ORIGINAL RESEARCH

Theory of meaningful information, background, excerpts and meaning representation

Boris Sunik

Software Development, Zuken E3 GmbH, Ulm, Germany

Correspondence: Boris Sunik. Email: boris.sunik@GeneralInformationTheory.com.

Received: December 30, 2012 **Accepted:** April 18, 2013 **Online Published:** May 20, 2013
DOI: 10.5430/air.v2n3p102 **URL:** http://dx.doi.org/10.5430/air.v2n3p102

Abstract

The Theory of Meaningful Information is a general theory that can be applied to information of every type, level and complexity. The theory explains the nature and functionality of information and enables the production of relevant definitions regarding language and knowledge, which remain operative also in the case of non-human languages and knowledge systems. The non-executable language T, built in accordance with the theory, allows the creation of code bases functioning in the way of semantic nets which can be used for the representation of a different kind of knowledge.

Key words

General information theory, Theory of meaningful information, Knowledge representation

1 Introduction

We live in the age of applied sciences. Just consider the role of applied biology responsible for the unbelievable progress in medicine or applied physics and applied chemistry, forming the basis of modern industry and modern infrastructure in general. Our society is also hardly imaginable without the applications of social sciences like economics, sociology, social psychology, political science, criminology etc.

The only big exception to this rule is the area of information sciences concerned with meaningful information. While there are many theories and approaches dedicated to the subject, none of them was ever able to produce a really applicable theory of meaningful information.

Attempts of formalizing meaningful information can be found in semantics [1] and knowledge representation [2]. Lu [3] and Losee [4] tried to produce a feasible formalization and generalization of meaningful information based on Claude Shannon's theory of communication [5]. Mathematical approaches were common, e.g. the one based on probability theory proposed by Cooman et al. [6] and the theory of information in the form of "a mathematical model of information flow" by Keith Devlin [7]. A theory considering information to be a basic property of the universe much like matter and energy was developed by Tom Stonier [8]. Other approaches, however, negated mathematical interpretations and described the world as consisting of entities and interactions, where information is a reflected relationship of both and has no mathematical core [9]. Burgin [10] tried to build a general information theory by systemizing features of information.

The failure of information science to produce an applicable information theory had hardly any consequences for the relevant practical activities, the first and the most important of which is high level programming. Each high level programming language is able to express the meaningful content restricted to the depicting of bit sequences allocated in the computer memory and the operations with these bit sequences. Another way of working with meaningful information consists of using the respective software applications (e.g., an application recognizing the meaning of some text).

Taking into account that the software is also composed with the help of the high level programming languages, the means of the high level programming are in actual fact the only practically usable methods of formalizations of meaningful information.

The means of high level programming are basically inexpressible in the mathematized formalisms characteristic to the information science. This directly follows from the history of programming theory.

The idea of expressing a programming language with the help of the mathematized formalisms is nearly as old as programming languages themselves. The initial work was carried out by Noam Chomsky [11] who proposed a containment hierarchy of classes of formal grammars and formulated the idea of a universal grammar. The next step was taken by John Backus who developed the formal notation known as BNF (Backus Normal Form). BNF was first presented in the paper describing IAL, which was later renamed ALGOL. The method, which until now remains the only practical method used to formally represent the syntax of programming languages, was formulated by Backus as follows: "There must exist a precise description of those sequences of symbols which constitute legal IAL programs… For every legal program there must be a precise description of its 'meaning', the processor transformation which it describes, if any…" [12].

Despite the fact that BNF failed to achieve the second part of this objective, it inspired further intensive research, which produced miscellaneous methods for formally expressing a program's meaning. The methods developed during the last five decades are traditionally grouped into three general approaches known as denotation semantics, operational semantics and axiomatic semantics [13].

Unfortunately, none of these approaches was ever able to attain any status outside of pure theory because of the lack of useable results. It is very difficult to imagine that a programmer studying some programming language could extract any useful knowledge from formal specifications of this language produced with the help of the known methods of semantics representations. Taking into account the longevity and the intensity of research, its inclination to the formal mathematical models and the diversity of developed solutions, the only plausible explanation of the failure is the presence of certain fundamental constraints basically hindering an effective solution with the help of formal methods [32].

The Theory of Meaningful Information (TMI), presented here, is the attempt to develop an applicable general information theory by extending the universal methods of information formalization developed in programming outside of their original domain.

TMI is based on an overall definition of information, which can be applied to information of every kind, level and complexity. The definition of information enables a view of the world in terms of objects, actions, relations and properties. Information is considered as the feature manifesting itself in the relations between certain real world entities. TMI provides both the uniform view of all kinds of information and the universal language actually supporting the requested representations.

TMI is an axiomatic theory, defining the key notions information, knowledge, language, sign solely on the basis of their functional characteristics. In this it differs from the other approaches of information science restricting these terms to the entities of their interest (i.e., most information-related theories understand a language as a natural human language, the term knowledge as human knowledge etc.).

According to TMI, every real or imaginary entity can be viewed as an object whose behavior can be expressed in the manner of programming algorithms. For example, the algorithm of the Earth consists of the cyclical movement around the sun and the rotation around its axis each day. In the same way the structure of the Sun, Earth and their algorithms can be described in English they can also be described in the universal representation language T, created on the basis of the TMI approach.

T is built by extending the conceptual system of C++ to the representation of entities exceeding the C++ representation world. The definition of T is not associated with any particular implementation and consists of the definition of the language grammar, usage rules and core vocabulary in the way of a natural language like English or German. Notions of T like class, instance, procedure, variable, parameter and so on stand for nouns, verbs, adjectives, pronouns and other parts of speech.

Assuming there is sufficient vocabulary, each text composed in any of human languages can be adequately translated to T in the same way as it can be translated into another human language.

Neither TMI nor T employs common formal models and approaches like first order calculus, semantic nets, conceptual graphs, frames, which constitute the basics of formal languages traditionally used in the realm of knowledge representation like OWL [14], CycL [15], KM [16] and others.

2 Definition of information

According to the standard theory of cosmology, literally everything – matter, space and time – started with the Big Bang. In the beginning, all matter and energy which made up the universe was squeezed into an infinitely hot, dense, unstructured singularity. It then started to expand, became more structured, the fundamental forces were divided, matter formed to atom nuclei, leptons and molecules. The content separated into distinct components, parts of which were more stable than others and so got to preserve their form and size over long periods of time.

So was formed our universe, which we can consider as a super heap consisting of stable objects of different levels of matter organization and their heaps. The process of universe formation and evolution consists of permanent births, deaths, and changes of all entities of the heap. A stable object here is understood as a three dimensional item with mass, which is reasonably steady, has a location or position in space at any moment, and which can be changed by exerting force. The stable objects of the lower levels of the matter organization are inanimate physical bodies like stars, planets, stones, molecules, atoms and subatomic elements like hadrons, leptons etc. and those of the upper levels are living organisms of various complexity levels starting from unicellular bacteria up to human beings.

Under the influence of force, a stable object changes its speed, movement direction, starts/stops moving if it is exposed to the physical (energetic) influence of another object or process (like one object hitting another object and changing its movement etc.) and is distorted. A body in rest can come into motion if the balance of forces maintaining its immobility is disturbed, e.g. a bridge is stable because its weight is balanced by the counter-pressure of its pillars and thus fails if one of the pillars is damaged.

A general characteristic of force-induced changes is that the effectual change occurs exclusively because of the energy produced or withdrawn by the causal action (as in the case of the falling bridge).

Stable objects are not really static but rather dynamic combinations of elements. Electrons revolve around atomic nuclei, quarks exchange in hadrons; molecules perform chemical reactions in the body of cosmic objects and cellular organisms. Stated more precisely, stable objects are dynamic systems that are better at adapting to the environmental conditions around them and therefore can retain their form and stability. This ability to adapt means they are able to resist external and

internal influences by either ignoring them or changing in order to counteract the influence. Since the ability to resist is limited, a stable object will collapse if the force influencing it exceeds a certain threshold, and it will be changed.

The life of a stable object can last from a fraction of a second to billions of years and during all this time it is constantly moving and/or changing. The sequence of changes and movements of a stable object is defined in this work as the object's algorithm that can be represented with the help of various expression means like pictures, schemes and texts composed in various languages.

The subject to be considered here is the causal relations between the changes of stable objects and their environments. The phenomenon of causality, famously characterized as the cement of the universe by David Hume [17], includes many miscellaneous causal relationships, but we will limit ourselves to changes occurring in objects under various influences in a Newtonian world.

According to the Random House Unabridged Dictionary [18], the term causality denotes "a necessary relationship between one event (called cause) and another event (called effect) which is the direct consequence (result) of the first."

The term change is defined in the same dictionary as "to make the form, nature, content, future course, etc., of (something) different from what it is or from what it would be if left alone".

The axiom of this work is that all causal changes in the real world are based on only four distinct schemes: two primitive causal relations occurring under the influence of physical forces and two complex ones. Changed entities are either complex stable objects including several components or systems thereof.

2.1 Change I driven by an internal force (internal change)

Changes of this kind are independent from the external environment of a changed object or at least considered to be independent on the level of the analysis of the object. A stable object alters its form, position or structure under the force produced by its internal processes. Such a change is the most frequent case of alterations among living organisms. Growth of living creatures, human mental processes or cellular division can be cited as examples.

Non-living physical entities are also changed in this way, especially those consisting of a dynamically balanced set of physical processes such as cosmic stars. Stable atoms disintegrate due to spontaneous radioactivity, which can take a very long time. This simplest causal mechanism occurring within a single stable object is designated here as internal change.

Internal change will be represented with the help of a horizontal rectangle that is split along the x-axis into two parts. The upper half refers to the changed entity and the bottom half describes the change to this entity. Because this object changes spontaneously without any apparent external force or influence, there is only one rectangle and no other objects or arrow directions contained in this graphic.

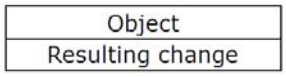

Figure 1. Spontaneous change

2.2 Change II driven by the external force (forced change)

The second basic kind of causal relation consists of changes that occur as a result of externally working forces. According to Britannica force is "any action that tends to maintain or alter the motion of a body or to distort it" [19].

The mechanism of this change is simple: the influence of a force alters the affected object or its movement: A cosmic body changes its orbit due to the influence of gravity produced by some external entity. A bullet accelerates because of the powder explosion in the cartridge; glass breaks when it falls to the floor etc. This form of causal mechanism is called forced change here.

Forced changes represent the simplest form of a causal relation between two objects, the first of which — a causal action of force — delivers energy causing the effectual change of a stable object. Forced change is represented as follows:

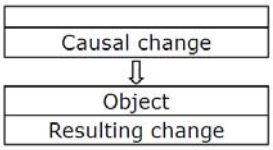

Figure 2. Externally driven change

The empty part of the upper rectangle illustrates the fact that a causal change can have an arbitrary origin. The arrow designates the direction of force. The vertical order of rectangles represents their relationship in time.

2.3 Change III activated by external force (activated change)

This and the following causal mechanism are based on the sequence of primitive changes described above. The entities being subject to change are complex stable objects with several components and internal energy sources or systems consisting of several components and energy source(s). Furthermore, the changed objects are either organic organisms or inorganic systems of artificial origin. Inorganic objects of natural origin are too primitive for such complex behavior.

The activated change consists of changing a stable object (system) by external non-forceful influence. A changed object (system) has to possess (or have access to) an energy source and an externally controlled lock (switch), which opens and closes the energy flow produced/delivered by the energy source.

A typical example of this kind is an electric light activated by an electrical button or lever when someone (something) switches it on and the electrical bulb lights up. The sequence of actions includes the following four activities:

(1) external force presses the button;
(2) the button closes the electrical circuit thereby unlocking the energy source;
(3) electricity flows to the bulb;
(4) the bulb lights up.

The general graphical schema of this change is as follows:

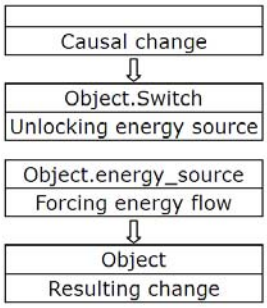

Figure 3. Externally activated change

The difference between change II and change III is the addition of the two externally activated steps two and three. The lack of an arrow between them shows the missing energetic influence between the respective changes. This change is thus a sort of combination between change I and change II with the internal change being triggered by an external force.

The thing activating an internal change by external influence is referred to in this work as an activating switch or an activator.

Examples among biological objects are unicellular organisms registering the change of the environment by their sensors and changing in response the process of their metabolism. Components built on the principle of activated changes are integral parts of any technical system like electronic circuits.

A more elaborate example would be a traffic light at a pedestrian crossing (crosswalk). A person approaches a crosswalk, pushes a button, the traffic light changes to red and the crosswalk sign flashes green. The only work a person has to do is push a button and the rest is done by the internal component(s) and energy source(s).

The specifics of this type of causal relation are the energetic independence between the activating action and the resulting change which allows various activation-result pairs with the same internal mechanism. The energetic independence radically increases the survival chances and adaptability of systems implementing it.

Let us imagine there is a sea lagoon fed by an intermittent river. Some microorganisms, whose nutrition is delivered by the river's water, have been able to adapt to the changeable conditions by slowing its metabolism after the level of oxygen in the water drops below a critical point. The internal mechanism registering the level of oxygen is the typical activator changing the behavior pattern of these microorganisms in result of the alternation of the external environment.

If some species of this microorganism are carried by currents into the open sea, they encounter another changeable food source consisting of unsteady sea currents of very salty water. Surviving in these conditions requires another activator, which would regulate the microorganism's metabolism by reacting to the change in water salinity, while activator-controlled basic structures can be left unchanged.

From the viewpoint of evolution, it is a much more efficient way of development than the development of a completely new microorganism, which would be necessary if the activating switches were not involved in the evolutional process.

2.4 Change IV communicated by external force (communicated change)

The mechanism of activated change provides an essential part of flexibility relative to primitive causal mechanisms I and II, on the other side, it also has its restrictions convincingly demonstrated by the traffic light example. In the form delineated above, these traffic lights are hardly usable on a busy street, because many pedestrians who want to cross the street will actually block the traffic by persistently pushing the crosswalk button.

In order to solve this problem our traffic light needs to be able to define the switching moment on its own, i.e. by changing the cause-effect direction. Assume that a person doesn't push the button, but the button will be periodically (e.g. once every 3 minutes) pushed against the finger of a person presumably pressing the button. If the finger blocks the button, the crosswalk lights switches to green for the pedestrians and red for cars. Otherwise the color of the traffic light remains the same.

Certainly this is not a very convenient schema for traffic lights and it doesn't give pedestrians utter satisfaction either. A more efficient solution to this problem consists of augmenting the switching schema by adding one more element – a passive mediator object (in this case a button together with associated components), which can be pressed by a pedestrian and queried periodically by a traffic light. By pushing such a button, the pedestrian only switches the button to another

state which can be ascertained by the traffic light during the querying process. If the button is set, the traffic light unsets it and then switches the crosswalk light to green for the pedestrians.

This example demonstrates the work of information, which is a state of some passive object set by some actor and queried by some reactor in order to produce some resulting action. An actor has to set at least one state and a reactor has to distinguish between at least two states (the absolute minimum is a dichotomous variable – the state set by an actor and the lack thereof).

The following definitions will be used throughout this work:

- The mediating object is a variable. Wherefrom follows a yet even shorter definition of information being conceived as a value of a variable used in an algorithm.
- Communication is the process of interaction between an actor setting a variable's value and a reactor identifying it and performing these or other actions as the result.
- An actor is designated as an Information Setting Entity (ISE) and a reactor as an Information Driven Entity (IDE).
- The resulting change constitutes the semantics of information.

Relative to the activated change, this kind of causal relation has been enhanced by the addition of three new steps inserted after the causal change.

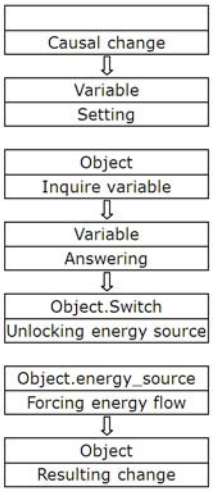

Figure 4. Communicated Change

In this work, a variable is not considered a mathematical construction, but an object in the discernible world. Since the features of communicators and communications can differ, the structure of the variable can differ, too. Thus, in implementation where the crosswalk light directly differentiates between pressed and non-pressed buttons, the variable consists of the complete button mechanism. In alternative implementations where the button mechanism is connected to the input port of the traffic light's internal processor the mediated variable is the input computer port and the button mechanism is an auxiliary appliance used by a pedestrian for setting this variable.

It is not required that a variable can be switched unlimitedly between its states. A variable is considered to be constant if it cannot change during its whole life span like e.g. a printed letter, which is essentially a picture painted on some surface. Other special cases are variables that can be changed irreversibly, i.e. only once, e.g., an undamaged pencil can be used as one sign and a broken pencil as another sign.

This understanding of information complies with the one actually applied in programming. The variables in programming are restricted to bits or bit sequences and the only two fundamental operations with variables are set and query, whereas all others are based on these two operations.

Another restriction in programming is that the actor and the reactor are usually the same, so the communication occurs not between different objects but between different states of the same computer.

2.5 Summary of causal relations

All four types of changes are given in Figure 5. Each rectangle contains the stable object (if present) and its change. The order from top to bottom represents the timely flow. The arrows represent a force being exerted.

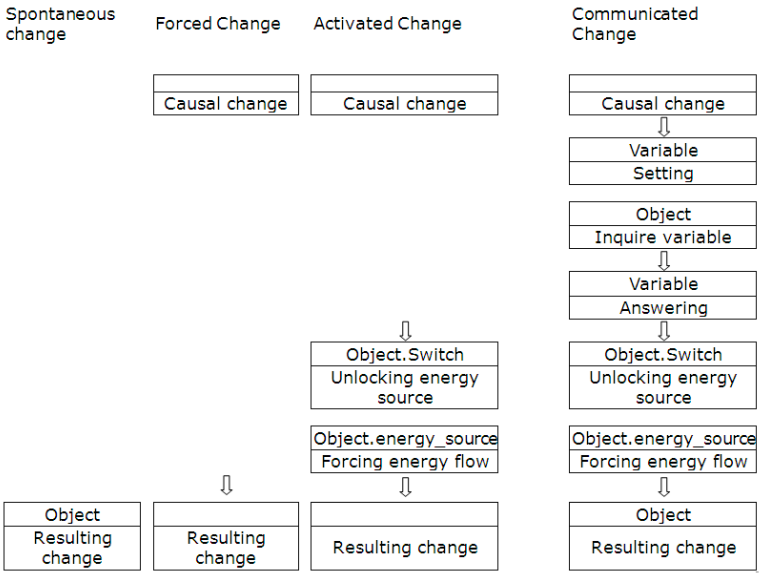

Figure 5. Summary of causal relations

2.6 Information atoms

(1) In general, there are three groups of processes associated with communication: information production (setting activity), transmission of information from one place to another and use (resulting activity) as depicted in the following Figure 6.

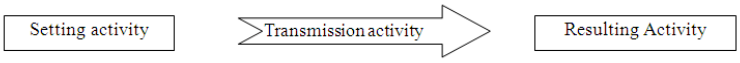

Figure 6. Information related activities

The transmission process (which was the only subject of Claude Shannon's seminal theory of information [5]) can be reduced to nothing, as in the examples of the traffic light and microorganisms. In other cases, it can be very sophisticated and may include numerous steps, each of them producing its own information representations. The stages of the transmission process are designated in this work as information processing.

(2) Activators and variables represent two levels of information carriers. The activators are low level entities which unite both set and query in the only action while the variables represent the high level information processing in

which these two functions are separated from each other. Essential differences between these two forms of information application are:

While an activating switch immediately causes the resulting change, the setting of a variable is detached in time from the querying result.

A single variable can cause many changes in miscellaneous IDE algorithms versus the only change caused by an activator; Many variables can be queried with the help of a single inquiring mechanism, which is essentially a switch.

The difference between an activator and a variable is functional, not physical. A computer bit, which is the most primitive variable on the computer level, is an activator on the level of its physical implementation. Another good example would be a neuron in the brain acting as a variable, whereas a motor neuron connecting the brain with a muscle is an activator.

(3) The term varier is used in this work as the most abstract designation of an information carrier, which can be either an activator switch or a variable. Variers of naturally developed ICEs from the level of cnidarians (corals, jellyfish) are neurons. In this work a neuron is considered an activating switch changing between no-output and output states by reason of the relevant external signal for this neuron. Depending on the type of input signal, neurons are classified as changeable and non-changeable. An example of a non-changeable neuron is a motor neuron which transmits impulses from a central area of the nervous system to an effector, such as a muscle. An example of a changeable neuron is a brain neuron, whose activating signal can be reprogrammed to respond to a number of signals [20].

(4) A stable object can only be considered an IDE if it contains at least one varier. The communication between an environment and an IDE occurs with the help of sensors, which are essentially activators. The architecture of an IDE containing only variables but no activators is physically possible but doesn't comply with IDE principles because such an entity is completely independent from the environment.

(5) Of both IDE and ISE, only the first inherently possesses the information related functionality whereas an ISE can be an arbitrary phenomenon of the organic or inorganic world with or without information related abilities. An IDE with the mediated varier as its interface constitutes the kernel part of the communication environment, which in general can get information from miscellaneous ISEs. An object possessing IDE capabilities with or without ISE capabilities is designated as an Information Capable Entity (ICE).

(6) As long as the same cause produces the same effect, the nature and structure of (a) varier(s) is irrelevant. If a certain lagoon is periodically filled with water without sufficient nutrients, the organisms living there can change their metabolism by reacting to miscellaneous environmental parameters such as water salinity, water temperature or water density with the help of the distinct inquiring mechanisms. The same is true in the case of artificial devices. The traffic light will function in the same way with miscellaneously designed switching buttons.

3 Information Capable Entities (ICE)

3.1 Classes of ICE

All currently known ICEs were developed on Earth either as a result of the evolutionary process or human engineering. However, their features and structures are very different. Naturally developed unicellular and multicellular organic

organisms were followed by social organisms and artificial devices created by the most complex natural ICEs — human beings. Known ICEs can be separated into the following classes and subclasses based on certain fundamental characteristics:

(1) Organisms

- Unicellular microorganisms — the simplest form of naturally occurring ICEs
- Multi-cellular organisms from non-structured sponges up to human beings
- Social organisms ranging from primitive organizations like ant colonies up to various created human societies and states.

(2) Artificial objects and systems

- Non-computer artificial objects and systems. These are entities, which use the simplest forms of information and are normally unable to produce information. Among them are all devices powered by electrical currents equipped by at least one controlling switch. Such devices also possess various sensors accepting these or other information.
- Computers and non-computer systems containing one or more embedded processors (everything from dish-washers up to cars, airplanes etc.)
- Components of all aforementioned entities, possessing ICE functionality.
- Software entities.

3.2 Stages of ICE evolution

The objects controlled by the information influence differ by their inner organizations. The simplest ICE are single-level objects with one or several sensors. These are systems like the aforementioned traffic light or such unicellular organisms, which can choose the moment of reaction by themselves.

The next level of complexity are programmed objects, which in addition to externally accessible information carriers, also possess internal variables whose values are set during the process of creating the ICEs (hard-coding) or during their lifetime (soft-coding).

Evolutionally hard-coded systems preceded soft-coded ones. Thus, all primitive organisms without conditional reflexes such as insects, amphibians, and so on mainly relate to this kind. Soft-coded organisms are those with conditional reflexes and are mainly mammals that can learn and in this way better adapt to changing environmental conditions.

The following enumeration of stages of ICE evolution –- from the simplest to the most complex — is in no way complete. Its only purpose is to provide a general description of the different kinds of ICEs without going into detail. It describes the types of information-related subsystems of naturally developed ICEs, but all these types can also be demonstrated on examples of artificial ICEs, because they are much more simple and transparent than the natural ones.

3.2.1 One varier ICE

The simplest ICE possesses either one or several independent sensors delivering information from the environment. The example of such an ICE is a unicellular organism (as in 2.3) which can react to a lowering the oxygen level by freezing its metabolism, and vice versa.

3.2.2 ICE with hierarchy of variers

Just imagine that the species of a microorganism described in 2.3 are transported by sea currents from the lagoon to the open sea and back. In order to survive they need to adapt themselves to changeable environmental conditions, activating their metabolism with increasing oxygen levels in the lagoon as well as increasing salinity in the open sea.

In order to do that, they need a third sensor recognizing the environment. Because water in lagoon is much warmer than that in the open sea, such a sensor can recognize the environment by measuring the water temperature.

In this case there is a hierarchy of activators with the upper sensor recognizing the environment and two subdued sensors recognizing the state of the respective environment and functioning only after being activated by the upper sensor.

3.2.3 Programmable ICE

Programming is normally associated with modern digital computers, when, in fact, its history can be traced back to medieval times. The earliest known programmable machines (i.e., machines whose behavior can be controlled and predicted with a set of instructions) were Al-Jazari's programmable Automata in 1206. One of Al-Jazari's robots was originally a boat with four automated musicians that floated on a lake to entertain guests at royal parties. Programming this mechanism's behavior meant placing pegs and cams into a wooden drum at specific locations. These would then bump into little levers that operated a percussion instrument. The result was a small drummer playing various rhythms and drum patterns [21].

Another example is the Jacquard loom, which was invented by Joseph Marie Jacquard in 1801 [22]. The loom simplified the process of manufacturing textiles with complex patterns. It had holes punched in a paste board, each row corresponding to one row of the design. Multiple rows of holes were punched on each card and the many cards that compose the design of the textile were strung together in order.

As an example of the simplest programmed ICE can be considered the modification of above defined three-varier microorganism, which lives its complete life in one environment. The environment of such a microorganism can be checked only once at the moment of its birth by measuring the water temperature and then storing the measurement in the internal variable. Instead of constantly measuring the water temperature, it only has to evaluate the state of its internal variable, which is in actual fact the simplest program controlling the behavior of such a microorganism.

This kind of program was probably not used very often at the beginning of organic evolution. The primitive programs were molded in neurons controlling the movements in multicellular organisms. Thus, even the simplest flying insect can do a lot of maneuvering during flight, which requires extended programs to control the movement of its body parts and process the information delivered with the help of miscellaneous external sensors.

3.2.4 ICE with base learning

Soft-coded organisms are those with conditional reflexes. Examples of primitive organisms using this feature are bees, which can remember the source of nectar and communicate it to other bees. Mammals possess an advanced form of this mechanism and are able to learn complex things and thus better adapt to changing environmental conditions.

3.2.5 Passively adaptive ICE

As soon as organisms became complex enough they started to optimize the learning process by imitating the behavior of their parents and other associates. In order to do that, they needed a way to establish the links between themselves and the associates. In the simplest case, this can occur as a result of inborn knowledge.

An example of such a link was demonstrated by the Austrian zoologist Konrad Lorenz in his famous experiment with newly hatched ducklings. He appeared before the newly hatched ducklings and imitated a mother duck's quacking sounds, after which the young hatchlings regarded him as their mother and thus followed him [23]. This occurs soon after hatching

because the young ducklings intuitively learn to follow someone who they identify as their parents. The process, which is called imprinting, involves visual and auditory stimuli from the parent object; thus eliciting a following response in the young that affects their subsequent adult behavior.

3.2.6 Actively adaptive ICE

More complex animals are able to acquire knowledge regarding themselves, which allows them to identify with other organisms having the same smell, the same appearance and same behavior. This is an ability e.g., great apes or pigs exhibit [24].

3.2.7 Language able ICE

The use of external language for exchanging information in the manner human beings do.

3.3 A human being as a biocomputer-controlled device

A formal view of a human being constructed in the sense of TMI is designated here as Homo Informaticus (HI). A HI is viewed as a self-programming system consisting of a brain (the biocomputer) with the rest of the human body controlled by this computer. The concept of HI is based on the view first formulated by John Lilly in 1968: "Programming and Metaprogramming in the Human Biocomputer" [25]. The model allows the expression of the complete information used by a human being, starting from the simplest perceptions and ranging to high-level intellectual functionality.

(1) A human being is seen as a self-programming system consisting of a brain (a biocomputer) with the rest of the human body controlled by this computer. The model allows the expression of the complete information content used by a human being, starting from the simplest perceptions and ranging to high-level intellectual functionality.

(2) The body of an HI is seen as a complex set of moving parts, each consisting of a muscle attached to a bone or an organ. A muscle executes one of two commands: contract or relax, which in turn causes a movement of a bone or organ. The command to contract sent by the brain by the intermediary of motor neurons increases the contraction of the muscle to a certain degree and the command to relax decreases the contraction.

(3) Senses are considered as systems of sensors functioning similar to the external receptors connected with the input ports of conventional computers. Change of a receptor automatically changes the state of its port. A feeling is a state of a sensor related port(s).

A group of receptor-port pairs could be viewed as an array of variables, whose states can be queried through their ports. Simultaneously, an HI receives information from millions of receptors organized in various arrays. Thus, cutaneous receptors of a finger can be considered a system of 13 arrays, that is, three fingers' phalanxes; each equipped with four arrays placed on each side and the last array placed on the fingertip (it would be 9 arrays for the thumb). Cutaneous receptors are stimulated when a surface with receptors contacts an external object, that is, the array receives a value composed of the values of all array elements.

(4) A simple program (routine) consists of a sequence of operators, which are either commands to muscles moving respective body parts or conditional operators making choices based on the changes of various receptors.

For example, making a fist is a set of actions including the contraction of the muscles of the four fingers and the thumb during which the brain is informed about the stages of muscle contraction and the physical contacts of fingers and the palm. Differing from classical computer programs, these are mainly parallel algorithms including simultaneous actions of several moving parts (all the fingers of a hand are clenched at the same time to make a fist).

Complex routines are skills such as walking, talking, riding etc.

The upper level is represented by activities, which are sequences of routines targeting a goal, like preparing food, going from point to another, studying a subject, etc. Activities can be extremely complex and take many years to achieve such as making a career.

The process of self-programming an HI starts immediately after birth and continues until death. This view also includes a detailed description of the self-programming algorithms as well as the ways in which these programs are used.

(5) Perception of external objects.

The execution of a program is accompanied by feelings generated by the various senses. Distinct executions of the same program produce distinct feelings if there are differences in participating entities. Making a fist e.g. will produce distinct feelings because of injuries and diseases of the palm and the finger. The main distinction, however, is the difference in the external environment: The different taste feelings that will be produced by eating different food, the different feeling of touch that will be produced by walking on different surfaces, the different sense of smell that will be generated by smelling different objects etc.

In contrast, the same external object in the same state will produce similar feelings during execution of the same program and these feelings are used by the HI's brain as the indication of these objects.

For example, an image of an apple includes such indications as visual images, taste, touch, and smell perceptions. The indications are generated during certain actions. Thus, when an HI moves closer to an apple it gets a sequence of visual images, each containing a larger and more detailed picture of the apple. Accordingly, when an HI backs away (or the apple is moved away), the eyes produce a series of diminishing apple images. Also the view is changed when the apple is seen from different angles revealing the part of the apple hidden in previous perspectives.

Other senses are involved when an HI comes into direct contact with the apple. Specific tactile sensations are received during the process of picking up the apple. The HI smells the specific smell when the apple is placed under its nose and a specific gustatory sensation when chewing the apple.

The plurality of indications of the same external object, which are produced during the execution of miscallaneous programs, create the internal image representing this external object. Images are sequences of feelings and muscle controlling commands, which are generated during the interaction between a HI and an external object and are stored in the internal data storage of a biocomputer.

(6) Internal images are the building blocks of the basic data representation system guaranteeing the effectiveness of an HI functionality. As soon as an HI sees an apple, the complete set of information associated with various algorithms such as moving towards it, picking up and chewing, is activated by the brain, enabling the HI to make a decision, for example, whether to pick up the apple.

Images create the layer of the private information (tacit knowledge) that cannot be directly communicated to other persons as it is built on the base of perceptions and commands to various muscles. Since different HIs have different experiences, they possess different internal images of the same external object. A general way to get the same tacit knowledge for different HIs is to compel them to study the same set of external entities in the same way, for example, using the same books, learning the same theories.

(7) Knowledge of various kinds (skills, beliefs, concepts) is represented by the programming code and data (perceptions).

 ISSN 1927-6974 E-ISSN 1927-6982

(8) The intellectual operations performed by the HI's brain are interpreted as manipulations with the code of these programs, and a natural language is considered as a system that enables programs to be exchanged between various individuals.

(9) A communicated information (language) consists of changes and their structures. A change is the sequence of states of the same external object produced by internal comparison algorithms from the information delivered by HI's senses.

4 Knowledge representation

The representation tool of TMI is the language T [30], which is built by extending the conceptual system of C++ to the representation of entities exceeding the C++ representation world. T allows code to be produced as executable (imperatives) and non-executable (narratives). The latter is the default code-form of this language.

T is not restricted by whatever predefined communicators or communicated themes and can be used for expressing any information ever formulated in any human or computer language. Assuming there is sufficient vocabulary, each text composed in any human language can be adequately translated in T in the same way as it can be translated to any other human language.

4.1 Semantics in TMI

Consider the following situation: some person buys some gadget and doesn't know how to use it. So, the person can read the user manual enclosed with this gadget and get the understanding from there. After reading the manual, the person can do as desired: use the gadget according to the provided instructions; give the gadget back if it doesn't answer the person's requirements; throw it away etc.

The semantics of a text in T is tantamount to the semantics of the user manual in the above example, i.e., the semantics of a text is the understanding provided to a reader of a text. A text in T can be composed by some writer and read by some reader for an arbitrary purpose, similar to a text in a natural language. The only difference is that because of its formal grammar such a text is mainly purposed for human-computer or computer-computer interactions.

The differences between the indicative and imperative semantics could be seen on the following example.

The sentence "the cup is put on the table" is a narrative describing a process of putting some cup on some table.

- The process can occur everywhere, anytime or even be imaginable
- The process is not confined to the producer or a reader of this sentence.
- It is implied that a reader of this sentence has to understand its meaning (semantics).

The sentence "put the cup on the table" is a command the sentence's producer gives to its reader.

- The process can only occur if a code reader is able to take the required cup and put it on the required table.
- The process designated by this sentence will only occur when a code reader actually executes the command.
- A code reader may do its function also without understanding the semantics of the sentence, e.g., when a code reader is a robot and this command is the one it can carry out.

Summing up, a reader of an indicative code has to be able to understand the meaning of the message, but it doesn't need to have a physical connection for the action's environment. To the contrary, a code reader of an imperative command may act without understanding the commands meaning, but it needs the execution environment.

4.2 Communications in T

The definition of T is based on the explicit view of the communication environment of C++. The latter is based on the following definition originally defined in 31. A programming language is essentially the machine code of the processor implicitly introduced by the definition of this language. Thus, the definition of the programming language Pascal presumes a certain logical processor, which runs programs composed in Pascal code; the definition of C++ presumes another logical processor that runs program composed in this language etc. Such a processor is called the innate processor.

The innate processor of a programming language is a pure logical construct that is not purposed for whatever physical implementation, but it completely defines the semantics of its language. Thus, a C++ program is considered as a message used in the communication between a programmer (code writer) and an innate processor of C++ (code reader). In the first approximation, an innate processor could be seen as a non-optimized interpreter executing a source program.

Consider the famous "Hello World" example:

```
void main()
{
    printf("Hello World");
}
```

This is a text message that a programmer sends to an innate processor of C++. The innate processor reads the code, finds the function `main()` and starts executing its content. The content consists of a text string requiring the invocation of the function `printf()`, which prints the message on the system monitor. The innate processor executes the function `printf()`and finishes its work.

The real implementation of C++ replaces the innate processor with a low-level physical processor without changing the C++ semantics. The real implementation uses a compiler for translating C++ source code into the low-level object code, a linker making the executable code from the object code and a physical processor running the executable code.

Differently from C++, T allows arbitrary number of code readers and code writers, whose semantics can be exhaustedly represented in the language itself. A T module can be attributed by a particular producer and/or reader. It is possible to specify a communicator with the structure and functionality of the innate processor of C++ , and compose the modules with the C++ semantics by specifying them in the following way.

```
_reader = _cplusplus
void main()
{
    printf("Hello World");
}
```

The general specification of the C++ innate processor can be found in [30].

Assume there is another logical processor registering operations performed by an innate processor of C++ and writing a protocol file in T. The protocol of the execution of the aforementioned example can be grammatically similar to the example but it is a pure narrative enumerating actually executed functions

```
_writer = _cplusplus_logger
main()
{
    printf("Hello World");
}
```

 ISSN 1927-6974 E-ISSN 1927-6982

4.3 Representation of meaning

Consider the following C++ example.

```
int  num1, num2;
bool isEqual(int, int);
…
bool bVar = isEqual( num1, num2)
```

This code defines:

- two integer variables num1 and num2;
- an external function `isEqual` comparing two arbitrary integer variables and making conclusions whether they are equal or not;
- a Boolean variable `bVar` holding the result of comparison of `num1` and `num2`;
- Below is the TMI interpretation of the above code:
- the type bool is seen as the simplest language whose only two sentences are `true` and `false`.
- The variable `bVar` is seen as an information carrier containing a message composed in `bool`.
- The output value produced by the function `isEqual` designates a meaning. The sentence "`true`" means "`num1` and `num2` are equal", the sentence "`false`" means "`num1` and `num2` are not equal".
- A message in `bVar` is a copy of the original value produced by `isEqual`. While the original sentence is an assertion proving the fact of equality, a copy is an assumption prone to possible distortions, because its value can be produced in any possible way.

In TMI meaning is a value of some type produced by a comparison procedure. A comparison procedure is the simplest observer. It compares an arbitrary number of entities, returning the designation of the comparison. The complex meaning is the collection of the simple and complex meanings. Complex meaning requires complex observers including many comparison procedures.

Meaning in TMI is represented with a special literal structure called info statement.

An info statement is syntactically similar to a function call extended by the result of a comparison as its first parameter. Thus the invocation "`isEqual( num1, num2)`" producing true is designated with the info statement

```
isEqual(true,  num1, num2); // this indicative sentence means that   "num1 is equal to num2"
```

Accordingly, the invocation producing false is designated with the info statement

```
isEqual(false, num1, num2);  //this indicative sentence means that    "num1 is not equal to
num2"
```

All complex descriptions can be reduced to the series of the info statements. Examples:

- "Green" in the phrase "the grass is green" has to be produced during comparison between the actual grass color and the set of available colors. Alternatively the same phrase can be produced by checking whether the grass is or isn't green.

- The phrase “a man was smiling” indicates two comparisons: first, the world “was” defining the moment of smiling relative to the moment of producing this phrase; second defining the face mimic as smiling as opposing to frowning, crying, etc.

- All changes are based on comparison between previous and current states of a changed thing. The phrase “the apple grows” assures that the size of the apple in the moment of producing this phrase is bigger than before. Taking into account that changes are elementary activities, used to express all complex processes, this feature actually enables adequate representation of all possible activities.

To express the process of the growth of an apple we need to compare two images of the same apple which can be expressed in TMI in the following way.

```
class Apple;                                    //the class describing an apple fruit
extern Apple old_image, new_image;              //two states of the same apple fruit

bool isIdEqual(Apple oldImage, Apple newImage);//the procedure checks the sameness of the
identifiers
                                               //of two images

enum sizeChange {grow, samesize, lessen};    //enumeration of all possible size changes
SizeChange szCheck(Apple oldIm, Apple newIm);//the procedure compares states of an apple

isIdEqual(true, old_image, new_image);     //the assertion states the  sameness identifiers
.
szCheck(grow, old_image, new_image);         //assertion of the apple's size dynamic
```

The above text is a narrative transmitting the information understandable by a HI which is the formal equivalent of a human being. The features of this text can be described as follows:

- The representation world of this code is the content of the data memory of an HI's brain (biocomputer). All human languages have the same representation world.

- The info statement informs about the outcome, which has to be produced by the procedure `szCheck()` in case it will be invoked with the given input parameters. Basically, this is an assumption stating that such a relation exists. Accordingly, it can be produced by either actually executing `szCheck()` and protocolling its input and output parameters or by directly producing this info statement without regard to any real measurement. In the first case, this info statement represents a proven fact, in the second, it is an assumption (proposition, statement etc.).

- A human knowledge is represented by the abstract declarations of HI's functionality. The procedure `szCheck()` is a general declaration, allowing many different implementations. Thus, the comparison can occur by comparing visual images of this apple, by comparing images generated by coetaneous receptors, by comparing the lengths, found by measuring the apple etc. Because different HIs can have different experiences, the exact meaning of this code can differ between HIs.

- The class `Apple` represents the collection of data and algorithms associated with an external entity known as an apple fruit. The class consists of series of images generated during the interactions between a HI and an apple. Several alternative image sequences can be generated during the execution of a single algorithm. Thus, a HI can get visual images, touch, and smell perceptions while grabbing an apple. In most cases a single sequence (e.g., visual images) is sufficient for identifying a particular external thing.

 ISSN 1927-6974 E-ISSN 1927-6982

- The semantics of this code is the part of the much bigger knowledge cluster. Thus, the detection of the apple's growth requires comparison of two images of the same apple produced at different times. That in turn, requires the algorithm allowing identifying the concrete apple, but different apples can hardly be identified by their appearances, because apples of the same sort look quite similar. An apple might be identified by its location on a branch, the branch by its location on a tree and the tree by its location in a garden. Hence an HI would only be able to register an apple's grow if it possesses knowledge about the features of such things as the garden, the tree, the branch. Furthermore it can involve other notions needed to define the above entities.

4.4 TMI representations versus traditional approaches

The purpose of T is similar to that of KIF, OWL as well as other languages of knowledge representation. The statement from the KIF specification "The basis for the semantics of KIF is a conceptualization of the world in terms of objects and relations among those objects" [26] can also be used for the characterization of T as well.

T allows producing arbitrary ontologies [29], part 5 for definition of events and time. The differences existing between the traditional representation languages and T can be considered using the example of the KIF representation of a physical quantity made by Thomas Gruber [27]. According to Gruber, a physical quantity can be defined as "a pair comprising a number and a unit of measure. The KIF form below defines a physical-quantity as an object consisting of a magnitude and a unit, which are given by the unary functions `quantity.magnitude` and `quantity.unit`. The definition says that the magnitude of a physical quantity must exist and be of type `double-float` and its unit must be a member of the standard set of units. These constraints are analogous to slot value restrictions in object-centered languages. Since the definition is an if-and-only-if (<=>) condition, it also says that every pair of such magnitudes and units defines a quantity.

```
(defrelation PHYSICAL-QUANTITY
(<=> (PHYSICAL-QUANTITY ?q)
(and (defined (quantity.magnitude ?q))
(double-float (quantity.magnitude ?q))
(defined (quantity.unit ?q))
(member (quantity.unit ?q)
(setof meter second kilogram
ampere kelvin mole candela)))
```

Physical-quantity is a class (i.e., a unary relation that holds over instances of the class). For describing individual instances of the class, we define a constructor function called the-quantity. The term expression (the-quantity ?m ?u) denotes a physical quantity `?q` whose magnitude is `?m` and unit is `?u`.

```
(deffunction THE-QUANTITY
(<=> (and (defined (THE-QUANTITY ?m ?u))
(= (THE-QUANTITY ?m ?u) ?q))
(and (physical-quantity ?q)
(= (quantity.magnitude ?q) ?m)
(= (quantity.unit ?q) ?u))))
```

The definition of physical-quantity already stated that all quantities are determined by the values of their magnitudes and units; this definition simply adds vocabulary to be able to denote a specific quantity with a term. For example, the following states that `X` is the quantity 3 meters, where meter is one of the possible units of measure.

```
(= X (the-quantity 3 meter))"
```

The following code contains excerpts of the functionally similar representation in T.

```
struct Double {…};     //has to be defined as a sequence of pictures on some surface
Double magnitude;
class<typename T> unit //unit has a procedure measuring some entity
{
    Double measuring( T subject ) = 0;
}

enum standardUnits unit{ meter, second, kilogram, ampere, kelvin, mole, candela);
class process {…}      //an entity with the feature time
unit<process> second;
class line {…}         //an entity with the feature length
unit<line>     meter;
…
```

As can be seen from this example, T code can be unlimitedly extended by missing definitions in the way of conventional C++ classes and so provide the necessary detailedness to any specification. The expression power of this language is guaranteed by both the superior C++ conceptual system and the object-oriented world-view of TMI. T doesn't use the mathematized approaches like first order calculus, semantic nets, conceptual graphs etc. because the common sense reasoning can be expressed in its terms.

5 Conclusions

(1) The keystone of the TMI is a definition of information which can be applied to information of every kind, level and complexity. This definition actually introduces a uniform view of all entities which are able to produce and use information. This view forms the basis for formal representation of all information (/knowledge) related tasks.

(2) Knowledge is viewed in TMI as the internal information content of an ICE that reflects the features of the ICE's environment, providing abilities for a flexible reaction to alteration of the environment. Any living creature or artificial device with the abilities for information employment can be viewed as an ICE which is able to acquire and use knowledge. The most capable of all known ICEs is a human being, whose information related capacities essentially supersede those of all other ICEs. ICE modeling human beings is designated in this work as HI.

(3) The model of an HI perceives a human being as a self-programming system commanded by the controlling computer (brain). The body of an HI is seen as the complex set of moving parts, each consisting of a muscle attached to a bone or an organ. A muscle executes one of two commands contract or relax, which, in turn, causes a movement of a bone or organ. Another important part of this system is the senses, which consist of multiple receptors interpreted as volatile variables functioning similar to the input ports of conventional computers.

(4) Human knowledge is represented by the programming code and data (perceptions) of an HI. The model provides the method for representing complete human knowledge starting from low-level perceptions that humans acquire during their lives and continuing until encompassing high-level knowledge. Various kinds of knowledge are interpreted as programs of distinct levels. The built-in algorithms of the lowest level are unconditional reflexes and basic skills, the low-level soft programs designate skills, and the algorithms of the high level stand for beliefs, concepts, etc.

(5) The intellectual operations performed by the HI's brain are interpreted as manipulations with the code of these programs, and a natural language is considered as the system enabling programs' exchange between various individuals. Only programs of a high level can be exchanged in a natural language, other programs constitute tacit knowledge, which can only be transmitted with the help of immediate learning if at all when one acquires new skills by copying the behavior of another HI.

 ISSN 1927-6974 E-ISSN 1927-6982

(6) The development of a human being from a baby to a mature individual is viewed as the process obeying the general law of software development: "the more software is implemented on a certain system, the more powerful this system is". At the beginning, there is only built-in functionality which is used for producing elementary programs, which in turn constitutes the building stones of the high-level entities.

References

[1] Lyons, John (1977). Semantics. Cambridge University Press, 619f; Yule, George (1996). The study of language. Cambridge University Press, 101.

[2] Sowa, John F. (2000). Knowledge Representation: Logical, Philosophical, and Computational Foundations. Pacific Grove: Brooks Cole Publishing Co., actual publication date 16 August 1999; Sowa, John F. (2010). Ontology. Available from: www.jfsowa.com/ontology/index.htm [last accessed December 28, 2012].

[3] Lu, C.-G. A Generalisation of Shannon's Information Theory. International Journal of General Systems. 1999; 28(6): 453-490. http://dx.doi.org/10.1080/03081079908935247

[4] Losee, Robert M. A Discipline Independent Definition of Information. Journal of the American Society for Information Science. 1997; 48(3): 254-269. http://dx.doi.org/10.1002/(SICI)1097-4571(199703)48:3<254::AID-ASI6>3.0.CO;2-W

[5] Shannon, Claude E. Shannon & Weaver, Warren (1963). The Mathematical Theory of Communication. Chicago.

[6] Cooman G. De/Ruan, D./Kerre E. (Eds.). Foundations and Applications of Possibility Theory. Singapore: WorldScientific. 1995.

[7] Devlin, Keith J. Logic and Information. Cambridge: Cambridge University Press. 1992.

[8] Stonier, Tom. Information and the internal structure of Universe: An exploration into information physics. London: Springer. 1990. http://dx.doi.org/10.1007/978-1-4471-3265-3

[9] Markov, Krassimir/Ivanova, Krassimira/Mitov, Ilia. Basic Structure of the General Information Theory. International Journal "Information Theories & Applications". 2007; 14: 5-19. Available from: www.foibg.com/ijita/vol14/ijita14-1-p01.pdf (last accessed December 28, 2012).

[10] Burgin, M.S. 2005. Is Information Some Kind of Data? Available from: http://www.mdpi.org/fis2005/F.08.paper.pdf (last accessed December 28, 2012).

[11] Chomsky, Noam. Three models for the description of language. IRE Transactions on Information Theory. 1956; 2: 113–124; Chomsky, Noam (1957). Syntactic Structures. The Hague: Mouton.

[12] Backus, W. The syntax and the semantics of the proposed international algebraic language of the Zurich ACM-GAMM Conference. Paris: ICIP Proceedings. 1959: 125-132.

[13] Pratt, Terrence/Zelkowitz, Marvin. 2000. Programming Languages: Design and Implementation (4th Edition). Prentice Hall.

[14] Smith, Michael K. Welty, Chris/McGuiness, Deborah L. (Eds.) (2004, February 10). OWL Web Ontology Language Guide: W3C Recommendation. Available from: www.w3.org/TR/2004/REC-owl-guide-20040210/ (last accessed December 28, 2012).

[15] Parmar, Aarati (2001, May 7). The Representation of Actions in KM and Cyc, Technical Report.

[16] Clark, Peter & Porter, Bruce. KM – The Knowledge Machine 1.4.0.: KM's Situation Mechanism. http://userweb.cs.utexas.edu/users/mfkb/km/situations.pdf (last accessed December 28, 2012).

[17] Hume, David (1958(1739)). A Treatise of Human Nature. London: John Noon, 662.

[18] Causality. In: Random House Webster Unabridged Dictionary. 1996) Change. In: Random House Webster Unabridged Dictionary (1996).

[19] Force. In: Encyclopedia Britannica Ultimate Reference Suite. 2009.

[20] Best, Ben. The Anatomical Basis of Mind. Available from: www.benbest.com/science/anatmind/anatmind.html (last accessed December 28, 2012).

[21] Gabarin, Ahmad Yousef al-Hassan. History of Science and Technology in Islam. Al-Jazari and the History of the Water Clock. Available from: http://www.history-science-technology.com/articles/articles%206.htm (last accessed December 28, 2012); History of Computers. The Dream for Thinking Machine. The Arabic Automata. http://history-computer.com/Dreamers/Arabic.html (last accessed December 28, 2012).

[22] Computer: The Jacquard Loom. In: Encyclopedia Britannica Ultimate Reference Suite (2009).

[23] Lorenz, Konrad. In: Encyclopedia Britannica Ultimate Reference Suite (2009).

[24] Gallup, G. G. Jr. Chimpanzees: Self Recognition. Science. 1970; 167(914): 86–87.

[25] Broom, Donald, Sena, Hilana, Moynihan, Kiera L. Pigs learn what a mirror image represents and use it to obtain information. Animal Behaviour. 2009; 78(5): 1037-1041.

[26] Lilly, John C. (1968). Programming and Metaprogramming in the Human Biocomputer: Theory and Experiments, Communication Research Institute.

[27] Knowledge Interchange Format, Paragraph 5.1 draft proposed American National Standard (dpANS), NCITS.T2/98-004 http://logic.stanford.edu/kif/dpans.html#Terms

[28] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? International Journal of Human-Computer Studies. 1995; 43(5–6): 907-928.

[29] Boris Sunik. 2011. Theory of Meaningful Information. Available from: http://generalinformationtheory.com/theory.*php;

[30] Boris Sunik. 2011. Theory of Meaningful Information: Chapter 4: The Universal Representation Language T. Available from: http://generalinformationtheory.com/content4.php.

[31] Boris Sunik. 2003. The paradigm of OC++. SIGPLAN Notices. 2005; 38(6): 50-59. The language T. SIGPLAN Notices 40(5), 28-38.

[32] Boris Sunik. 2012. The Ultimate Representation of C++ Semantics. Available from: http://generalinformationtheory.com/cpp.php.