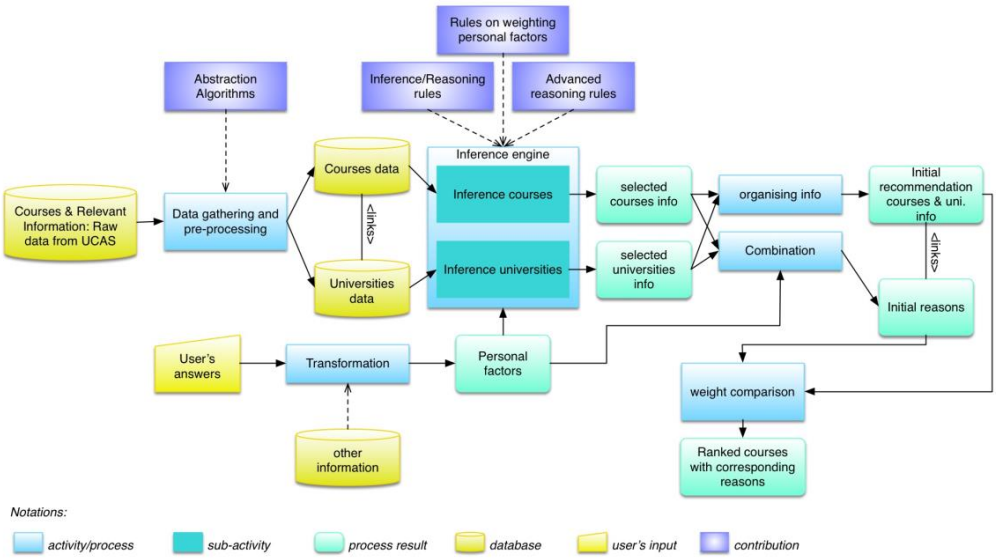


Figure 10. Creative Idea Generation Approach for CRS

The creative recommendation engine starts with gathering and pre-processing raw data. Abstraction algorithms are kernel support of this activity. Processed information should be saved as two database, courses database and universities database. There are links between them so that it is able to find which university a course belongs to. Then there is the most important part, inference engine, which contains activities to inference courses and universities. Personal factors are also inputs of the inference engine, which are

generated from user's answers via transformation activity. Through this process, certain users' answers will be transformed to new concepts to support inference activities as personal factors. The designed inference rules are kernel of this inference engine. In these rules, users' answers are keywords; Courses and universities information are domain knowledge base. Hence, the inference engine realise useful and creative reasoning. Also, there are rules designed for weighting personal factors employed in the inference engine. Information of selected courses and universities are organised as a part of initial recommendation results. Meanwhile, based upon selected courses and universities information, there is a combination activity to generate initial reasons and produce creativity in it. Finally, these courses and corresponding reasons are ranked via a weight comparison activity as eventual results of the recommendation engine.

Although this project is on the early stage, there is a prototype developed to represent how the application, especially the proposed creative recommendation engine, should work. Two screenshots of this prototype is listed below (Figure 11 and 12) to visualise users input and generated recommendation output.

Firstly, there are some questions displayed on the main interface. After a user answered some or all questions as Figure 11 and pressed search button, the creative recommendation engine starts working in the system background. Through designed creative idea generation approach, there is a set of courses with relevant information and reasons generated to display to the user as recommendations. For instance, Figure 12 shows there are four courses nominated with two very surprising candidates. As mentioned before, this whole approach can be an iterative process that allows users to search repeatedly until a desired course is found.

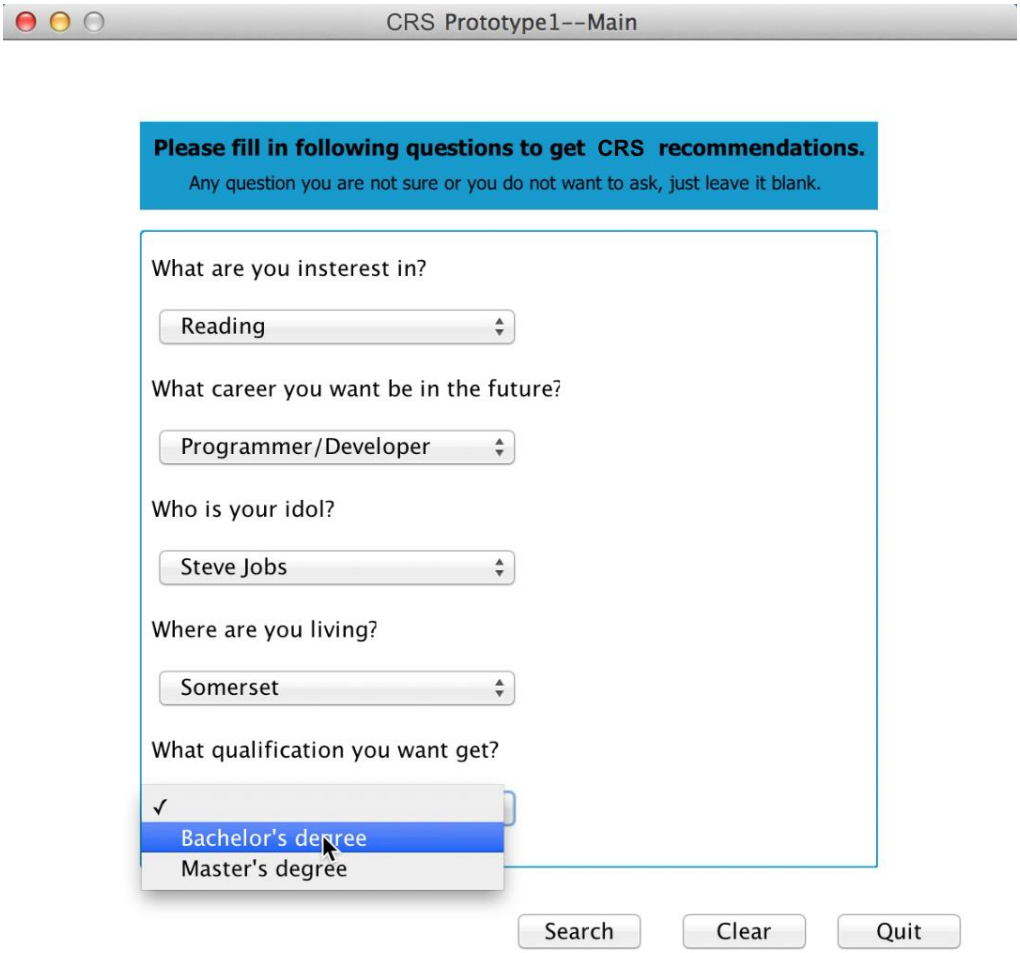


Figure 11. UI with Questions and Answers

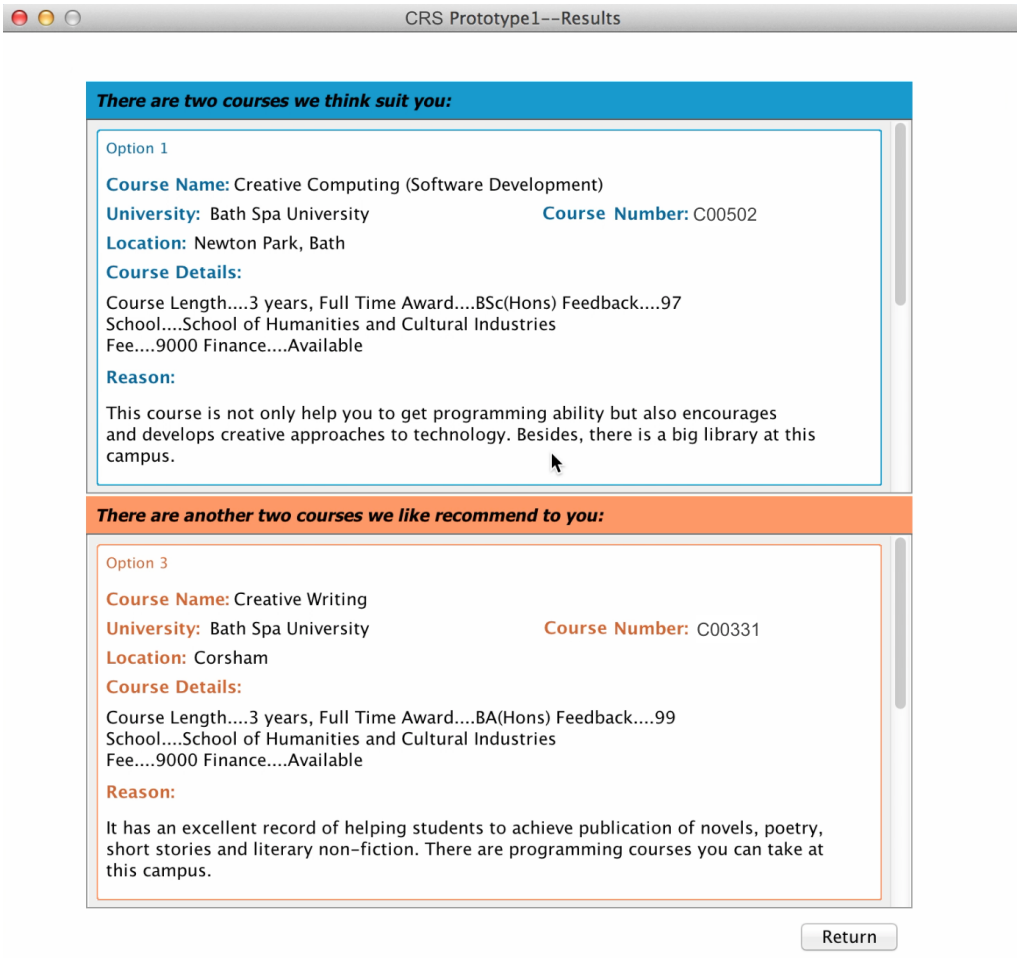


Figure 12. Recommendation Results

5. CONCLUSION

The aim of this paper was to discuss the realistic of domain-specific 'idea-tion'. The challenging issue it faces is how to generate new ideas. In other word, there is lack of works to make machine generated ideas novel, surprising and valuable. This research proposed a Creative Computing method and approach to realise domain-specific 'idea-tion'. Meanwhile, some creative techniques, including combination, exploration and transformation, were employed in the proposed inference rules to increase creativity. Furthermore, a project named Courses Recommendation System (CRS) was represented as use case to demonstrate and visualise the proposed approach's feasibility.

Overall, this research proved that domain-specific 'ideat-ion' could be realised by proposed Creative Computing method. Moreover, it can be deducted that there should be ways bring general machine 'idea-tion' from utopia to real possibility.

REFERENCES

[1] D. Graham and T. T. Bachmann, *Ideation: The Birth and Death of Ideas*. Wiley, 2004, pp. 1–240.

[2] B. Johnson, "Design Ideation: The Conceptual Sketch in the Digital Age", *Design Studies*, vol. 26, no. 6, 2005, pp. 613–624.

[3] K. Delic and J. Riley, "Current and Future Trends in AI", in *2013 XXIV International Symposium on Information, Communication and Automation Technologies (ICAT)*, 2013, pp. 31–34.

[4] D. L. Waltz, "The Future of AI Evolution , Sociobiology , and the Future of Artificial Intelligence", *IEEE Intelligent Systems*, vol. 21, no. 3, 2006, pp. 66–69.

[5] A. Yurchyshyna, M. Léonard and P. Brough-Heinzman, "Towards a Services-Based Approach for Supporting Idea Development Process", in *Fifth International Conference on Internet and Web Applications and Services (ICIW)*, 2010, pp. 321–326.

[6] R. Ellis, "AI Developers : Stand Up and Be Counted !", *IEEE Intelligent S*, vol. 23, no. 3, 2008, pp. 69–71.

[7] Y. Wang, "On Abstract Intelligence and Brain Informatics: Mapping the Cognitive Functions onto the Neural Architectures", in *IEEE 11th International Conference on Cognitive Informatics & Cognitive Computing*, 2012, pp. 5–6.

[8] W. T. O'Connor, "What Can the Brain Science of Learning Teach Us About Cybernetics?", in *IEEE 11th International Conference on Cybernetic Intelligent Systems (CIS)*, 2012, pp. 36–40.

[9] A. M. Truta, "Neural Nets Activity in Design Cognitive Model of Brain", in *the 7th International Conference on The Experience of Designing and Application of CAD Systems in Microelectronics*, 2003, pp. 118–123.

[10] S. Yuan and Y. Chen, "Semantic Ideation Learning for Agent-Based E-Brainstorming", *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 2, 2008, pp. 261–275.

[11] A. F. Osborn, *Applied Imagination: Principles and Procedures of Creative Problem-Solving*, 3rd ed. Charles Scribner's Sons, 1979, pp. 1–299.

[12] M. Diehl and W. Stroebe, "Productivity Loss in Idea-Generating Groups: Tracking Down the Blocking Effect", *Journal of Personality and Social Psychology*, vol. 61, 1991, pp. 391–403.

[13] M. Aiken, M. Vanjani and J. Paolillo, "A Comparison of Two Electronic Idea Generation Techniques", *Information and Management*, vol. 30, 1996, pp. 91–99.

[14] R. O. Briggs and B. A. Reinig, "Bounded Ideation Theory : A New Model of the Relationship Between Idea- quantity and Idea-quality during Ideation", in *40th Hawaii International Conference on System Sciences*, 2007, pp. 1–10.

[15] A. F. Osborn, *Applied Imagination: Principles and procedures of creative thinking*. New York, USA: Scribeners and Sons, 1963.

[16] M. M. Crossan, H. W. Lane and R. E. White, "An Organizational Learning Freamwork: From Intuition to Institution", *Academy of Management Review*, vol. 24, no. 3, 1999, pp. 522–537.

[17] T. M. Amabile, S. G. Barsade, J. S. Mueller and Barry Staw, "Affect and Creativity at Work", *Administrative Science Quarterly*, vol. 50, 2005, pp. 367–403.

[18] I. Nonaka and R. Toyama, "Knowledge Creation as a Synthesizing Process", in *Hitotsubashi on Knowledge Management*, H. Takeuchi and I. Nonaka, Eds. Singapore: John Wiley & Sons, 2003, pp. 125–151.

[19] C. Shalley and J. Perry-Smith, "The Emergence of Team Creative Cognition: the Role of Diverse Outside Ties, Sociocognitive Network Centrality, and Team Evolution", *Strategic Entrepreneurship Journal*, vol. 2, 2008, pp. 23–41.

[20] A. Hargadon and R. Sutton, "Technology Brokering and Innovation in a Product Development Firm", *Administrative Science Quarterly*, vol. 42, no. 4, 1997, pp. 716–749.

[21] C. Moser, "Exploring Ideation : Knowledge Development in Science Through the Lens of Semantic and Social Networks", in *46rd Hawaii International Conference on System Sciences (HICSS)*, 2013, pp. 235–243.

[22] S. W. Knoll and G. Horton, "Changing the Perspective : Improving Generate thinkLets for Ideation", in *43rd Hawaii International Conference on System Sciences (HICSS)*, 2010, pp. 1–10.

[23] U. A. Jogalekar and A. Mangla, "Idea Generation Algorithm Based Systems", in *3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, 2010, pp. 14–17.

[24] S. Wu and W. Fang, "The Effect of Consumer-to-Consumer Interactions on Idea Generation in Virtual Brand Community Relationships", *Technovation*, vol. 30, no. 11–12, Nov. 2010, pp. 570–581.

[25] C. M. Crawford and C. A. DiBenedetto, *New Products Management*, 10th ed. Boston: McGraw-Hill, 2010, pp. 1–592.

[26] S. Majaro, *Managing Ideas for Profit: The Creative Gap (Marketing for Professionals)*. London: McGraw-Hill, 1991, pp. 1–360.

[27] A. M. Webb and A. Kerne, "Integrating Implicit Structure Visualization with Authoring Promotes Ideation", in *the 11th Annual International ACM/IEEE joint conference on Digital Libraries*, 2011, pp. 203–212.

[28] A. Goncher, A. Johri, S. Kothaneth, V. Lohani and V. Tech, "Exploration and Exploitation in Engineering Design : Examining the Effects of Prior Knowledge on Creativity and Ideation", in *39th IEEE Frontiers in Education Conference*, 2009, pp. 1–7.

[29] J. Zhang, J. Ma, D. Zhang, R. Tan and A. K. Source, "CAI-Driven New Ideas Generation for Product Conceptual Design", in *IEEE International Conference on Management of Innovation and Technology (ICMIT)*, 2012, pp. 824–830.

[30] J. Goldenberg, D. Mazursky and S. Solomon, "Creative Sparks", *Science*, vol. 285, 1999, pp. 1495–1496.

[31] A. Hugill, "Creative Computing Processes : Musical Composition", in *IEEE 8th International Symposium on Service Oriented System Engineering*, 2014, pp. 459–464.

[32] M. A. Boden, "Artificial Intelligence Creativity and artificial intelligence", *Artificial Intelligence*, vol. 103, no. 1–2, 1998, pp. 347–356.

[33] M. A. Boden, *The Creative Mind: Myths and Mechanisms*, 2nd ed. London: Routledge, 2003, pp. 1–344.

[34] A. Koestler, *The Act of Creation*. London: Hutchinson, 1964, pp. 1–751.

[35] D. N. Perkins, *The Mind's Best Work*. Cambridge, MA: Harvard University Press, 1981, pp. 58–324.

[36] E. De Bono, *Lateral Thinking: Creativity Step by Step*. New York: Harper & Row, 1970, pp. 23–272.

[37] R. Liu and H. Yang, "Chaos and Fractal for Creative Computing", in *IEEE 8th International Symposium on Service Oriented System Engineering*, 2014, pp. 470–473.

[38] A. Hugill and H. Yang, "The Creative Turn : New Challenges for Computing", *International Journal of Creative Computing*, vol. 1, no. 1, 2013, pp. 4–19.

[39] E. F. Hill, *Jess in Action: Jave Rule-Based System*. Greenwich, CT, USA: Manning Publications Co., 2003, pp. 1–12.

[40] D. Jing, H. Yang and Y. Tian, "Abstraction Based Domain Ontology Extraction for Idea Creation", in *13th International Conference on Quality Software*, 2013, pp. 341–348.

Applied Science Journal

Volume 1(1) (2016)

AsJ

RESEARCH ARTICLE

Time Series Analysis for a $1/t^{\beta}$ Memory Function and Comparison with the Lyapunov Exponent using Volatility Scaling

J. Blackledge

University of KwaZulu-Natal, Durban, South Africa

P. Walsh

Dublin Institute of Technology, Dublin, Ireland

Being able to provide accurate forecasts on the trending behaviour of time series is important in a range of applications involving the real-time evolution of signals, most notably in financial time series analysis, but control engineering in general. This paper reports on the use of an indicator that is based on a Memory Function of the form $\sim 1/t^{\beta}$, $\beta > 0$, and, in terms of a comparative analysis, the Lyapunov Exponent λ coupled with an approach whereby both parameters (i.e. λ and $\beta - 1$) are scaled according to the corresponding Volatility σ of the time series. A 'back-testing' procedure is used to evaluate and compare the performance of the indices $(\beta - 1)/\sigma$ and λ/σ for forecasting and quantifying trends over a range of time scales. However, in either case, a critical solution for providing high accuracy forecasts is the filtering operation used to identify the position in time at which a trend occurs subject to a time delay factor that is inherent in the filtering strategy used. The paper explores this strategy and presents some example results that provide a quantitative measure of the accuracy obtained.

Categories and Subject Descriptors: []:

General Terms: [], []

Additional Key Words and Phrases: Financial Time Series Analysis, Generalised Kolmogorov-Feller Equation, Memory Functions, Lyapunov Exponent, Volatility Scaling, Filtering Strategies.

1. INTRODUCTION

Continuous Time Random Walk models are important in developing algorithms for both simulating and analysing time series data. Most models of this type are based on Einstein's evolution equation, and, on the basis of this equation, we consider a continuous time solution based on a Memory Function of the form $1/t^{\beta}$ and show that the first order

Correspondence address: University of KwaZulu-Natal, Durban, South Africa

fundamental solution is equivalent to considering a Lévy distributed system with a $\delta(t)$ memory function. This is the subject of Section 2 and 3, respectively. The purpose of this is to understand the origins of parameters such as $\beta > 0$, the Lévy index $\gamma \in (0,2]$ and, for comparison, the Lyapunov Exponent λ (which is briefly considered in Section 3), for example, in terms of their use as indicators in regard to the analysis of an evolving time series. In this context, we explore the use of these parameters for forecasting both the type and stability of time series trends by scaling them with the corresponding Volatility. For this purpose, we provide a short derivation of the short time Volatility in Section 4 using a 'Phase Only Condition'. Coupled with a novel filtering strategy, it is shown that the key to forecasting the onset of time series trends are the zero crossings associated with $\beta - 1$, $(1/\gamma) - 1$ and λ, in the latter case, for example, the demarcation between an upward and a downward trend being predicated on whether $\lambda > 0$ or $\lambda < 0$, respectively. This is discussed in Section 5 which includes example results on trend forecasting for an energy commodities time series and quantifies the accuracy of the forecasts obtained using a back-testing procedure presented in Section 5.1.

The work reported in this paper can be contextualised in terms of the application of memory functions to complex systems analysis. In complex systems, the elements adapt to the aggregate pattern they co-create. As the components react, the aggregate changes, as the aggregate changes the components react anew. Barring the reaching of some asymptotic state or equilibrium, complex systems keep evolving to producing stochastic fields. Such systems arise naturally in an economy. Economic agents, be they banks, firms, or investors, continually adjust their market strategies to the macroscopic economy which their collective market strategies create. It is important to appreciate that there is an added layer of complexity within the economic community. Unlike many physical systems, economic elements (human agents) react with strategy and foresight by considering the implications of their actions, i.e. the decisions taken are subject to risk management and although we can not be certain whether this fact changes the resulting behaviour, we can be sure that it introduces feedback which is basis for many complex systems models that generate chaotic fields with self-affine structures. This foresight is based to a certain extent on the 'memory' of past events and hence, the concept of defining a 'memory function' to analyse a complex system has a natural synergy with the underlying issues associated with modelling complex systems.

Complex systems can be split into two categories: equilibrium and non-equilibrium. Equilibrium complex systems, undergoing a phase transition, can lead to 'critical states' that often exhibit random self-affine structures in which the statistics of the stichastic field are scale invariant. Non-equilibrium complex systems give rise to 'self organised critical states' and financial markets can be considered to be non-equilibrium systems because they are constantly driven by transactions that occur as the result of new economic information over a range of time scales. They are complex systems because the market also responds to itself, often in a highly non-linear fashion, and would carry on doing so (at least for some time) in the absence of new information. The 'price change field' is highly non-linear and very sensitive to exogenous shocks and it is probable that all shocks have a long term effect for which a long term memmory function may be sought. Market transactions generally occur globally at the rate of hundreds of thousands per second. It is the frequency and nature of these transactions that dictate the behaviour of stock market indices nearly all of which are statistically self-affine.

2. THE GENERALISED KOLMOGOROV-FELLER EQUATION

For a Probability Density Function (PDF) $p(x)$, Einstein's evolution equation is, [1]

$$u(x,t+\tau) = u(x,t) \otimes_x p(x) \tag{1}$$

where $u(x,t)$ is a 'density function' representing the concentration of a canonical ensemble of particles undergoing elastic collisions and $\otimes_x$ denotes the non-causal convolution integral. For arbitrary values of τ, Taylor expansion allows us to write

$$\tau\frac{\partial}{\partial t}u(x,t) + \frac{\tau^2}{2!}\frac{\partial^2}{\partial t^2}u(x,t) + ... \equiv \tau m(t) \otimes_t \frac{\partial}{\partial t}u(x,t) = -u(x,t) + u(x,t) \otimes_x p(x) \tag{2}$$

where $m(t)$ is a Memory Function and $\otimes_t$ is taken to denote the causal convolution integral over t, convolution with a memory function replacing the infinite series representation through Taylor expansion of the function $u(x,t+\tau)$. Equation (2) is the Generalised Kolomogorov-Feller Equation (KFE) which reduces to the Classical KFE when $m(t) = \delta(t)$ and is equivalent to the case of considering $\tau << 1$ in the Taylor series expansion of equation (1), [2], [3].

2.1 Othonormality

For any memory function for which there exists a function or class of functions of the type $n(t)$, say, such that $n(t) \otimes_t m(t) = \delta(t)$ we can write equation (2) in the form

$$\tau\frac{\partial}{\partial t}u(x,t) = -n(t) \otimes_t u(x,t) + n(t) \otimes_t u(x,t) \otimes_x p(x)$$

where the Classical KFE is recovered when $n(t) = \delta(t)$. Any solution obtained to the Generalised KFE is dependent upon the choice of memory function $m(t)$ used. There are a number of choices that can be considered, each of which is taken to be a 'best characteristic' of a stochastic time series in terms of the influence of its time history. However, it may be expected that the time history of physically significant random systems is relatively localised in time.

We consider a memory function of the type [4]

$$m(t) = \frac{1}{\Gamma(1-\beta)t^{\beta}}, \quad \beta > 0$$

where

$$n(t) = \frac{1}{\Gamma(\beta-1)t^{2-\beta}}$$

given that

$$\int_0^{\infty} \frac{\exp(-st)}{\Gamma(\beta)t^{1-\beta}}dt = \frac{1}{s^{\beta}}$$

and

$$\int_0^{\infty} \delta(t)\exp(-st)dt = 1$$

2.2 Fundamental (Green's Function) Solution

By writing equation (2) in the form

$$\tau\frac{\partial}{\partial t}u(x,t)+u(x,t)=u(x,t)-n(t)\otimes_t u(x,t)+n(t)\otimes_t u(x,t)\otimes_x p(x)$$

the Green's function solution is given by

$$u(x,t)=g(t)\otimes_t u(x,t)-g(t)\otimes_t n(t)\otimes_t u(x,t)+g(t)\otimes_t n(t)\otimes_t u(x,t)\otimes_x p(x), \qquad (3)$$

the Green's function being given by

$$g(t)=\frac{1}{\tau}\exp(-t/\tau), \quad t>0$$

which is the solution to

$$\tau\frac{\partial}{\partial t}g(t)+g(t)=\delta(t)$$

assuming initial conditions $u(x,0)=0$ and $g(0)=0$.

Theorem 2.1

If the Laplace transform of the function $n(t)$ exists, then a solution of equation (3) is

$$u(x,t)=h(t)\otimes_t u(x,t)\otimes_x p(x), \quad h(t)\leftrightarrow\frac{\bar{n}(s)}{\tau s+\bar{n}(s)}$$

where $\leftrightarrow$ denotes the Laplace transformation, i.e. the mutual transformation from t-space to s-space.

Proof of Theorem 2.1

Using the convolution theorems for the Fourier and Laplace transforms, respectively, equation (3) can be written as

$$\bar{\tilde{u}}(k,s)=\bar{g}(s)\bar{\tilde{u}}(k,s)-\bar{g}(s)\bar{n}(s)\bar{\tilde{u}}(k,s)+\bar{g}(s)\bar{n}(s)\bar{\tilde{u}}(k,s)\tilde{p}(k)$$

where

$$\bar{\tilde{u}}(k,s)=\int_0^\infty\int_{-\infty}^{\infty}u(x,t)\exp(-ikx)dx\exp(-st)dt, \quad \bar{g}(s)=\int_0^\infty g(t)\exp(-st)dt,$$

$$\bar{n}(s)=\int_0^\infty n(t)\exp(-st)dt$$

and

$$\tilde{p}(k)=\int_{-\infty}^{\infty}p(x)\exp(-ikx)dx$$

Thus, noting that $\bar{g}(s)=(1+\tau s)^{-1}$, we can write

$$\bar{\tilde{u}}(k,s)=-\frac{\bar{g}(s)}{1-\bar{g}(s)}\bar{n}(s)\bar{\tilde{u}}(k,s)+\frac{\bar{g}(s)}{1-\bar{g}(s)}\bar{n}(s)\bar{\tilde{u}}(k,s)\tilde{p}(k)$$

$$= -\frac{\bar{n}(s)}{\tau s}\bar{\tilde{u}}(x,t) + \frac{\bar{n}(s)}{\tau s}\bar{\tilde{u}}(k,s)\tilde{p}(k) = \bar{h}(s)\bar{\tilde{u}}(k,s)\tilde{p}(k)$$

which, upon inverse transformation yields

$$u(x,t) = h(t) \otimes_t u(x,t) \otimes_x p(x) \tag{4}$$

Theorem 2.2

For an initial solution $u_0(x,t)$, the convergent N^{th}-order iterative solution of equation (4) is

$$u_N(x,t) = j = 1Np(x)k = 1Nh(t) \otimes_x \otimes_t u_0(x,t), \quad \|h(t)\|_2 \times \|p(x)\|_2 < \frac{1}{\sqrt{2\pi}} \tag{5}$$

where

$$j = 1Nf(t) \equiv f(t) \otimes_t f(t) \otimes_t f(t) \otimes_t ...$$

denotes the N^{th} convolution of $f(t)$.

Proof of Theorem 2.2

Consider an iteration of equation (4) as defined by

$$u_{n+1}(x,t) = h(t) \otimes_t u_n(x,t) \otimes_x p(x), \quad n = 1,2,...,N$$

Using the convolution theorem, the equivalent iteration in Fourier-Laplace space is

$$\bar{\tilde{u}}_{n+1}(k,s) = \bar{h}(s)\bar{\tilde{u}}_n(k,s)\tilde{p}(k)$$

for initial solution $\bar{\tilde{u}}_0(k,s)$. Thus, after N iterations, we can write

$$\bar{\tilde{u}}_N(k,s) = [\bar{h}(s)]^N[\tilde{p}(k)]^N\bar{\tilde{u}}_0(k,s)$$

so that upon inverse Fourier-Laplace transformation, the result is obtained.

The criterion for convergence as stated in Theorem 2.2 is obtained by considering an n^{th} order error function $\varepsilon_n(x,t)$ so that $u_n(x,t) = u(x,t) + \varepsilon_n(x,t)$. We can then write

$$\bar{\tilde{\varepsilon}}_{n+1}(k,s) = \bar{h}(s)\tilde{p}(k)\bar{\tilde{\varepsilon}}_n(k,s)$$

so that

$$\bar{\tilde{\varepsilon}}_n(k,s) = [\bar{h}(s)\tilde{p}(k)]^n\bar{\tilde{\varepsilon}}_0(k,s)$$

and it is clear that, since we require $\bar{\tilde{\varepsilon}}_n \to 0$ and $n \to \infty$, $[\bar{h}(s)\tilde{p}(k)] < 1 \quad \forall(k,s)$. The condition for convergence therefore becomes

$$\|\bar{h}(s)\tilde{p}(k)\| \leq \|\bar{h}(s)\| \times \|\tilde{p}(k)\| < 1$$

or, for Euclidian norms, and, using Rayleigh's theorem,

$$\|\bar{h}(s)\|_2 \times \|p(x)\|_2 < \frac{1}{\sqrt{2\pi}}$$

In (k,t)-space

$$\tilde{\varepsilon}_n(k,t) = k = 1nh(t)[\tilde{p}(k)]^n \otimes_t \tilde{\varepsilon}_0(k,t)$$

so that, using Hölder's inequality,

$$\|\widetilde{\varepsilon}_n(k,t)\| \leq \|k = 1nh(t)[\widetilde{p}(k)]^n\| \times \|\widetilde{\varepsilon}_0(k,t)\| \leq \|h(t)\|^n \times \|\widetilde{p}(k)\|^n \times \|\widetilde{\varepsilon}_0(k,t)\|$$

from which the condition for convergence is thus derived.

2.3 First Order Impulse Response Function

Form equation (5), if the initial solution is an impulse, i.e. $u_0(x,t) = \delta(x)\delta(t)$, then the Impulse Response Function (IRF), denoted by $R_N(x,t)$, say, is given by

$$R_N(x,t) = j = 1Np(x)k = 1Nh(t)$$

with 'transfer function'

$$\widetilde{\bar{R}}_N(k,s) = [\bar{h}(s)\widetilde{p}(k)]^N$$

The first order IRF is

$$R_1(x,t) = p(x)h(t)$$

and it is then clear that the 'space integrated IRF' is given by $h(t)$, i.e.

$$h(t) = R_1(x,t)dx, \quad p(x)dx = 1$$

For memory function

$$m(t) \leftrightarrow \frac{1}{s^{1-\beta}}, \quad \bar{h}(s) = \frac{1}{1+\tau s^{\beta}} \sim \frac{\tau}{\tau s^{\beta}}, \quad \tau >> 1 \Rightarrow h(t) \sim \frac{1}{\tau\Gamma(\beta)t^{1-\beta}}$$

Thus, if we consider an initial solution $u_0(x,t) = \delta(x)r(t)$ where $r(t)$ is some random varying function, then we arrive at a model for $R(t)$ given by (ignoring scaling by $1/\tau$)

$$R(t) = \frac{1}{\Gamma(\beta)t^{1-\beta}} \otimes_t r(t)$$

This model is based on the Riemann-Liouville integral transform which has self-affine properties, properties that exhibit 'Stochastic Trending Characteristics'. In other words $R(t)$ defines a random fractal function whose IRF is $1/t^{1-\beta}$, a result that is, in light of the above analysis, been shown to be a PDF independent first order solution to the Generalised KFE for a memory function of the type (ignoring scaling) $1/t^{\beta}$.

2.4 Application to Stochastic Time Series Analysis

By considering a short time model for a time series which is predicated on the scaling law t^{α}, $\alpha = \beta - 1$, regressions methods can be used to estimate the parameter α on a moving window basis, a least squares estimate, for example, being given by, for a uniformly sampled time series $R(t_i) > 0 \forall t_i, \quad i = 1,2,...,N$

$$\alpha = \frac{\sum_{i=1}^{N} \ln R(t_i) \sum_{i=1}^{N} \ln t_i - N \sum_{i=1}^{N} \ln R(t_i) \ln t_i}{\left(\sum_{i=1}^{N} \ln t_i\right)^2 - N \sum_{i=1}^{N} \ln t_i^2}$$

In this context, and, from the point of view detecting trends in time series data $R(t_i)$, $\alpha > 0$ indicates a positive upward trend and $\alpha < 0$ indicates a negative downward trend.

The case when $\alpha \sim 0$ is an indication a non-trending behaviour. The transition points between the start and end of any given trend can therefore be identified by those positions in time at which $\alpha \sim 0$ through the application of a moving window process where an estimate of is computed at each position of the windowed data.

2.5 Equivalence with the Lévy Index

Consider a (symmetric) Lévy distribution $p(x)$ defined in terms of the Characteristics Function

$$\widetilde{p}(k) = \exp(-a \mid k \mid^{\gamma}) \sim 1 - a \mid k \mid^{k}, \quad \gamma \in (0,2]$$

where $a << 1$ is a constant and γ is the 'Lévy index'. Application of a first order Taylor series to equation (1) under the condition that $\tau << 1$ and application of the Convolution Theorem then yields the Fractional Diffusion Equation

$$\left(\frac{\partial^{\gamma}}{\partial \mid x \mid^{\gamma}} - \frac{\partial}{\partial t} \right) u(x,t) = 0; \quad \frac{\partial^{\gamma}}{\partial \mid x \mid^{\gamma}} u(x,t) = \frac{1}{2\pi} \widetilde{u}(k,t) \mid k \mid^{\gamma} \exp(ikx) dk$$

where $\tau := \tau / a = 1$ (for normalised units).

Theorem 2.3

The Green's function for this equation has the operational form

$$g(x,t) = \frac{i}{2\Gamma(1/\gamma) t^{1-1/\gamma}} \otimes_t \left[\delta(t) + \sum_{n=1}^{\infty} \frac{(ix)^n}{n!} \frac{\partial^{n/\gamma}}{\partial t^{n/\gamma}} \right] = \frac{i}{2\Gamma(1/\gamma) t^{1-1/\gamma}}, \quad x \to 0$$

Proof of Theorem 2.2

For the source function $-\delta(x,t)$, computation of the Green's function by Fourier and Laplace transformation in x-space and t-space, respectively, yields the result (for the pole at $\mid k \mid = s^{1/\gamma}$)

$$g(x,t) = \frac{1}{2\pi i} \int_{-i\infty}^{i\infty} \frac{i \exp(i s^{1/\gamma} x)}{2 s^{1/\gamma}} \exp(st) ds$$

Thus, by writing

$$\exp(i s^{1/\gamma} x) = 1 + \sum_{n=1}^{\infty} \frac{(ix)^n}{n!} s^{n/\gamma}$$

the result of obtained, given that

$$s^{n/\gamma} \leftrightarrow \frac{\partial^{n/\gamma}}{\partial t^{n/\gamma}}$$

This result shows that the IRF for a Lévy distributed system is determined by the function $1/t^{1-1/\gamma} \quad t > 0$ for $x \to 0$, and, in this sense, it is clear that $\gamma = 1/\beta$ for the range $1/2 \leq \beta < \infty$.

3. THE LYAPUNOV EXPONENT AND KOLMOGOROV-SINAI ENTROPY

For a set of discrete time steps $t_n, \quad n = 1,2,...$, equation (1) can be considered to be an Iteration Function System defined as

$$u(x_m,t_{n+1}) = p(x_m) \otimes_x u(x_m,t_n)$$

where the density function u is also cast in terms of a set of discrete steps in space $x_m, \quad m = 1,2,...$ and $\otimes_x$ is taken to denote the 'Convolution Sum'. Suppose that after many time steps, this iteration converges to the function $\phi(x_m,t_\infty)$, say. We can then represent the iteration in the form

$$u(x_m,t_{n+1}) = \phi(x_m,t_\infty) + \varepsilon(x_m,t_n)$$

where $\varepsilon(x_m,t_n)$ denotes the error at any time step n. Convergence to the function $\phi(x_m,t_\infty)$ then occurs if $\varepsilon(x_m,t_n) \to 0 \forall m$ as $n \to \infty$.

Consider a 'model' for the error at each time step given by (for some real constant ε)

$$\varepsilon(x_m,t_{n+1}) = \varepsilon \exp(t_n\lambda) \Rightarrow \varepsilon(x_m,t_{n+1}) = \varepsilon(x_m,t_n)\exp(\lambda)$$

We can then construct an equation for λ given by

$$\lambda = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \log \frac{\bar{\varepsilon}(t_{n+1})}{\bar{\varepsilon}(t_n)} \quad \textit{where} \quad \bar{\varepsilon}(t_n) = \lim_{M\to\infty} \frac{1}{M} \sum_{n=1}^{M} \varepsilon(x_m,t_n)$$

If λ is negative, then the iterative process is stable since we can expect that as $N \to \infty$, $\bar{\varepsilon}(t_{n+1})/\bar{\varepsilon}(t_n) < 1$ and thus $\log[\bar{\varepsilon}(t_{n+1})/\bar{\varepsilon}(t_n)] < 0$. However, if λ is positive then the iterative process will diverge. This criterion for convergence/divergence is of course dependent on the exponential 'model' used to represent the error function at each iteration, and, within this context, λ is known as the Lyapunov Exponent. Note that we have purposely derived an expression for this exponent in regard to the evolution equation - equation (1) - in order to demonstrate the connectivity between the parameter λ and the parameters $\beta - 1$ and $1 - 1/\gamma$ (as discussed in Sections 2.4 and 2.5, respectively). In this sense, the 'unifying framework' for all such parameters is equation (1).

3.1 The Lyapunov Exponent

In a general context, The Lyapunov Exponent, denoted by $\lambda(t_0)$ for some initial condition in time t_0, is a quantitative measure of the exponential divergence of trajectories starting from the neighbourhood of t_0 [5]. For a one-dimensional system, i.e. a time series function $f(t)$, say,

$$f_n(t_0+\varepsilon) - f_n(t_0) = \varepsilon \exp[n\lambda(t_0)],$$

where ε is a small perturbation from the initial condition t_0 and n is the number of iterations.

Generally, λ depends on the initial condition, and thus we can only estimate its average value. In a measure-preserving system λ is constant for all trajectories and is given by the limit

$$\lambda(t_0) = \lim_{n\to\infty} \lim_{\varepsilon\to 0} \frac{1}{n} \log \frac{f_n(t_0+\varepsilon) - f_n(t_0)}{\varepsilon}$$

or

$$\lambda(t_0) = \lim_{n\to\infty} \frac{1}{n} \sum_{k=1}^{n} \log f'(t_k) = \lim_{n\to\infty} \frac{1}{n} \log \prod_{k=1}^{n} f'(t_k)$$

For each k, $f'(t_k)$ tells us how much the function f is changing with respect to its argument at the point t_k. This derivative expresses the magnitude of change in the transition from t_k to t_{k+1}. The limit of the average of the derivative logarithms over n iterations is taken to provide a measure of how fast the orbit changes as (discrete) time propagates. In the context of a discrete stochastic time series $R(t_1), R(t_2), ..., R(t_N)$, computation of the exponent is typically undertaken using the result

$$\lambda = \frac{1}{N} \sum_{n=1}^{N} \log \frac{R(t_{n+1})}{R(t_n)} \tag{6}$$

and provides a method of indicating the trending behaviour of a time series where positions in time at which the exponent crosses zero are an indication of the transition points between the start and end of a trend.

3.2 Kolmogorov-Sinai Entropy and Lyapunov Exponents

For a d-dimensional system we have a set $\lambda = \{\lambda_1, \ldots, \lambda_d\}$ and more complex behaviour, but still qualitatively the same as the one dimensional case [6]. However, a more accurate measure is the Kolmogorov-Sinai (KS) entropy because it considers the resolution (the precision) under which the system is observed. The Lyapunov Exponents measure how fast we loose the capability to predict the behaviour of a stochastic system. The disadvantage is that this measure does not consider the resolution under which the system is observed, unlike the Kolmogorov-Sinai entropy [5], [7] and [8].

Let the partition $\beta = \{T_1, T_2, ..., T_m\}$ be the observer's resolution. Looking at the system state t, the observer can only determine the fact that $t \in T_i$ and reconstruct the symbolic trajectory $\alpha_n = \{s_{m_1}, s_{m_2}, \ldots, s_{m_n}\}$ corresponding to the regions visited. The entropy of a trajectory α_n with respect to partition β is given by

$$H_n^{\beta} = -\sum_{\alpha_n} \Pr(\alpha_n) \log_{\{A\}} \Pr(\alpha_n)$$

where $\Pr(\alpha_n)$ is the probability of occurrence of the substring α_n. The conditional entropy of the $(n+1)^{th}$ symbol provided the previous n symbols are known is defined as

$$h_n^{\beta} = h_{n+1|n^{\beta}} = \left\{ \begin{array}{ll} H_{n+1^{\beta}} - H_{n^{\beta}}, & n \geq 1; \\ H_{1\beta}, & n = 1. \end{array} \right\}.$$

The entropy for a partition β is given by

$$h^{\beta} = \lim_{n\to\infty} h_n^{\beta} = \lim_{n\to\infty} \frac{1}{n} H_n^{\beta}$$

and the Kolmogorov-Sinai entropy is the supremum over all possible partitions

$$h_{KS} = \sup_{\beta} h^{\beta}$$

α-Index	Levy Index	Lyapunov Exponent	Hurst Exponent	Fractal Dimension
$\alpha = 0$	$\gamma = 2$	$\lambda = 0$	$H = \frac{1}{2}$	$D = 1.5$
$\alpha > 0$	$\gamma < 1$	$\lambda > 0$	$H > \frac{1}{2}$	$D < 1.5$
$\alpha < 0$	$\gamma > 1$	$\lambda < 0$	$H < \frac{1}{2}$	$D > 1.5$

Table I: Equivalence of the numerical ranges associated with the α-parameter, the Lévy index ($0 < \gamma \leq 2$) and the Lyapunov Exponent with the Hurst exponent ($0 < H < 1$) and the Fractal Dimension ($1 < D < 2$). .

The KS entropy is zero for regular systems, is finite and positive for a deterministic chaos and infinite for a random process and is related to the Lyapunov exponents by

$$h_{KS} = \sum_{1 \leq d \leq D} \lambda_d$$

being proportional to the time horizon T on which the system is predictable.

The behaviour of the Lyapunov Exponent when applied to a time series is similar to that of the 'α-index' defined in Section 2.4. Both parameters are related to other metrics associated with the fields of non-Gaussian stochastic systems in regard to the equivalence of the numerical ranges to be expected that differentiate between persistent and anti-persistent behaviour. In this context, Table 1 quantifies the comparative ranges of some selected metrics (subject to their upper and lower bounds).

4. THE VOLATILITY AND VOLATILITY SCALING

Consider the short time rate equation

$$f(t) = \frac{d}{dt} \ln R(t) = \sigma u(t)$$

where σ is the 'Volatility'. The Volatility is a measure of the randomness associated with $R(t)$ and we therefore require an estimate for σ in terms of the function $R(t)$ through the elimination $u(t)$. This requires a condition estimate of σ to be formulated.

4.1 Phase Only Condition

Theorem 4.1

If $u(t)$ is a phase only function with unit amplitude, bandwidth Ω and compact support T then

$$\sigma = \sqrt{\frac{2\pi}{\Omega}} \| f(t) \|_2, \quad \| f(t) \|_2 := \left(\int_{-T/2}^{T/2} | f(t) |^2 \, dt \right)^{\frac{1}{2}} \tag{7}$$

Proof of Theorem 4.1

If $u(t)$ is a phase only function (assuming unit amplitude), then

$$\widetilde{u}(\omega) = \exp[i\theta(\omega)], \quad \sigma \| u(t) \|_2 = \| f(t) \|_2$$

where $\omega(\omega)$ is the 'Phase Spectrum'. Using Parseval's Theorem, we have

$$\int_{-T/2}^{T/2} |\, u(t)\,|^2\, dt = \frac{1}{2\pi} \int_{-\Omega/2}^{\Omega/2} |\, \widetilde{u}(\omega)\,|^2\, d\omega = \frac{\Omega}{2\pi}$$

from which the result is thus derived.

Theorem 4.2

For a uniformly sampled discrete function $f_(t_n), \quad n = 1,2,3,...,N$, equation (7) becomes

$$\sigma = \frac{2\pi}{\Omega}(\|f(t_n)\|_2, \quad \|f(t_n)\|_2 := \left(\sum_{n=1}^{N} |\, f(t_n)\,|^2\right)^{\frac{1}{2}}$$

Proof of Theorem 4.2

For a uniform sampling interval of Δt, say, the discrete version of equation (7) is

$$\sigma = \sqrt{\frac{2\pi\Delta t}{\Omega}}\|f(t_n)\|_2$$

The sampling interval Δt of $f(t_n)$ is related to the sampling interval $\Delta\omega$ of the Discrete Fourier Transform of $f(_n)$ by the equation

$$\Delta t \Delta\omega = \frac{2\pi}{N}$$

and since the bandwidth of the discrete spectrum of f_n is $N\Delta\omega$ is is clear that $\Delta t = 2\pi/\Omega$ from which the result is thus derived.

Corollary

The scaling constant $2\pi/\Omega$ can be used to define a re-scaled the Volatility given by $\sigma := \sigma\Omega/2\pi$ thereby yielding the expression

$$\sigma = \left[\sum_{n=1}^{N-1} \ln\left(\frac{-R(t\_n+1\tilde{})}{R(t\_n\tilde{})\tilde{}}\right)^2\right]^{\frac{1}{2}} \quad (8)$$

Comparing equation (8) with equation (6), we observe a similarity in both forms with regard to the commonality of the quotient $R(t_{n+1})/R(t_n)$ and the logarithmic operation but where $\lambda < 0$ or $\lambda \geq 0$ but where $\sigma \geq 0 \forall n$.

4.2 Volatility Scaling

The zero crossings associated with computing of the α-index (as discussed in Section 2) and/or the Lyapunov exponent (Section 3) on a moving window basis provides the positions in time where there is a transition in the trend type. The value of the Volatility indicates the 'stability' of the time series, the temporal characteristics of all indicators being dependent of the size of the window or 'period' used. This suggests scaling the indices with the Volatility, i.e. computing the quotient $\alpha_\sigma = \alpha/\sigma$ and $\lambda_\sigma = \lambda/\sigma$, in order to assess not

only changes in the direction of a trend but the corresponding stability of that trend. This idea has obvious applications to a range of time series but especially in regard to financial time series analysis where forecasting both the type and characteristics of a trend is of fundamental importance, a positive trend with low volatility indicating a good investment horizon, for example.

5. FILTERED ZERO-CROSSINGS ANALYSIS

On the basis of the ideas considered in the previous section, the critical points at which a trend forecasting decision is made are the zero crossing points associated with λ_σ (or α_σ). We consider the case associated with parameter λ_σ but note that the arguments and analysis presented in this section applies in equal measure to the parameter α_σ, a comparison of the two parameters being given later on in Section 5.2.

By computing $\lambda_\sigma(t)$ on a moving window basis where t is the position in time of the window, identification of the zero crossings denoted by the function $z_c(t)$ involves the follow basic procedure:

$$z_c(t) = \begin{cases} +1, & \lambda_\sigma(t) < 0 \;\;\&\;\; \lambda_\sigma(t+\varepsilon) \geq 0; \\ -1, & \lambda_\sigma(t) > 0 \;\;\&\;\; \lambda_\sigma(t+\varepsilon) \leq 0; \\ 0, & \textit{otherwise}. \end{cases} \tag{9}$$

where ε is a small perturbation in time. This procedure generates a series of Kronecker delta functions whose polarity determines the position(s) in time at which a trend is expected to be positive or negative. Thus the function $z_c(t)$ identifies the zero crossings associate with the end of an upward trend and the start of a downward trend as is the case when $z_c(t) = -1$ and the end of downward trend and the start of an upward trend as the case $z_c(t) = +1$. This is a 'critical indicator' in regard to forecasting the trending behaviour of a time series, and, with regard to the back-testing algorithm to be discussed in the following section, is the primary performance evaluator.

In practice, the points at which the zero crossings are evaluated according equation (9) depend on the accuracy of the algorithm used to compute λ_σ which in turn, depends on the intrinsic noise associated with the time series used. This can yield errors in the positions at which the zero-crossings are computed especially with regard to changes associated with very short time micro-trends. In the context of longer term macro-trends, such micro-trends may legitimately be interpreted as noise although, in the context of financial times series analysis, for example, the term 'noise' must be understood to include legitimate price values. To overcome this effect, and, since the same argument applies to the computation of the Volatility, λ_σ is filtered thus:

$$\Lambda_\sigma(\tau) = \frac{1}{T} \int_{-T/2}^{T/2} w(t+\tau)\lambda_\sigma(t)dt \tag{10}$$

where

$$w(t) = \begin{cases} 1, & |\, t \,| \leq T/2 \\ 0, & |\, t \,| > T/2 \end{cases}$$

However, in the context of an evolving time series such as financial time series, a filtering strategy must be used given that

$$\lambda_\sigma(t) = \begin{cases} \exists \forall t \in (-\infty, \xi]; \\ 0 \forall t > \xi. \end{cases} \tag{11}$$

where ξ defines the finite extent of the times series at any point in time beyond which the series is undefined and can thereby be set to zero as given in equation (11) . For this reason, we consider a modification to equation (10) and write

$$\Lambda_\sigma(\tau) = \frac{1}{T} \int_{-T/2}^{T/2} w(t+\tau)\lambda_\sigma(t+\delta T)dt, \quad s \in [0,1] \tag{12}$$

The parameter δ defines a 'shift' which provides control over the extent to which the evolving time series is 'zero padded'. This parameter provides a critical difference in the accuracy of the results obtained through back-testing in terms of the reliability and accuracy of the function $z_c(t)$ to indicate the nature and duration of a trend. The most successful results occur when the 'shift' is increasingly closer to 1. However, in the context of an analysis of a financial time series (as discussed in the following section) this introduces a need for a trader to 'hold their position' until the position of a zero crossing has 'stabilised' (subject to the filter applied), thereby retaining a fixed position in the interim. This requirement introduces a 'Trading Delay Factor' given by $T - \delta T$. Thus an optimum value for the parameter S is the smallest value that yields a maximum back-testing accuracy.

5.1 Back-testing Results

Back-testing algorithms are designed to 'gauge' the accuracy of the results in terms of trend predictions and are usually but not exclusively related to testing a strategy for forecasting the behaviour of financial time series. In this context, the procedure operates on the basis that the price difference should be positive if the interval between the start and end points of a predicted trend are correct. In those cases where this occurs throughout the duration of the time series considered, the predicted entry and exits points are taken to be correct, else, they are taken to be incorrect. The accuracy associated with the back-test is then computed as a percentage in terms of the predictions being correct and only correct.

The algorithm for the back-tests used is compounded in the following procedure as predicated on the computation of $z_c(t_i)$ - equation (9) - using the filtered index $\Lambda_\sigma(t_i)$, *a priori* where s_+^+, s_+^-, s_-^- and s_-^+ denote a correct positive (up-ward) trend prediction, an incorrect positive trend prediction, a correct negative (down-ward) trend prediction and an incorrect negative trend prediction, respectively, in regard to the discrete time series data $R(t_i)$ used.

$R_+^+ = 0$; $R_+^- = 0$; $R_-^- = 0$; $R_-^+ = 0$;

$k = 0$; $\forall i$,
$if z_c(t_i) < 0, then \hat{\Lambda}_\sigma(t_k) = \Lambda_\sigma(t_i) \& \hat{z}_c(t_k) = z_c(t_i)$; $k = k+1$;
$if z_c(t_i) > 0, then \hat{\Lambda}_\sigma(t_k) = \Lambda_\sigma(t_i) \& \hat{z}_c(t_k) = z_c(t_i)$; $k = k+1$;

$\forall j$,
$if \hat{z}_c(t_j) - \hat{z}_c(t_{j+1}) < 0 \& \hat{\Lambda}_\sigma(t_j) - \hat{\Lambda}_\sigma(t_{j+1}) > 0, then R_-^+ = R_-^+ + 1$;

$$if \hat{z}_c(t_j) - \hat{z}_c(t_{j+1}) > 0 \& \hat{\Lambda}_\sigma(t_j) - \hat{\Lambda}_\sigma(t_{j+1}) < 0, then R_-^- = R_-^- + 1;$$
$$else$$
$$if \hat{z}_c(t_j) - \hat{z}_c(t_{j+1}) > 0 \& \hat{\Lambda}_\sigma(t_j) - \hat{\Lambda}_\sigma(t_{j+1}) < 0, then R_+^+ = R_+^+ + 1;$$
$$if \hat{z}_c(t_j) - \hat{z}_c(t_{j+1}) > 0 \& \hat{\Lambda}_\sigma(t_j) - \hat{\Lambda}_\sigma(t_{j+1}) > 0, then R_+^- = R_+^- + 1;$$

$$if R_-^+ + R_-^- > 0 then S = 100 R_-^+ / (R_-^+ + R_-^-) else S = 0;$$
$$if R_+^+ + R_+^- > 0 then L = 100 R_+^+ / (R_+^+ + R_+^-) else L = 0;$$

where L and S denote the percentage accuracy of going 'Long' (e.g. making an investment predicated on the forecast of an increasing price trend) and going 'Short' (e.g. selling an investment predicated on the forecast of a decreasing price trend). Note that this procedure (at least for $\delta > 0$) is applied to the filtered time series data and represents the accuracy associated a 'Delayed Call'.

5.2 Example Result using Energy Commodities Time Series Data

It is clear that the output associated with a normalised sample of the energy commodity *Brent Crude Weekly*, i.e. 1000 samples of Brent Crude Oil price sampled on a weekly basisexhibits a range of short and long time-scale trends. The plot also shows the filtered time series $\Lambda_\sigma(t_i)$. The periods used to compute $\lambda_\sigma(t_i)$ and $\Lambda_\sigma(t_i)$ are 15 and 20, respectively, with $\delta = 0.5$ yielding a Trading Delay Factor of $floor(T - \delta T) = 10$. For this case, $S = 71\%$ and $L = 83\%$ assuming a zero Trading Delay Factor which yields 'Instantaneous Call'. Using a 'Delayed Call', $S = 92\%$ and $L = 81\%$ thereby providing a higher combined percentage accuracy.

In comparison, we can show the effect of applying exactly the same computational procedure to the parameter α_λ using the least squares estimate given in Section 2.4. In this case, the 'Instantaneous Call' yields $S = 66\%$ and $L = 80\%$ but the 'Delayed Call' yields $S = 92\%$ and $L = 98\%$. In general, application of the (filtered) α_σ-index yields a better performance over application of the (filtered) λ_σ-index but this is at the expense of the higher computational overheads required to implement regression analysis. On the other hand, regression analysis generates a smoother time signature which explains the improved performance when compared with the time signature for the Lyapunov Exponent. It is also clear that, in comparison to the (filtered) λ_σ-index the (filtered)α_σ-index has a greater dynamic range. This is of value in regard to assessing the stability of a trend when the amplitude of α_σ increases. From the point-of-view of assessing the potential for increasing an existing investment, for example, this amplitude provides a confidence measure whose quantification lies beyond the scope of this work.

6. CONCLUSIONS AND DISCUSSION

The purpose of this paper has been to introduce a time series model based on Continuous Time Random Walk Models as derived from Einstein evolution equation - equation (1). In Section 2, we have shown that the first order temporal IRF associated with the Green's function solution to the Generalised KFE for a Memory Function of the type $1/t^\beta$ is given by $1/t^{1-\beta}$, $t > 0$ $\beta > 0$ which is independent of the PDF. This result is compatible with a Lévy Distributed PDF for a $\delta(t)$ Memory Function under the asymptotic condition $x \to 0$ (as discussed in Section 2.5). Within the unifying context of equation (1) we have also considered the Lyapunov Exponent and identified the respective numerical ranges associated with differentiating between the persistent and anti-persistent behaviour of a time

series (upward and downward trends, respectively) as provided in Section 3 and quantified in Table 1.

The application of the filtering operation discussed in Section 5 is crucial to the success of the 'predictive power' of the indices considered. Without application of this filter the accuracy of determining the correct zero crossings for going Long or Short remains relatively poor as predicated on the back-testing procedure considered in Section 5.1. Moreover, to yield the required accuracy necessary for most trading purposes, the filter must be applied with an appropriate shift. For the case of $\delta > 0$, the filter generates a result that is analogous to a forward-error-correction scheme. Without filtering, back-testing shows that the Long/Short accuracy is $\sim 50\%$. With the implementation of the filter (for $\delta = 0.5 - 0.7$), the accuracy is typically $\sim 80\%$++. The price that is paid for this accuracy is the delay required before application of a Long/Short Call. While this delay does not reduce accuracy in the prediction of a trend, its implementation yields a lower price difference between the entry and exit points. This dictates that smaller windows are used in the computation of an index subject to the back-testing accuracy obtained for a given time series.

In addition to the applications associated with financial time series analysis, which has been a focus of the results presented in this paper, the indices considered can be used for forecasting the behaviour of any evolving time series. Scaling by the Volatility yields a measure of stability, and, in this sense, the indices can be use to partition a time series in to stable and non-stable regions. In the former case, short-time forecasting techniques may be used with increasing confidence such as those based on the application evolutionary computing methods, for example, to predict the future value(s) of a time series as opposed to the predicting the trend alone. Finally, it is noted that the 'predictive power' of the α-index is slightly superior to that of the λ-index in terms of the results presented in this paper and those that have been studied by the authors to date. However, the computational overheads required to compute the λ-index are less and thus, for the development of applications on mobile trading devices, for example, the λ-index may be better suited.

Acknowledgements

The author would like to acknowledge the support of the Science Foundation Ireland and the National Digital Research Centre of the Republic of Ireland.

REFERENCES

[1] A. Einstein, On the Motion of Small Particles Suspended in Liquids at Rest Required by the Molecular-Kinetic Theory of Heat, Annalen der Physik, Vol. 17, 549-560, 1905.

[2] A. N. Kolmogorov, On Analytic Methods in Probability Theory, Selected Works of A. N. Kolmogorov, Volume II: Probability Theory and Mathematical Statistics (Ed. A. N. Shiryaev), Kluwer, Dordrecht, 61-108, 1992 (From the Original: Uber die Analytischen Methoden in der Wahrscheinlichkeitsrechnung, (1931). *Math. Ann.* 104: 415-458).

[3] W. Feller, On Boundaries and Lateral Conditions for the Kolmogorov Differential Equations, The Annals of Mathematics, Second Series, Vol. 65, No. 3, 527-570, 1957.

[4] F. W. Olver and L. C. Maximon, Mittag-Leffler function, Handbook of Mathematical Functions in Olver, (Eds. W. J. Frank et al.), NIST, Cambridge University Press, 2010.

[5] H. G. Schuster, Deterministic Chaos: An Introduction, VCH, Weinheim, 1988.

[6] V. I. Oseledec, A Multiplicative Ergodic Theorem: Lyapunov Charcteristic Numbers for Dynamical Systems, Trans. Mosc. Math. Soc., Vol. 19, 197-231, 1968.

[7] Y. G. Sinai, Introduction to Ergodic Theory, Princeton University Press, 1976.

[8] G. Boffetta, M. Cencini, M. Falcioni, and A. Vulpiani, Predictability: A Way to Characterize Complexity, 2001. http://www.unifr.ch/econophysics/.

Applied Science Journal

Volume 1(1) (2016)

AsJ

SHORT COMMUNICATION

Game-Oriented Learning in Virtual Education Space

V. Valkanova, A. Petrov and V. Valkanov

Plovdiv University, Bulgaria

In the article an architectural model is being revealed which represents a structure of software agents in their natural environment - a virtual world which the authors call the Game World. The components making up this model are being discussed in detail and the whole process of education and collection of player statistics is being revealed. A prototype game providing a way of learning road rules is presented. It is an example of a Belief-Desire-Intention (BDI) model used for educational purposes.

Categories and Subject Descriptors: []:

General Terms: [], []

Additional Key Words and Phrases: Game-Oriented Learning, Virtual Education Space, Software Agents, BDI-Architecture, Game World

1. INTRODUCTION

The current progress of software technologies opens many opportunities in the field of education. Artificial environments offer great educational potential. Artificial intelligence can not only help teachers and students craft courses that are customized to their needs, but it can also provide feedback to both about the success of the course as a whole. Some schools, especially those with online forms of education, are using AI systems to monitor a student's progress and to alert professors when there might be an issue with student performance. Combining artificial environment with artificial intelligence promises great results for the purposes of education. The Virtual Educational Space (VES) [**?**] presents possibilities in the context for supply with electronic educational services. One of the purposes in VES is insurance of educational process under the pretext of educational games. The game activity for educational purposes is based over these principles: activity, dynamism and entertainment, performing a certain role, problem drawbacks, molding activities, competition, independence and effectiveness [**?**]. The preparation of the students supposes not only varied and lasting knowledge but formation of knowledge and habits through which

Correspondence address: Plovdiv University, Plovdiv, Bulgaria

the knowledge is gaining efficient character. Learning through the activity plays a big role for building an applied practical and research abilities, encourages the creative work and ensures the possibility for independent work and team work. The game is enjoyable activity which has several rules and it often has a reward for the well-presented participants [**?**]. Ordinarily there are two models of the games in education:

—Instructivist model - the participants learn while they play educational games, presented and developed by others.

—Constructivist model - the participants learn while they play educational games, presented and developed by themselves. There are different classifications of the games depending on their purpose or the way the game flows. Here we will present briefly some of the basic types of games:

Creative Games.

—Subjective role games - plot and roles are the main components of the game. The realization is through game action with the children playing roles. During the game the children always face questions which need to be solved through discussion, contemplation, communication. The carefully selected role play and role for each participant allow to express each one's individual opinion and offer a creative solution to the real (play) problem.;

—Building constructive games - The building-constructive game is closely tied to the display of constructive creativity. The children's creative displays in this game are developed in the constructive activity with building elements which is realized as a complex system of mental and practical actions.

—Game dramatization - In this game the child is in the role of a literary character, can improvise and really undergoes a creative process and feels pleasure from the impersonation.

1.1 Rule-Based Games.

Those games are built from an earlier-set content and by already known and established rules without which you cannot participate in them. A typical example for this kind of games are the Didactic games (mathematical and musical), where a strictly defined didactic material is used which is selected with regard to known educational and learning goals. Movement Games. In the traditional movement games the children develop physical abilities such as quickness, agility, strength, skill, learn to be durable, persistent and hard-nosed.

1.2 Commercial and Serious Games.

The "serious games" [**?**] are these in which education (in its different forms) is the main purpose and the amusement comes afterwards. Didactic games are those serious games which teach the players in a particular area and they give them new knowledge through the playing approach [**?**]. The game has its own place in the educational process and it carries out important pedagogical functions:

—For simplifying the complex matter;

—For application of the theoretical knowledge;

—For testing of the gained knowledge and skills.

—For discovering new knowledge

—For personalizing education

The raised interest by the side of coming generation to the big number of computer and video games directed our attention to developing virtual games which allow us to apply their new learning or find new one. Every game in VES gives the student a virtual micro world.

2. VES OVERVIEW

Three types of assistants are supported in VES [**?**]. The personal assistants have to perform two main functions providing the needed "entry points" of the space (Fig.1, first orbit). First, they operate as an interface between their owners and the space and if necessary, carry out activities related to personalization and adaptation. Secondly, they interact with other assistants in the space in order to start and control the execution of the generated plans. In certain cases they operate as an intermediary for activation of scenarios or services. The personal assistants will be usually deployed over users' mobile devices. The specialized assistants are usually located on the server nodes of the VES, known as operatives (Fig.1, third orbit). They support the execution of the plans generated by the personal assistants; therefore they implement suitable interfaces to the available electronic services and data repositories. Operatives serve two subspaces, known as DiLibs-Subspace and Admin-Subspace respectively. Guards (Fig.**??**, second orbit) are special assistants which are responsible for safety and the efficient execution of the plans in the space. These are usually intelligent devices that react to various physical quantities in the environment, e.g. smoke, temperature, humidity. The guards act as an interface between the physical and the virtual world in the space.

3. GAME WORLD

Game World is part of the DiLibs-Subspace of VES, whose goal is to make educational games an indelible part of the education process. Furthermore, the games will be used for evaluating the students' abilities for applying new knowledge. We also expect that the games will increase the creativity during the learning process. Three assistants operate in the Game World (Fig.**??**) that we will describe briefly. Game Assistant (shortly Gamer) is an operative responsible for planning, management and control of conducting games. This assistant is implemented as an intelligent agent with bounded rationality [**?**] and is transparent to the players. Moreover the Gamer is collecting different information about the progress of game, the chosen approach (strategy), etc. This information is personalized for each player, because the agent keeps track of which actions belong to which player. The collected data is transmitted to the teacher's personal assistant (PAT). The teacher then can make conclusions about the errors, gaps in knowledge and plan an appropriate corrective action. In special situations, the Gamer can communicate with an operative (Creativity interface Agent, short CA) which provides interface to the environment known as Creativity Assistant [**?**]. The idea for the development of this environment is for the particular creator to make a structure known as Creativity Map. This structure presents different aspects of an artistic and creative process leading to the creation of a particular

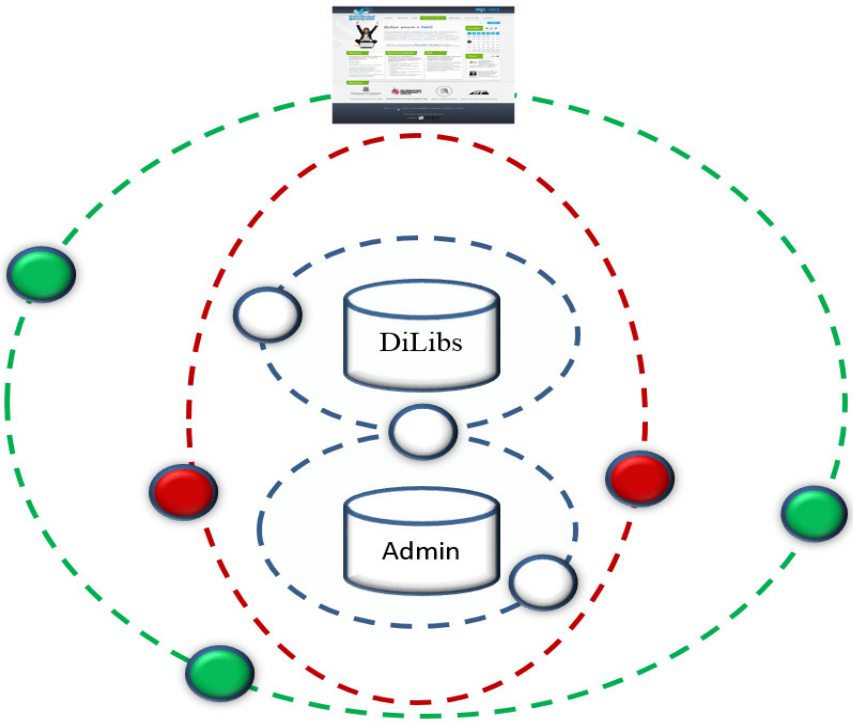


Fig. 1: VES Architecture

piece. By researching different cards, conclusions can be drawn from the creativity about common characteristics of creative thinking. Usage of the Creativity Assistant has been researched in different application areas, e.g. art and software engineering. Here we would like to explore its applicability for the goals of blended learning. In the suggested model the assistants' environment plays a central role. The environment is composed of two parts:

—Game World - this is an interactive visual environment in which the progress of the games is displayed. Only this part of the assistants' environment is visible to the players. Game World is a sharable structure which is only accessible by the players and by the Gamer. On the one hand, depending on the nature of the game, the players can enter the necessary information. On the other hand, the Gamer's activities which are designed for the player can be visualized here. A typical characteristic of the Game World is that the Gamer itself is transparent to the players but at the same time it has many faces to them. The Gamer appears in each game, making it interactive, reactive and proactive;

—Meta World - this part is completely invisible to players. It records various control data, constraints and rules that can be used to assess player achievements so that the environment can be customized and the game can adapt depending on the gathered information. For example, it can set restrictive conditions to personalize a game. Only the Gamer, the PAT and the CA have access to the Meta World. The three agents share this common environment through which they can interact as well.

Conducting a game in the Game World happens through separate steps, as follows (Fig. **??**.):

—Choosing a game - in this step the Gamer reviews the contents of the game room and offers a game to the student. The game is chosen after approval from the PAT (respectively the teacher);

—Initialization from the environment - in this step the teacher (through the PAT) can set various constraints related to the student's individual abilities, a pedagogical approach, the nature of the game et al, and this information is saved in the Meta World. The Gamer sends a message to the CA for readiness to activate the Creativity Environment (any control information about this environment can be saved in the Meta World);

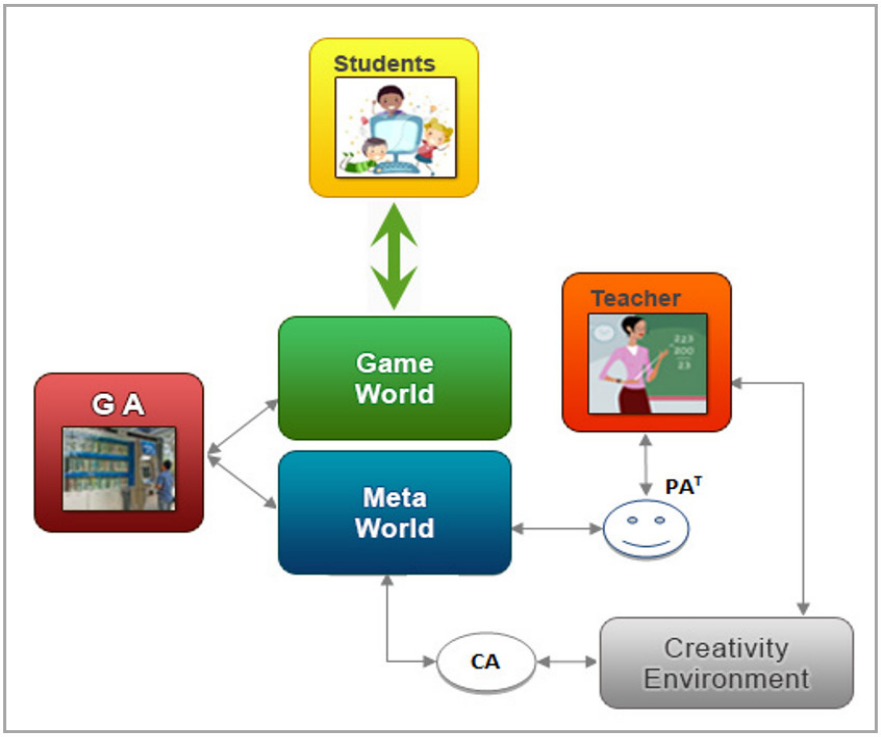


Fig. 2: Game World

—Game Execution - the Gamer creates a game session to conduct the selected game and afterwards activates the Game World where it is visualized. Usually the games are executed interactively, with the Gamer interacting periodically with the playing student's PAS. During this step the Gamer collects data saved in Meta World that will be used to evaluate the achievement of the player. The Gamer activates itself (proactive) when

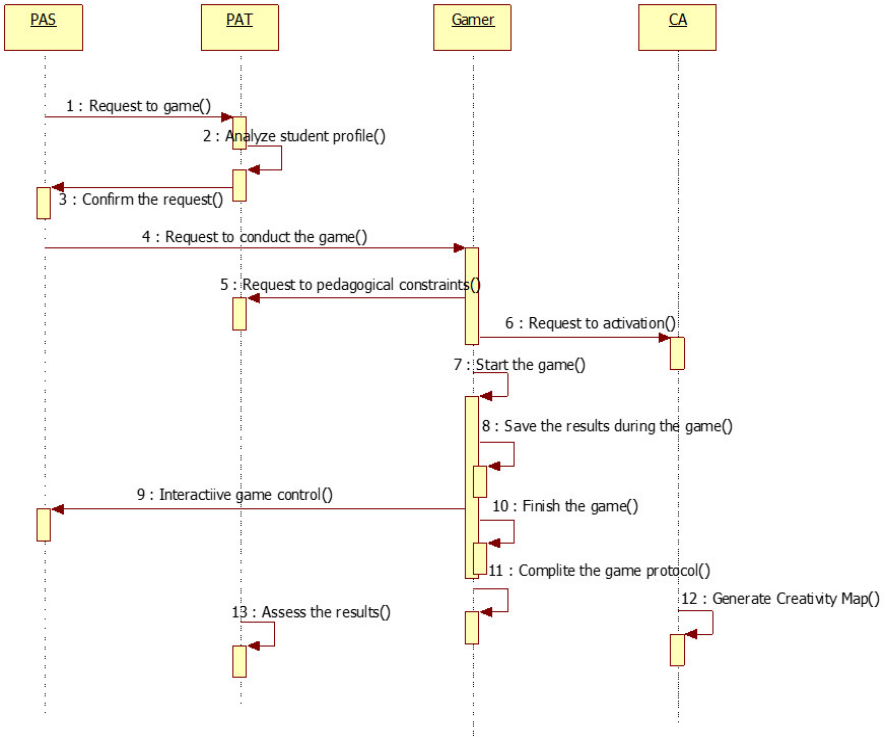


Fig. 3: Assistants' Interaction

circumstances arise such as the need for assistance (if player is eligible), violation of the game rules, completion of the game, or continuous inability to progress;

—Evaluation of the results (achievements) of students - in this step, the achievement state in Meta World is evaluated. Furthermore, the saved data can be used by CA to create a Creativity Map of the player or to update the existing one.

4. IMPLEMENTATION OF GAME WORLD

One of the first challenges in the development of the Game World is building a united technological environment which integrates supplies for computer graphics and animation with an environment in which intelligent agents can operate. In order to implement the Game World the following technologies are chosen: HTML5 [**?**], JavaScript - WebGL [**?**]. The libraries used for prototyping version of the virtual gaming world are Babylon.js [**?**] and Three.js [**?**], which allow relatively easy 3D objects management and handling of characters, lighting and camera. Future plans include the use of Unity3D as it advertises a lot of useful features in its newest version in which there will be a direct export to WebGL and built-in integration with Oculus Rift [**?**]. The assistants are implementing with the help of JADE [**?**]. The prototype realization of the Game World will be demonstrated with a particular game. Prevention of risk situations among children is an important task for Bulgarian schools. The classes delegated to such learning in the curriculum are extremely

insufficient. For that reason we expect that the inclusion of educational games in this topic as an addition to the curriculum will show convincingly the benefits of blended learning.

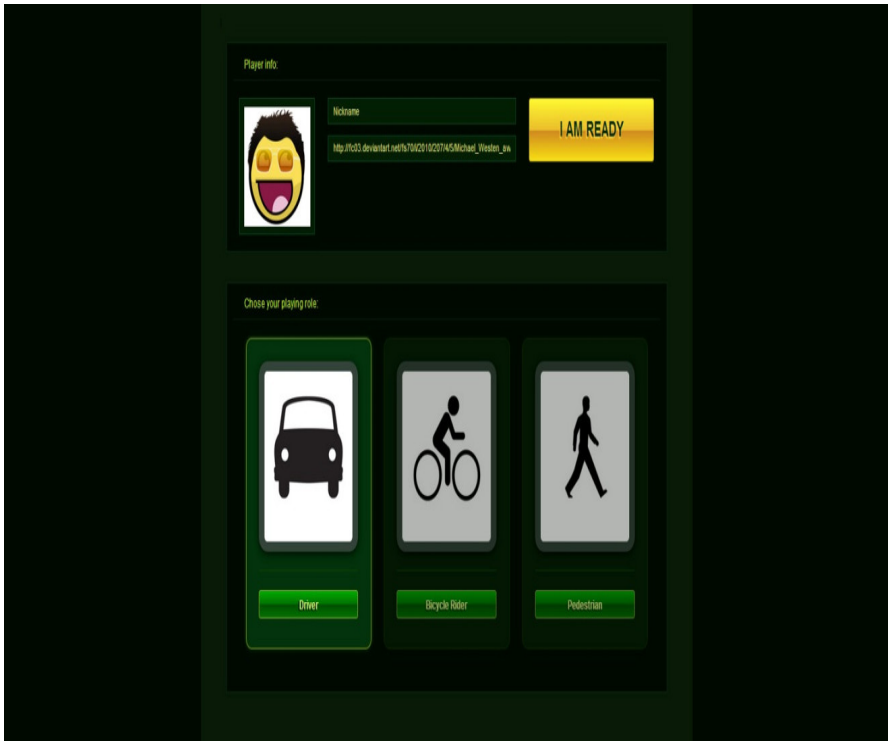


Fig. 4: Home Screen of the Game

At the starting point the player has to enter their personal data - nickname and avatar and chose a role: a driver, a cyclist, or a pedestrian (Fig. **??**.). Next the player has to choose a map and a route (Fig. **??**.). The routes can be from real population spots (e.g. Plovdiv) or from hypothetical ones.

The task of the player is to pass a given route successfully. On certain special points of the journey known as control points he or she has to take the right decisions according to the road rules. Control points on a route are places where the player must make a decision, for example related to a road sign or a specific situation (Fig. **??**.).

In this case, the Game World can be generated systematically or randomly from a pool of various elements such as the following:

—Different types of routes available in the city;

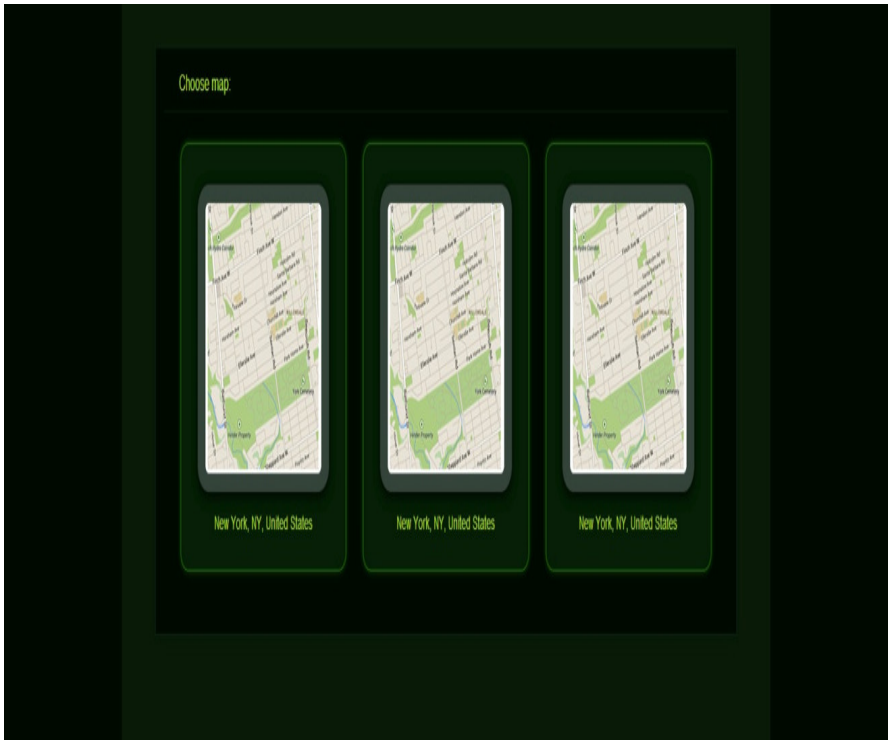


Fig. 5: Route Selection

—Depending on the specific user role (Driver, cyclist or pedestrian) different events may happen;

—Different road signs may appear;

—Different traffic situations may occur. Upon reaching a control point, during the deliberation the Gamer has to determine the current objective of the game. In this place, the mental states of the Gamer are specified as next:

—Beliefs are reached (checked) control points on the selected route so far, in other words the state of player's progress;

—Desires are certain elements of the current control point (e.g. red traffic light at a crossing) or possible situations at the current control point (e.g. the player has to cross a street);

—Intention is a selected element or situation of the current control point (element of desires). During the means-ends step, according to the current goal the agent should generate a plan of action. Different options are possible:

—Proposal of different options to move through a specific point from which the player must choose one or more;

—Prompt for the player to propose an option or to choose an action. The prototype will be improved in order to get more customized game play according to the specific player and his previous results.

Fig. 6: Control Point

5. CONCLUSION

The presented prototype is a good example of a way in which a game-oriented blended learning could be supported. Game-based learning provides the following benefits:

—To assess the ability to apply knowledge gained in game situations - a game may be one of the levels of learning in Bloom's taxonomy [**?**], used in Bulgarian education;

—The game can be integrated in the whole learning process;

—To assess creative thinking and students' actions, but also to evaluate how individual games contribute to the establishment of such thinking and action.

REFERENCES

[1] V. Valkanova, Researching the Virtual Learning Space in the Secondary School, PhD Thesis, Sofia, 2014, ISBN 978-954-322-768-6.

[2] I. Ivaniv, Intelligent Education Methods, Varna, 2005.

[3] Zyda, M. , "From visual simulation to virtual reality to games", IEEE Computer, vol. 38, no. 9, pp. 25-32, September 2005.

[4] D. Michael, S. Chen, Serious games: games that educate, train and inform. Boston, USA: Thomson Course Technology, 2006.

[5] E. Paunova, K. Stoilova, Information Technologies Supporting Knowledge Creation, Electronic Journal, Free University of Varna, Vol. 6, pp. 1-27, 2013, ISSN 1313-7514.

[6] S. Stoyanov et. al., Virtual Education Space, (accepted for the same isuue).

[7] H. Zedan, A. Cau, K. Buss, S. Westendorf, Mapping Himan Creativity. Leicester, UK: De Montford University, 2008.

[8] HTML5, http://www.w3.org/TR/html5/.

[9] T. Parisi, WebGL: Up and Running. Building 3D Graphics for the Web, O'Reilli Media, 2012.

[10] http://www.babylonjs.com/.

[11] http://threejs.org/.

[12] http://www.oculusvr.com/.

[13] Bellifemine F., Caire G., Greenwood D., Develping multi-agent systems with JADE, John Wiley & Sons, Ltd, February 2007.

[14] Bloom B. S., Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. New York: David McKay Co Inc., 1956.

Applied Science Journal

Volume 1(1) (2016)

AsJ

SHORT COMMUNICATION

A Proposed Technique for Medical Diagnosis Using Data Mining

Alaa H. AL-Hamami(Arab University for Graduate Studies, Jordan)
Mohammad A. AL-Hamami (Applied Science University, Bahrain)
Soukaena H. Hashem (University of Technology, Iraq)

Abstract: This research introduces a proposal to improve medical Diagnosis using Association rules of Data Mining Technique. For medical diagnosis related with cancers, we aim to find new relationships for new predications that by build a multidimensional database which their basic dimensions are blood, serum and tissues. Each dimension has its specific attributes. Here we will build six of the proposed multidimensional database, the first one for healthy, second for suspected, third for early infected, fourth for stage one infection, fifth for stage two infections, sixth for stage three infections.
Then after finding all associations rules from all of these six multidimensional databases will be extracted by using the proposed method of mining. Then genetic algorithm will be applied on all the resulted association rules from the six databases to find new association rules. We are hoping that the resulted rules will give new predictions for cancers disease.
keywords: Data mining, Medical Diagnosis, Multidimensional Databases, Association Rules, and Genetic Algorithms.

1 Introduction

Data mining derives its name from the similarities between searching or valuable information in a large database and mining rocks for a vein of valuable ore. The more general terms such as Knowledge Discovery in Databases (KDD) describe a more complete process. Data mining is being put into use and studied for databases, including relational databases, object-relational databases and object oriented databases, data warehouses, transactional databases, unstructured and semi structured repositories such as the World Wide Web, advanced databases such as spatial databases, multimedia databases, time-series databases and textual databases, and even flat files [1]. This process of knowledge discovery is finding a connection between data and facts. It can, in fact, be said to be a relationship between prior facts and subsequent facts. The prior fact is called the antecedent and the subsequent fact is known as the consequent. There should be no assumption of an underlying cause and effect connection between the antecedent and consequent. Through the process of data mining, it should always be kept in mind that data is not facts, in order to have knowledge and facts one needs data.

The problem is stated as follows, see table 1, Let I = {i1, i2, ...im} be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that T $\subseteq$ I. A unique identifier TID is given to each transaction. A transaction T is said to contain X, a set of items in I, if X $\subseteq$ T. An association rule is an implication of the form "X $\Rightarrow$ Y", where X $\subseteq$ I, Y $\subseteq$ I, and X $\cap$ Y =$\varnothing$. The rule X $\Rightarrow$ Y has a support s in the transaction set D is s% of the transactions in D contain X $\cup$ Y. In other words, the support of the rule is the probability that X and Y hold together among all the possible presented cases. It is said that the rule X $\Rightarrow$ Y holds in the transaction set D with confidence c if c% of transactions in D that contain X also contain Y . In other words, the confidence of the rule is the conditional probability that the consequent Y is true under the condition of the antecedent X. The problem of discovering all association rules from a set of transactions D consists of generating the rules that have a support and confidence greater that given thresholds. These rules are called strong rules [4, 5].

3. THE PROPOSED TECHNIQUE

Medical data mining applications are determines disease outcome and effectiveness of treatments, by analyzing patient disease history to find some relationship between diseases. The proposed technique will introduce suggestion to improve medical diagnosis using association rules of data mining technique. The proposed technique is presented in the following steps:

Step one:

Build a multidimensional database has three dimensions these are blood, serum and tissues. Blood has four attributes RBC, no. of RBC, WBC, no. of WBC. Serum has two attributes antibody and interleukin. Tissues have four attributes no. of cells, size of cells, nuclease shape, and rate of division. See figure (1) which show the proposed Multidimensional Data Base (MDB).

This technique aims to build six of the proposed MDB the first one for healthy, second for suspected, third for early infected, fourth for stage one infection, fifth for stage two infections, sixth for stage three infections.

Dims	Blood				Serum		Tissues			
Tid	RBC	No. RBC	WBC	No. WBC	intelukien	Antibody	No. cells	size cells	neuc shape	Rate div

Figure 1: The proposed multidimensional database for medical diagnosis.

Step two:

We get the information of these six databases from medical diagnosis experts, for each database we take 100 patients. We see from our point of view is better to take those patients from one sex type only (female or male) since the values of some attributes are differ from one sex to another. Also when we get the data of the attributes these data were numbers and logical states.

For encoding such data to be suitable for the proposed mining technique is to consider the following:

- The attributes have number values; we will make a threshold for normal range of numbers, if the values consistence with it then the attribute will not appear. But if the value exceeds the threshold this attribute will be presented by some character such as A.
- The attributes have logical states values; also we will do the same by assign a threshold for normal state (which may be assigned yes or no). If the values consistence with it then the attribute will not appear. But if the value exceeds the threshold this attribute will be presented by some character such as B.

The proposed encoding with our attributes for male;

Blood attributes:

1. RBC: the normal range of diagonal = 8 micron, if it was in that range then A will not appear, but if the range is more, then A will appear.
2. No. of RBC: the normal range of these cells 5,2 million in one cubic millimeter, if it was in that range then B will not appear, but if the range is more, then B will appear.
3. WBC: the normal range of diagonal = 6-15 micron, if it was in that range then C will not appear, but if the range is more, then C will appear.
4. No. of WBC: the normal range of these cells 8 thousands in one cubic millimeter, if it was in that range then D will not appear, but if the range is more, then D will appear.

Note: the same consequence for Serum attributes and Tissues attributes.

Step three:

Using the Proposed technique to deal with multidimensional database to extract the association rules, by divide the database to its three dimensions the first one for blood attributes, the second one for serum attributes and the third one for tissues attributes. This technique is presented by the following points:

A- Extract the frequent itemsets from the multidimensional database by the following points:

1. Take the first dimension D1 and find the frequent items in it. For example the threshold of minimum support is equal to 50% TIDs, so we find many frequent itemsets, such as: (D1_itemset1, *, *) and (D1_itemset2, *, *) because the values D1_itemset1 and D1_itemset2 occur 50% times; value * for the other two dimensions shows that they are not relevant. Repeating the operation for the second dimension we will find many frequent itemsets in them, for example (*, D2-itemset1, *) only because the value D2-itemset1occurs 80% times; value * for the other dimensions shows that they are not relevant. Repeat the operation for the third dimension we will find many frequent itemsets in them, as an example (*, *, D3-itemset1) only because the value D3-itemset1occurs 70% times; value * for the other dimensions shows that they are not relevant.
2. Now take the first dimension with concern of the second and third dimensions we will get the following frequents itemsets, for example: (D1_itemset1, D2-itemset1, *) and (D1_itemset1, *, D3-itemset1). Now take the second dimension with concern of the third dimension we will get the following frequents itemsets for example: (*, D2_itemset1, D3-itemset1). Since there are no more dimensions in our case, the search will end. We propose to take the minimum support for the itemsets in different dimensions.
3. If there are more than three dimensions then we will apply the same thing by taking each dimension alone, and then take each dimension with others.

B- The general frequent itemsets for the multidimensional database will be as in the follow:

{
D1_itemsets1, D1_itemsets2, D1_itemsets3,….
D2_itemsets1, D2_itemsets2, D2_itemsets3,……
D3_itemsets1, D3_itemsets1, D3_itemsets1,……
D1_itemsets1D2_itemsets1 , D1_itemsets1D2_itemsets2,…
D1_itemsets1D3_itemsets1 , D1_itemsets1D3_itemsets2,……
D2_itemsets1D3_itemsets1 , D2_itemsets1D3_itemsets2,…….
}

C. Ggenerate the association rules according apriori algorithms using the frequent itemsets displayed in the previous step.

Step four :

After extracting the association rules for all six multidimensional databases then we aim to mix these association rules by using Genetic Algorithm (GA) [5] to obtain new rules for new predictions, this could be done by:

1. A genetic representation or encoding schema for potential solutions to the problem. Each association rule will be presented as a series of numbers (all alphabets representing the attributes which are coded by numbers such that A=1, B=2,.. and finally the --→ = 0). For example the association rule ABD--→FG has the following numbers representation (124067).
2. One way to create an initial population of potential solutions, the initial population already created with association rule algorithm which established on the multidimensional database. So this means the initial population of GA will be all the association rules which are represented by series of numbers.
3. An evaluation function that plays the role of the problem environment (novel association rules), rating solutions in terms of their "fitness". Here the proposed evaluation function for each rule is consist of three parts, these are:
    - First part: counter of each number in each series of numbers is equal to one.
    - Second part: each number will appear only on the left or the right of zero.
    - Third part: confidence of the rule will pass the minimum confidence.
4. Genetic operators that alter the composition of offspring. One-point crossover is the most basic crossover operator, where a crossover point on the genetic code is selected at zero number occurs in the series of numbers, and two parent rules are interchanged at this point.
5. Crossover exploits existing rule potentials, but if the population does not contain all the encoded information needed to find the novel rules, no amount of rules mixing can produce a satisfactory solution. For this reason, a mutation operator capable of spontaneously generating new frame is included. The most common way of implementing mutation is to flip a bit with a probability equal to a very low, given Mutation Rate (MR). A mutation operator can prevent any single bit from converging to a value through the entire population and, more important, it can prevent the population from converging and stagnating at any local optima.
6. Values for the various parameters that used by the genetic algorithm (population size, rate of applied operators, etc.). For our particular problem we use the following parameters of the genetic algorithm: Population size, pop-size = 4000 (the parameter was already used), Probability of crossover, PC = 1, Probability of mutation, PM = 0.001 (the parameter will be used in a mutation operation) [4].

Continue with genetic processing until the optimized rules will be optimized to be the novel rules.

4. THE PROPOSED SOFTWARE ENGINEERING APPROACH

To explain the proposed technique practically, we introduce the proposed software, see figure (2) which explain the main window for the software. This window contains six commands and these are: display database, update database, mining medical database, applying GA on AR and finally analyze and display prediction.

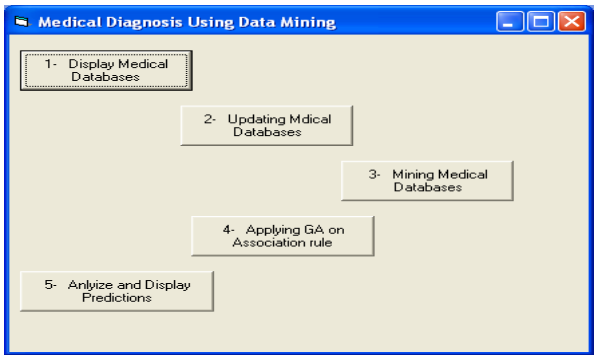
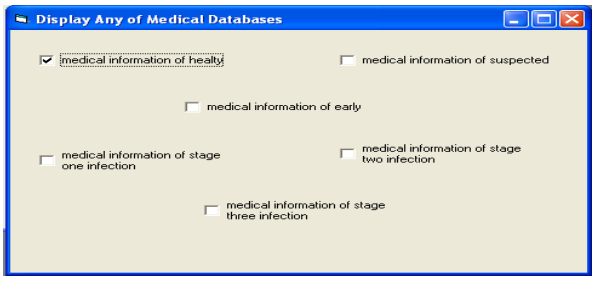
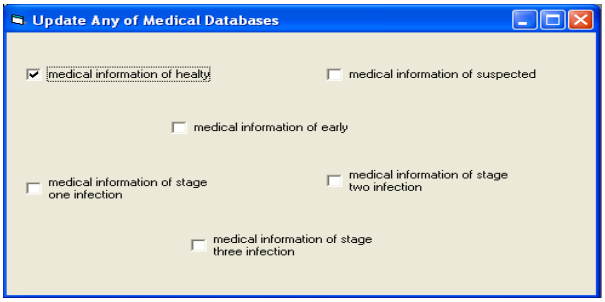


Figure 2: The main window of the software.

The first and second command: If the first command (display medical database) clicked then figure (3-a) will be appear to make user choose which database wanted for display. If the second command (update medical database) clicked then figure (3-b) will be appear to make user choose which database wanted for update. Figure (4) shows the first database for healthy persons.



(a)



(b)

Figure 3: The windows for display and update databases.

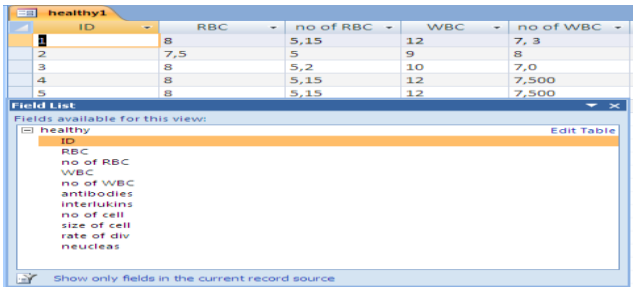


Figure 4: The database Window for healthy person.

The third command: If the third command (mining medical database) clicked the window in figure (5) will appear.

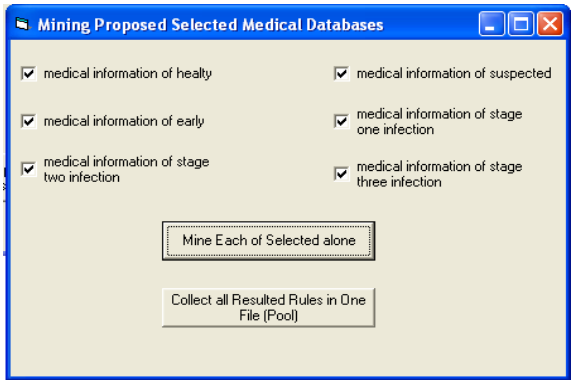


Figure 5 The window of mining process.

This window provides a medical diagnosis analyzer for the option to choose which of the six databases, the user likes to mine and then collect their resulted AR from one file. Here we will choose the entire six, then click the command (mine each of selected alone). This is to apply the proposed AR mining on each database to extract their resulted ARs in separated six files, see figure (6). After clicking the command (collect all), all these files will be in one file called pool.txt, see figure (7).

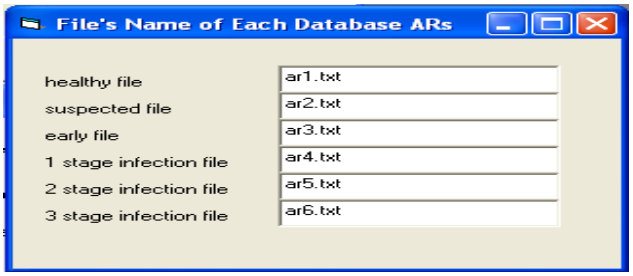


Figure 6: The window which display files name of ARs for each database.

Figure (7): The window which display the file name collect all ARs in all six files.

The fourth command: This command called (apply GA), if it clicked then figure (8) will appear.

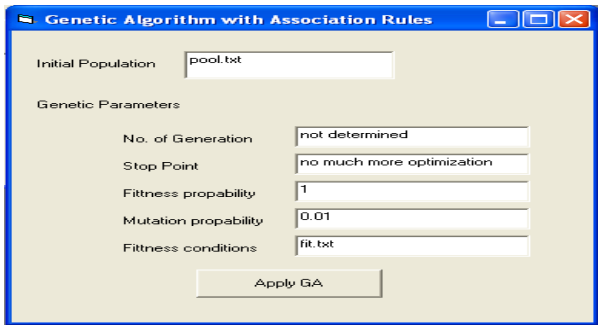


Figure 8: The Window of GA parameters.

This window displays the file which has all ARs of all the six databases. Also explains that there is no finite number of generation but the stop point indicates that there is no much more optimization, fitness must be perfect no error tolerance, the probability of mutation must be very little, and fitness function which has three conditions will be found in file called fit.txt.

Finally if the command called (apply GA) in figure (8) will be clicked then all ARs in pool.txt file will be converted in to the proposed encoded schema. The GA on these encoded ARs will be applied according to the presented parameters. After this process the resulted ARs will be stored in file name's gaar.txt. The last one will be decoded the new encoded ARs to the normal ARs format, store these final ARs in file name's agar.txt, see figure (9)

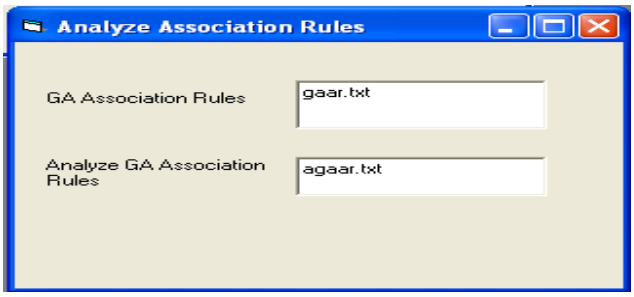


Figure 9: The Window which displays the names of files resulted from applying GA on ARs.

The fifth command: If that command is clicked then all the analysis process will be done to give a report contains an analysis for ARs and determines the predictions extracted from them.

5. RESULTS AND CONCLUSIONS

From the proposed research we conclude the following:

1. Here we built modest software to implement the proposed technique. That software customized for the following specification: built six multidimensional databases as explained in early sections, each one has three dimensions; first one the blood which has four attributes, second has three attributes and finally third has five attributes.
2. We tend to make each one of these six proposed multidimensional database has 100 transaction (each one has the medical information for 100 person. Those persons are the 100 healthy people in first multidimensional database, suspected in second multidimensional database, and so on.
3. Applying our proposal for extracting frequent itemsets from these databases will save time and space. Because of extracting the frequent itemsets with cancelling the dimension consideration (deals with the proposed database as normal transactional database) will make the no. of attribute is 12 and that will consume much time and space.

4. After extracting the association rules from each database we tend to mix these association rules which are extracted from six separated database by using genetic algorithm.
5. We customize the genetic algorithm for our proposed technique, that by propose a schema for encoding the rules, and then making the initial pool is all the extracted rules from the six databases. Making the crossover point is -→ provide much justifying generating new child rules from parents rule.
6. Fitness function customizes to satisfy the basic conditions in building the association rules according the data mining techniques.
7. Although our limited experience in the medical diagnosis field, we get a new association rules which give us new relations. We think these relations could give a new prediction to discover cancers in very early times. From these relations we introduce the following:
    - After analysis stage there are new rules which predicate very strong relations among changes of blood components with sold cancer, in addition to the changes of tissues components. These rules traditionally must appear with infected persons but in our proposed technique we see it appear with healthy, suspected, and early infected persons in addition to the three infected stages.

6. REFERENCES

[1] M. S. Chen, J. Han, and P. S. Yu. "Data mining: An overview from a database perspective". IEEE Trans. Knowledge and Data Engineering, 8:866-883, 1996.

[2] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. "Advances in Knowledge Discovery and Data Mining". AAAI/MIT Press, 1996.

[3] J. Han and M. Kamber. "Data Mining: Concepts and Techniques". Morgan Kaufmann, 2000.

[4] Mitra S., and Ahharya T., "Data Mining Multimedia, Soft Computing, and Bioinformatics", John Wiley and Sons, Inc., 2003.

[5] Mohammadian M., "Intelligent Agent for DM and Information Retrieval", Idea Group Publisher, 2004.

[6] Ala'a H. AL-Hamami, Mohammad Ala'a Al-Hamami and Soukaena Hassan Hasheem, " Applying data mining techniques in intrusion detection system on web and analysis of web usage", Asian Network for Scientific Information, Vol. 5, No. 1, pp: 57-63, Pakistan, 2006.

[7] Ala'a H. AL-Hamami, and Soukaena Hassan Hasheem," Privacy Preserving for Data Mining Applications", WORLD '07', The 2007 World Congress in Computer Science, Computer Engineering & Applied Computing, Las Vegas, Nevada, USA, June 25-28, 2007.

[8] Mohammad A. Al- Hamami and Soukaena Hassan Hashem, " Applying Data Mining Techniques to Discover Methods that Used for Hiding Messages Inside Images ", The IEEE First International Conference on Digital Information Management (ICDIM2006), Bangalore, India, 2006.

Applied Science Journal

Volume 1(1) (2016)

AsJ

Short Communications

Road Accidents Prediction Using Hybrid Reasoning Technique

Saif Al-Sultan and Mussab Aswad
Applied Science University

The decreasing number of road accidents around the world have attracted researchers from different disciplines to investigate the causes of accidents and to propose accidents prediction and prevention systems in an attempt to reduce the fatalities and save people's lives. Predicting road accidents is a complicated task due to the fact that, the accidents is considered as dynamic interaction between the driver, the vehicle, other vehicles on the road and the environment, meaning that it is affected by different factors and develops over time. Therefore, it is important to collect and analyse information about the above mentioned factors in order to predict the accident likelihood accurately. In this paper, we explain the contributory factors that lead to road accidents and we propose a hybrid reasoning technique for predicting the accidents likelihood accurately by combining information about the driver, the vehicle and the environment.

Categories and Subject Descriptors: Context-Aware Systems [**Vehicle**]:

General Terms: [Software Engineering], [Pervasive Computing]

Additional Key Words and Phrases: Accidents Prediction, Hybrid Reasoning, Driver Behaviour

1. INTRODUCTION

At the present time cars and other private vehicles are used daily by many peoples. The biggest problem regarding the increased use of private transport is the increasing number of fatalities that occur due to accidents on the roads; the expense and related dangers have been recognised as a serious problem being confronted by modern society. According to the UK department for transport report for road casualties in Great Britain for the year ending June 2014, there were 193,290 casualties of all severities including 24,580 killed or seriously injured due to road accidents. This number is considered as big dilemma that have to be tackled by governments and societies.

Road accidents is considered as uncertain context and a dynamic interaction between the driver, the vehicle, other vehicles on the road and the environment, meaning that it is affected by different factors and develops over time. According to the UK department

Correspondence address: Applied Science University, Al-Eker, Kingdom of Bahrain

for transport [1], the main factors that contribute in road accidents are classified into the following categorise: road conditions (i.e. slippery road), vehicle defects (i.e. defective brakes), injudicious action (i.e. following too close), driver error (i.e. failing to look properly), driver abnormal behaviour (i.e. drunk, fatigue, reckless, etc.), vision affected by fog, pedestrians and special conditions such as stolen vehicle. Including more factors during predicting (inferring) the accident likelihood leads to increase the accuracy and the completeness of the system. However, it is important to appropriately select the that have to be included in the reasoning in order to infer more accurate accident likelihood. In this paper, we present the main work that have been carried out in the area of accidents prediction and prevention, and we identify their limitations and challenges. Then we propose and discuss the use of hybrid reasoning techniques for inferring different context (contributory factors) in order to predict the accidents likelihood accurately.

The reminder of this paper is organised as follows. Section 2 introduces the work that have been done in the field of accidents analysis and prediction. The proposed idea of predicting the accidents using hybrid reasoning techniques is illustrated in section 3, and the conclution is given in section 4.

2. RELATED WORK

There have been several researchers working on the development of accidents prediction and driver behaviour detection systems using range of methods. Some of them have attempted to measure the driver context, the vehicle context or the environment context in order to predict the accidents likelihood. While, other researchers have tried to monitor the driver, vehicle and the environment, to detect the behaviour of the driver which is considered as a major factor in accidents. Below, we summarise the main work that have been carried out in these areas.

Tokoro et al. [2] utilising adaptive cruise control (ACC) to proposed a pre-crash safety system (PSC) aims to enhance vehicle safety and reduce the injuries caused by collisions, by activating a pre-crash seat belt and pre-crash brake to assist in the event of an unavoidable accident. The author has developed millimetre wave radar as well as a new signal-processing algorithm in order to increase the accuracy of the system. When the electronic control unit (ECU) detects an incoming obstacle based on the time to crash (TTC) decideabout the collision is avoidable or not, the PCS will activate a seat belt and brake assessment. The proposed system islead to inaccurate crash predictionsbecause it obtains information about the vehicle only. Moreover, the system deploys the action only when the crash is unavoidable, which is insufficient, as the aim of this work is to predict a crash early enough to avoid injuries.

In. [3] deployed a collision mitigation system (CMS) to calculate the possibility of a collision, using variety of high technology sensorsand a pre-crash algorithm, the system takes into account the wheel speed, steering angle, yaw-rate, lateral acceleration, side-slip angle, longitudinal and lateral velocity and acceleration, as an input to the system. The pre-crash algorithms used to enhance the airbag deployment performance by calculating the crash possibility, time to crash (TTC) and crash type via estimated information. An original method for crash type decision making were proposed to enhance airbag deployment, which involved defining a possible crash zone and lowering the crash algorithm

threshold using a bicycle model that had been modified.

Karlsson et al. [4] developed a collision avoidance system using late braking to avoid or mitigate any collision. A Bayesian approach with implementation of an extended Kalman filter (EKF) method and a particle filter approach were used to solve the tracking problem and to make decisions. Comparisons were made between the two filters for different sensor noise distributions, using the Monte Carlo simulation method. The braking decision was based on a statistical hypothesis test, in which the collision risk was measured in terms of required acceleration to avoid collision. This model considered the state vector relative to position, velocity, direction and distance, to calculate the hypothesis node probability.

In [5], the focus of the paper was on building a context-aware smart car by developing a hierarchical model that is able to collect, reason about, and react to contextual information about the driver, the vehicle, and the environment, providing a safe and comfortable driving environment. However, this system is restricted to warning the driver and controlling the vehicle and does not warn other vehicles on road by sending warning messages. [6], a context-aware system is proposed that is used to collect and analyze contextual information about the driver, the vehicle, and the environment in real-time driving. It also collects information from questionnaires completed by the drivers to create driving situations. The Bayesian network is used to reason about this contextual information, which is relatively uncertain information, by using a learning process to observe and predict the future behavior of the driver. The system was able to predict the future behavior of the driver and cannot detect the current state of the driver and warn other vehicles on the road.

In[7], the detection of the fatigue level of the driver using a video camera to extract different cues such as eye state, eyelid movement, gaze movement, head movement, and facial expression is attempted to measure the fatigue level and warn the driver via in-vehicle alarms. In [8], a system for drowsy driver detection in real-time driving by collecting information about the driver's behavior, such as the speed of the vehicle, the vehicle's lateral position, the yawing angle, the steering wheel angle, and the vehicle's lane position is proposed. Their system uses artificial neural networks to combine different indications of drowsiness and to predict whether a driver is drowsy and to issue a warning if required.

The above systems have achieved good results in predicting the accident likelihood and in detecting the behaviour of the driver. However, none of which has taken into account the combination of the driver, the vehicle and the environment in one system. In addition, all of which have only utilised one reasoning technique to infer the collected context, as they used one contributory factor which itself hard to be sensed using sensor and needs to be inferred using specific reasoning technique. We argue in this paper that, to predict the accident likelihood accurately, several contributory factors have to be inferred and several reasoning technique have to be applied to perform the inference process.

3. THE PROPOSED ACCIDENTS PREDICTION MODEL

In our previous work [9], we proposed a context aware accidents prediction and prevention system for VANET, the aim of the proposed system was to predict the accidents likelihood and severity. The accident likelihood was considered as uncertain context that evolves over the course of driving. The Dynamic Bayesian Network was utilised to combine different

contributory factors of road accidents including the vehicle, the driver, the environment and other vehicles on the road. However, the driver status was defined as having two mutually exclusive states, which are good and bad. While, the behaviour of the driver was defined in our previous work [10] as a complex and dynamic interaction between the driver, the vehicle and the environment, and is developing over time. Several contextual information was collected and inferred utilising the Dynamic Bayesian Network in order to deduce four types of behaviours (i.e. drunk, reckless, fatigue and normal behaviour). Both of the above systems have achieved good results, but defining the driver in [9] as having two states (good and bad) is insufficient considering the definition of behaviour that given in [10]. Moreover, detecting the behaviour of the driver in isolation will not provide a complete view about the likelihood of the accidents.

In this paper, we propose a hybrid reasoning technique that combines the previously proposed systems [9; **?**] together in one system in order to predict the accident likelihood more accurately. In [9], the focus was on the vehicle and the environment where few contextual information about the driver was considered. While, in [10] the focus was on the driver and the environment context, where few contextual information about the vehicle was considered (i.e. the position of vehicle between the lane markers). Combining the output of the first system, which is the accident likelihood with the output of the second system, which is the behaviour of the driver using another reasoning technique (i.e. Fuzzy Logic or Neural Networks) will provide more accurate prediction, as several contributory factors (i.e. road conditions, vehicle defects, driver error, driver abnormal behaviour and weather conditions) will be collected, analysed and inferred using different reasoning techniques. The level of accuracy in predicting the accident likelihood will be increased due to the fact that, the more data we include in the reasoning process, the more accurate the decision will be.

The main challenge that raised from the above argument is, how to integrate more than one reasoning technique in one system and how to integrate them with the system architecture. The available reasoning techniques are computationally expensive and are slow, while the time is a critical factor in the safety applications. Therefore, it is important to examine and analyse the performance and the complexity of the reasoners, and the way of integrating many reasoners in one system without affecting its resources and performance.

4. CONCLUSION

The increased number of fatalities and related dangers that have occurred as a consequence of road accidents, have been identified as a major problem being confronted by modern society. Road accidents might take place due to several reasons (factors), The UK department for transport has identified the main factors for road accidents. In this paper, we explained the main contributory factors that cause road accidents, then we summarised different proposed driver behaviour detection and accident prediction systems and identified their limitations and drawbacks. Then we proposed a hybrid reasoning technique for predicting the likelihood of road accidents, and discussed the challenges that might be faced when applying this method, the challenges are: integrating more than one reasoning technique in one system without affecting the system performance and provide a hybrid reasoner that is able to work in real time.

REFERENCES

[1] "Stats19." https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/230590/stats19.pdf. Accessed: 16/02/2015.

[2] S. Tokoro, K. Kuroda, A. Kawakubo, K. Fujita, and H. Fujinami, "Electronically scanned millimeter-wave radar for pre-crash safety and adaptive cruise control system," in Intelligent Vehicles Symposium, 2003. Proceedings. IEEE, pp. 304–309, IEEE, 2003.

[3] K. Cho, S. B. Choi, and H. Lee, "Design of an airbag deployment algorithm based on precrash information," Vehicular Technology, IEEE Transactions on, vol. 60, no. 4, pp. 1438–1452, 2011.

[4] R. Karlsson, J. Jansson, and F. Gustafsson, "Model-based statistical tracking and decision making for collision avoidance application," in American Control Conference, 2004. Proceedings of the 2004, vol. 4, pp. 3435–3440, IEEE, 2004.

[5] M. S. Devi and P. R. Bajaj, "Driver fatigue detection based on eye tracking," in Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on, pp. 649–652, IEEE, 2008.

[6] H. Singh, J. Bhatia, and J. Kaur, "Eye tracking based driver fatigue monitoring and warning system," in Power Electronics (IICPE), 2010 India International Conference on, pp. 1–6, IEEE, 2011.

[7] Z. Zhu and Q. Ji, "Real time and non-intrusive driver fatigue monitoring," in Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on, pp. 657–662, IEEE, 2004.

[8] H. Ueno, M. Kaneda, and M. Tsukino, "Development of drowsiness detection system," in Vehicle Navigation and Information Systems Conference, 1994. Proceedings., 1994, pp. 15–20, IEEE, 1994.

[9] M. Z. Aswad, "Context aware pre-crash system for vehicular ad hoc networks using dynamic bayesian model," 2014.

[10] S. Al-Sultan, A. H. Al-Bayatti, and H. Zedan, "Context-aware driver behavior detection system in intelligent transportation systems," Vehicular Technology, IEEE Transactions on, vol. 62, no. 9, pp. 4264–4275, 2013.