ARCHITECTURE AND DESIGN OF MICRO-KNOWLEDGE AND MICRO-MEDICAL PROCESSING UNITS

Shawon S. M. Rahman, Ph.D.
Associate Professor, Department of Computer Science and Engineering
University of Hawaii-Hilo, Hilo, Hawaii 96720, USA

Syed V. Ahamed, Ph.D.
Professor Emeritus, City University of New York,
Staten Island, New York 10134, USA,

ABSTRACT

In this article, we briefly present the evolution of conventional processor over the last four decades as a prelude to the evolution of knowledge processors. A system of specially designed chipsets from the traditional computer architectures facilitates the solution of most generic problems in sciences and in the society. The design of such chipsets permits the micro knowledge processor unit (μkpu) lodged in a generic knowledge machine to "understand" the context of the problem in reference to the global knowledge of such problems in the world wide web. Micro-processing within the more complex procedures such as knowledge and library functions is thus reduced to logical and then into consecutive hardware functions in the knowledge processors. Medical procedures are used for illustration of the potential of the μkpus. Such medical procedures, sub-procedures and micro-procedures can be performed upon medical super objects, objects, subordinate objects and micro-objects.procedures fragmented sufficiently for the μkpus, execute assembly level type of knowledge micro instructions. Conversely, the micro instructions reassembled vertically, in a hierarchy perform major procedures. Major medical procedures such as removing malignant tissues, curing a patient, performing surgeries, etc., bring about profound changes in patients. Minor procedures represent minor effects such as authorizing a prescription, recording the temperature or blood pressure, etc.

KEYWORDS

Knowledge Processors, Micro Knowledge Processors, medical sensor network, CPU, knowledge instruction, Micro medical processors

1. INTRODUCTION

The processor revolution in the computer industry is well over the three decades old. It is indeed comparable to the automobile revolution in the transportation industry. Both have influenced modern lives. The processor and the ensuing information revolution over thesefew decades is at a much faster pace. The influence in the human perceptual domain is insidious and rather alarming.

DOI: 10.5121/ijnsa.2017.9501

Table I. Time Line for Better Known Intel Processors, Chips and Chip Sets

	Yr	INTEL-Processors	In. Clock	Transistors	Technology (Manufacture)
1	1971	Intel® 4004 Processor	108KHz	2300	10
2	1972	8008®	800	3500	10
3	1974	8080®	2000	4500	6
4	1978	8086®	5000	29000	3
5	1982	286®	6000	134000	1.5
6	1985	386®	16000	275000	1.5
7	1989	486®	25000	1200000	1.0 micron
8	1993	Pentium®	66000	3100000	800 nm
9	1995	Pentium® ProProcessor®	200000	5500000	.350 nm
10	1997	Pentium II®	300000	7500000	250
11	1998	Celeron Processor	266000	7.5E+06	250
12	1999	Pentium III®	600000	9.5E+06	250
13	2000	Pentium 4®	1.5E+06MHz	42.E+06	180 nm
14	2001	Xeon Processor®	1.7E+06	42.E+06	180
15	2003	Pentium M®	1.7E+06	55.E+06	90
16	2006	2 Core Duo Processor®	2.66E+06	291.E+06	65
17	2008	2 Core Duo Processor®	2.4E+06	410.E+06	45
18	2008	Atom Processor from Q2/08 to Q4/13 53 variations exist	1.86E+06	47.E+06 (Varies)	45
19	2010	2nd Generation Core Processor	3.8E+06	1.16E+09	32
20	2012	3rd Generation. Core i3 Processor	2.9E+06	1.4E+09	22
21	2013	Fourth Generation Core® 10 versions of i5 & i7 processors	2.2 to 4.0E+06	Variable	22 nm
22	2014	Fifth Gen. core processor with Iris® Pro (Broadwell)	Adaptable	Variable	.014

Processors have started to become dramatically more powerful since the days of Intel 4004 (1971) to 286 (1982), 486 (1989), Pentium (1993), Atom (2008), through the Fifth Generation Iris Pro Broadwell (2014) processors. The Atom Processor has 53 versions from 2nd Quarter of 2008 to the 4th Quarter of 2013. The clock speeds have made variable and the transistor count is matched for the numerous applications. Table 1 indicates a range of chips and chip sets that have become the products of Intel Corporation. Table II presents the details of the well known Processors Penyrn, 2 Dou Core, Pentium 4,Intel Pentium Pro (P6), Intel Pentium (P5) and

Nahalem Quad core processors that have appeared in the market place from 2007 to 2010.The context to the human being is at the emotional, psychological levels. Information, knowledge and concepts provide garments of personality to the raw human mind. In a sense, other higher-level needs realization needs (Maslow [1]), search and unification (Ahamed [2], [11], [13]). To this extent, the information, knowledge and concepts processing in the of the next generation processors will influence the emotional, psychological and mental health of the generation to come, much as the evolving medical processing is influencing the physiological health of the current generation. Processing in the modern context is essential at any level of human need [1, 2] much as hunting and gathering processes were at the survival level of the prior generations. Complex processes in humans and computers are amenable to the science of morphology1. In the present context of processors, identification, analysis and description of any process are feasible at two levels: (a) at the operation code level and (b) at the operand level.

The conventional opcodes (opcs) are a group of finely further subdivided opcodes that constitute a (verb) function directed at one or a set of operands that will experience the opcode in the processor unit of any given processor. At the operand level, operands can be numbers, logical entities, plain objects, (noun) objects, data structures, sentences, paragraphs, books, etc. Computer scientists have classified single instruction (SI) and multiple instruction (MI) systems much as they have classified single operand (SO) and multiple operand systems (MO). The four architectures derived from SISD, SIMD, MISD and MIMD configurations [3, 4] constitute the four standard CPU designs for typical computer systems.

The processor development trend documented in Table I is compiled from the documents published by Intel and is in the public domain. Graphs of the major movements are depicted in Figures 1.a through 1.d. Few observations are evident. The earlier integration of the processors (1971-2000) follow an exponential growth pattern expected rewards of early integration from MSI to LSI and then to the VLSI. During the later years as Intel started the Dual and Quad processors the transistor count also increased double and four fold. The Logarithmic rate of increase of the Transistor count (Figure 1.b) also reflects this chip integration procedure. During the last few years new customized chips for different applications have needed fewer transistors but the production of Specialized CPU chips continues as tabulated in Table II.

Table II Intel's Penryn Duo and Pentium Family and Nehalem Processors

Transistor Count	Die Size	Manufacturing Process
Intel Core Duo 410M	107 mm^2	45 nm **(Penyrn)**
Intel Core 2 Duo 291M	143 mm^2	65 nm

1 Morphology is the systematic breakdown, structural identification, analysis and description of morphemes and other units in any discipline such as linguistics, biology, etc. Words in languages, organisms in biology, shapes and forms in images and even ideas and concepts in the human mind can be broken down, identified, and described. We extend this chain and go on to reconstitute the operation code and the operands in computer science and thus breakdown, decompose, reassemble, reconstitute objects to generate new objects, images, and even new ideas, inventions and evolve new concepts. This is the goal of most machines presented in this book.

The clock-rates have also increased dramatically and depicted in Figure 1c offering reduced process time by almost six and half orders of magnitude (108 KHz to about 4 GHz) over the last 4 decades. The manufacturing technology has been reduced from 10 microns (1971) to 22 nm (2012) going down to 14 nm in 2014 and depicted in Figure 1d. The combined effects is that of less expensive, faster and more compact and more powerful processors that have dominated the CPU markets for last two decades and likely to become more adaptable to performing complex computational (scientific) and social based human functions provided the appropriate software and social layers become available.

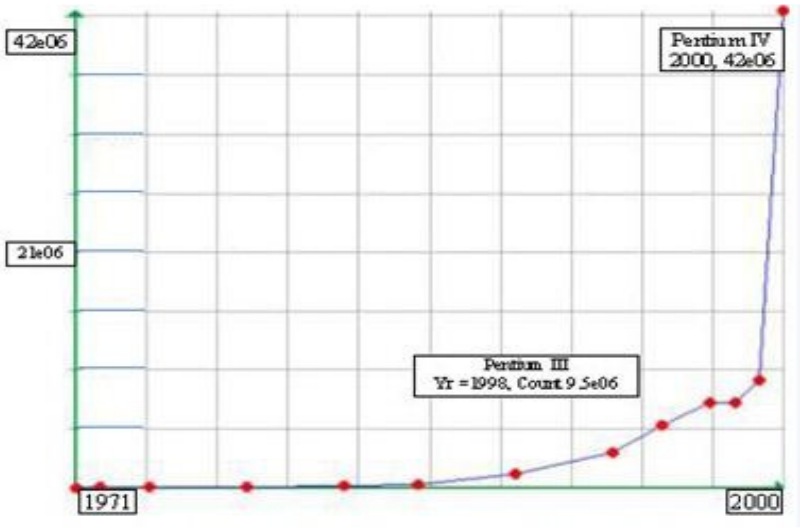

Figure 1.a : Increased Transistor Count within the
Intel Processor chips from 1971-2000.(See Table 1)

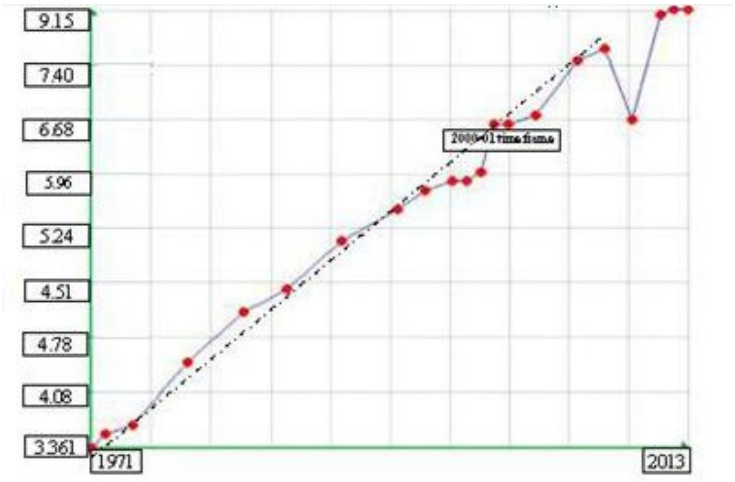

Figure 1.b : Logarithm of the increased Transistor Count within the numerous
Intel Processor chips over the last four and half des (1971-2013).

In the humanist domain, processing becomes more complex due to the assistive, passive or resistive nature of the human operands in the nature of operations. Such operations can be cooperative, neutral or coercive. Hence, the classifications become more cumbersome and exhaustive to determine the effect of one human, robot or machine "operator" operating upon another human, and robot or machine "operand". An in depth classification of such operations (or verb functions) on human and semi-human operands (or noun objects) is presented in [5]. Processors presented in Section 5 transform current objects and convert them into new objects to suit different applications and scenarios in the real world.

Increase in the clock rates for Processors 1971-2000.

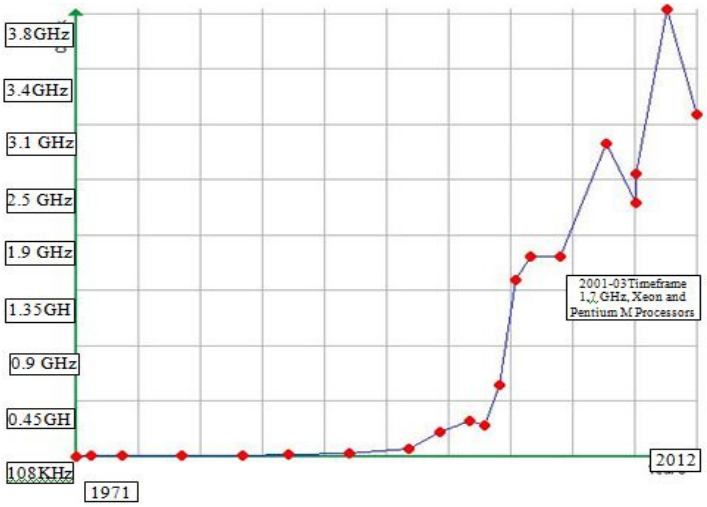

Figure 1.c : The increased Clock Rate of typical Intel Processors over the last 4.5 decades

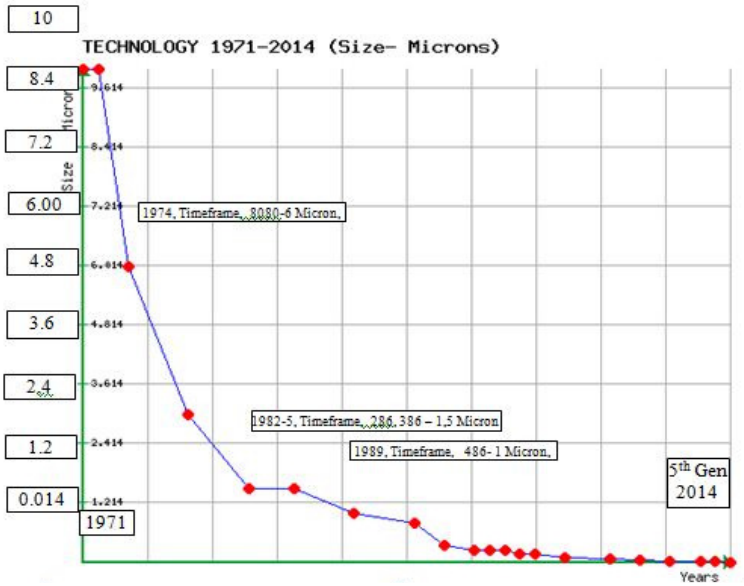

Figure 1.d : The reduction of the Manufacturing Technology over the last 4.5 decades for typical Intel Processors, thus contributing to the reduced chip size for the numerous processors.

2. KNOWLEDGE PROCESSING UNITS (KPUS)

Object and knowledge processing have a significant functional and thus architectural commonality. Both OPUs and KPUs process objects, their attributes, their inter-relationships and their structure. Both interpret, recognize and alter the properties of objects per se. Knowledge is indeed derived from objects, their nature, attributes, and their interactions. Thus, the processing capability of knowledge entails processing objects, their attributes and object interactions. The KPUs process further to detect the structural relations between objects and their properties (i.e., their attributes, their inter-relationships and their structure) and derive new properties, concepts, and perhaps insight and inference. Numerous design of KPUs become evident and in fact, they can be derived from the varieties of CPU's initially and then the CPUs that function as GPUs, and the finally the CPU's that can also serve as CPUs and GPUs The creativity of the individual KPU designer lies in matching the HW architecture to the application needs. The newest CPU architectures serve as ideal HW, FW and MW for KPU chips. KPUs being more expensive and processor intensive than CPUs are unlikely to become as numerous as CPUs that can be personalized to any whim and fancy of the chip manufacturers. The function of the KPUs depends on the capacity of the HW to manipulate or juggle (global and local) objects, based on their own syntax, and environmental constraints in the semantics of the user objective.

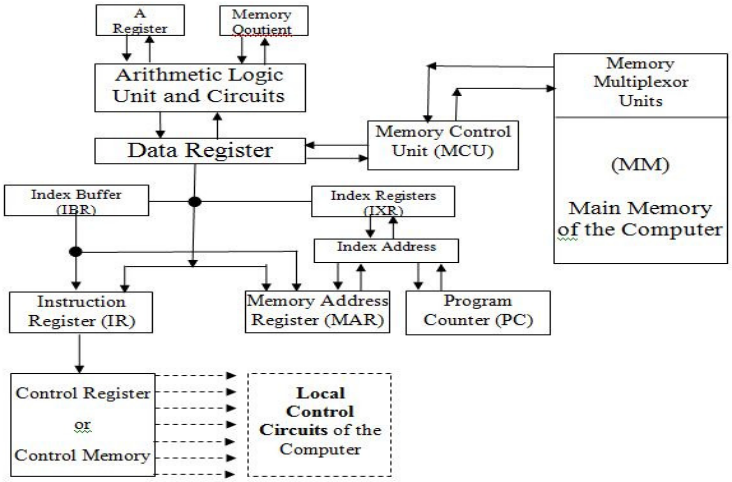

Figure 2: Representation of a Single Instruction Single Data (SISD) Processor of a typical computer that operates on a single piece of data at one time.

[2]A brief explanation of acronyms for this section is presented. CPU = central processor unit; KPU = knowledge processor unit; ALU = Arithmetic logic unit; NPU = numeric processor unit; SIMD = single instruction, multiple data; MISD = multiple instruction, single data; ,MIMD = multiple instruction, multiple data; OPU = object processor unit; SKI-SO = single knowledge instruction-single object; SKI-MO = single knowledge instruction- multiple object; MKI-SO = multiple knowledge instruction- single object; MKI-MO = multiple knowledge instruction- multiple object; SO processor = single object processor; MO processors = multiple object processors; SOI-SO = single object instruction-single object; SOI-MO = single object instruction-multiple objects and MOI-SO = multiple object instruction-single object; MOI-MO = multiple object instruction-multiple object SKI-SOI-SO = single knowledge instruction-single object instruction-single object; MKI-MOI-MO multiple knowledge instruction-multiple object instruction-multiple objects.

The computer itself can handle arrays of data that are stored in the main memory as indexed variables such as X(m,n). When this simple SISD layout of a typical CPU, is rearranged to handle Objects instead of data, and the binary machine instructions are rearranged to handle Object Instructions, then the system can be forced to work as a simple von Neumann Object Computer or a low level indexed Object Machine. The concept is presented in Figure 3.The CPU's functionality depends on the capacity to execute stylized operation codes on arithmetic and logical operands (in highly specialized formats and date structures). The configuration of a simple KPU is shown in Figure 3. Other variations based on SIMD, MISD, and MIMD variations of the CPU architectures can be built. Object processors that lie in between CPUs and KPUs bring in another degree of freedom because KPUs can deploy OPUs, much like CPU's can deploy ALUs and NPUs. Sequential, pipeline, and parallel execution of operations on objects in KPUs gives rise to at least eight possibilities: SKI-SO processors, SKI-MO processors, MKI-SO processors, and MKI-MO processors. Now if SO and MO processors have SOI-SO, SOI-MO and MOI-SO (pipeline structure), and MOI-MO (pipeline and/or multi-processor structure) have variation embedded within themselves, then at least eight design variations become evident. The SKI-SOI-SO is the simplest to build while the MKI-MOI-MO is the most cumbersome to build. From the first estimate, the HW for the simplest KPUs should be an order of magnitude more complex the IBM 360 CPU's (even though these CPUs deployed the microcode technology).

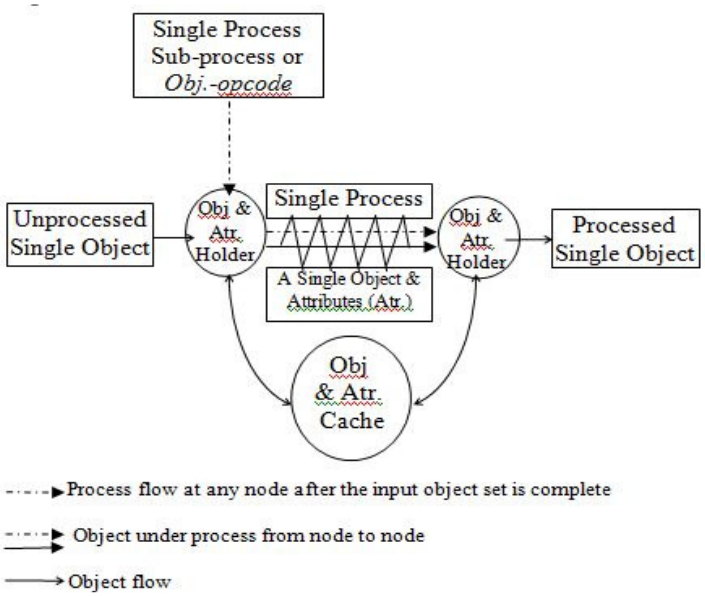

Figure 3 : Simplified representation of a Single Process Single Object (SPSO) Processor that operates on a single object and its attributes. When it replaces the FPU and ALU in the SISD layout of a typical CPU, the system can be forced to work as a simple von Neumann object computer.

Knowledge processing is based on rudimentary knowledge theory [5][34]. Stated simply, human knowledge is clustered around objects and object groups. Such objects can be represented by data and information structures. Data has numerous representations and information has several forms of graphs and relationships that bring order and coherence to the collection of objects. Such a superstructure of data (at the leaf level), objects (at the twig level), the object clusters (at the branch level) can constitute a tree of knowledge. Specific graphs moreover, relationships that bind information into a cogent and coherent body of knowledge bring (precedent, antecedent, and descendant) nodal hierarchy in a visual sense that corresponds to reality.

Knowledge processor [4][11] units should be able to prune, build and shape, reshape and optimally reconfigure knowledge trees, much as CPU's are able to perform the arithmetic (and logic) functions on numbers and symbols and derive new numbers (and logical entities) from old numbers (and logical symbols). All the most frequently used knowledge functions need a review for the KPU to perform the basic, elementary, and modular functions on objects. In the design considerations of the CPU, the more elaborate AU functions are known to be decomposable into basic integer and floating-point numeric (add, divide, etc.) operations. Similarly complex logical operations can be reconstituted as modular (AND, OR, EXOR, etc.) functions.

Knowledge bearing objects can be arbitrarily complex. Numerous lower level objects can constitute a more elaborate object entity. Like bacterial colonies, knowledge superstructure have dynamic life cycle. The order and methodology in the construction and destruction of such knowledge superstructures leads to "laws of knowledge-physics" in the knowledge domain under the DDS classification 530 to 539 Traditional laws of Boolean algebra and binary arithmetic do not offer the tools for the calculus of the dynamic bodies of knowledge undergoing social and technological forces in society.

However, if the new laws for the flow, dynamics, velocity, and acceleration of knowledge can be based on a set of orderly, systematic, and realistic knowledge operation codes (kopcs), then these laws can be written as machine executable routines that operate on the knowledge bearing objects. This approach is a bold digression from the approach in classical sciences where the new concepts enter the sciences as symbolic and mathematical equations. In the present society, information is exploding as multimedia WWW streams, rather than gracefully expansion in coherent and cogent concepts embedded in information. Time for extensive human contemplations is a rare luxury. Much as we needed digital data scanning systems for DSPs in the past, we need a machine-based common sense, sensing systems to separate junk level information [5]) from knowledge bearing information. Knowledge filtering, discussed in [5], accomplishes these initial, robust, sensible, and necessary humanist tasks [33][34].

The current scenario of science and innovation has given rise to the deployment of technology before it gets obsolete. To accommodate this acceleration of knowledge, we propose that we have a standard set of basic and modular *kopcs*. The complex knowledge operations that encompass the newest concepts are then assembled from the basic set of internationally accepted standard of *kopcs*. The proposed representation for the dynamics of knowledge paves the way between concepts that create new knowledge and the technology that uses such knowledge.

3.THE MICRO KNOWLEDGE PROCESSOR UNIT

Micro knowledge processors are the functional equivalents of control memory controlled central processor units. The assembly level instruction on typical CPUs is taken over to the particular address in the control memory and the micro instructions are executed upon the operand to complete the original assembler level instruction.

[3] It is proposed that any verb function can be performed in its own context and syntax thus leading to a (L x L) matrix in Figure 4. It appears impossible to perform any function on any noun in an unknown context and in an unknown syntax.

In the knowledge environment, the process gets cumbersome because the knowledge objects are data structures with other objects related to the main knowledge centric object (KCO) and each of the objects can have attributes and customized relationships. Hence the micro KPU needs to perform all the that HW, and closely associated SW functions such as performing the syntactic and semantic checks and making sure that the micro *kopcodes* are fed into the micro knowledge processor at the execution time [15].

Most registers in the traditional CPUs need the structure of cache memories or stacks. These enhanced structures substructures to hold other related objects caches or their addresses. The affected attributes of objects also need address and data space-store the effect of *kopc* in any knowledge assembly-level instruction.

The basic concept behind the micro KPU is shown in Figures 4, 5 and 6. These diagram is based on three laws dealing with (a) the concept of segmentation of (verb) functions, Figure 3, (b) that only selected actions can be implemented on corresponding noun objects, Figure 4 and (c) that behavior is founded on convolution (*'s) of verb functions (VF's) upon noun objects (NO's) . The computational framework to convert these ideas is presented in the following section.

- *Fragmentation:* Larger actions are based on decomposable smaller actions
- *Grouping:* Actions and objects are grouped and they abide by the syntactic and semantic laws of behavioral grammar.
- *Convolutions:* Human behavior that results from (VF ⊛ NO) abides by Marshals Law in maximizing derived utility of the resources expended.

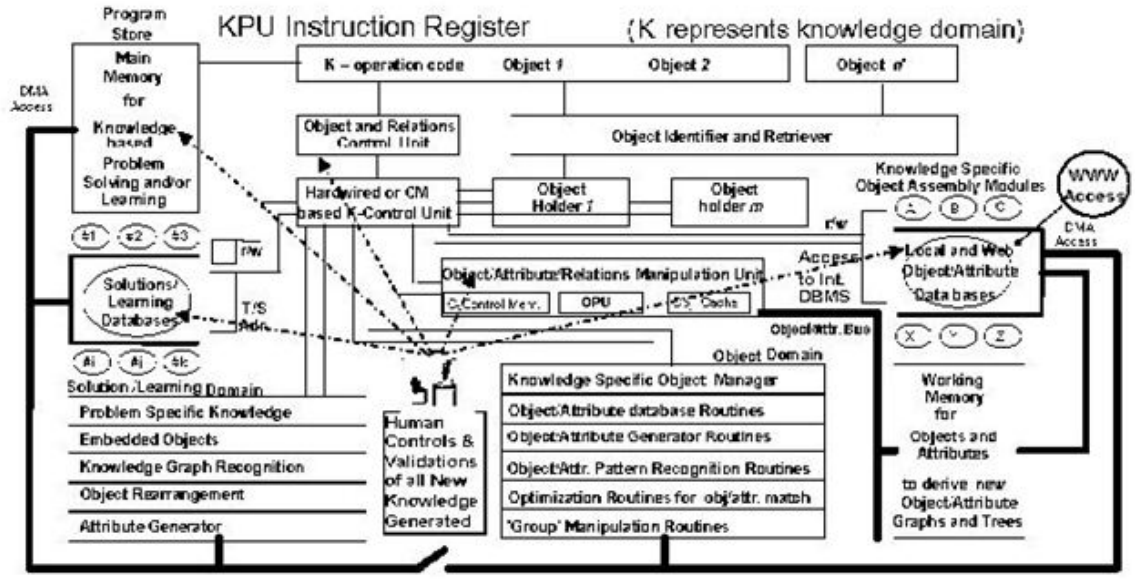

Figure 4 : Switch S-1, Open for Execution Mode for Knowledge Domain Problem Solving; Closed for Learning Mode.

The Learning programs 'process' the existing solutions and are able to extract objects, groups, relationships, opcodes, group operators, modules, strategies, optimization methodologies from existing solutions and store them in object and corresponding databases. The architecture permits

the KPU to catalog a new object in relation to existing objects and generate/modify existing pointers to and from new objects.

3.1 Effect of Fragmentation of Verb Functions

Human Behavior is a set of associated and smaller subsets of well-sequenced actions. Any action can be broken down into smaller action. Each smaller action can be written as:
VF = ∑ vfi where i ranges from 1 to L steps;

stated alternatively, VF = A column[2] of vf_i's; i ranging from 1 to L smaller *vf's*.

Actions involve at least two components: an active object ('doer') and a passive object ('done upon'). The 'doer' can be a human being whose behavior is being considered and 'done upon' is the old noun object NO. When the *kopc* is completed the new noun object NO' is generated in the micro KPU. There are at least two matrix functions involved to generate the new noun object NO'. The new object NO', is evolved after the $\mu kopc$ knowledge-operator matrix is multiplied by the old noun object matrix, NO. See Figures 3, 4 and 5.

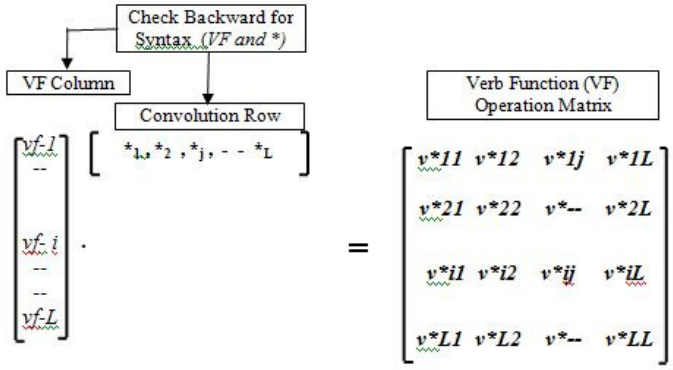

Figure 5 : Generation of VF operation matrix from verb function VF and its corresponding convolution ⊛.

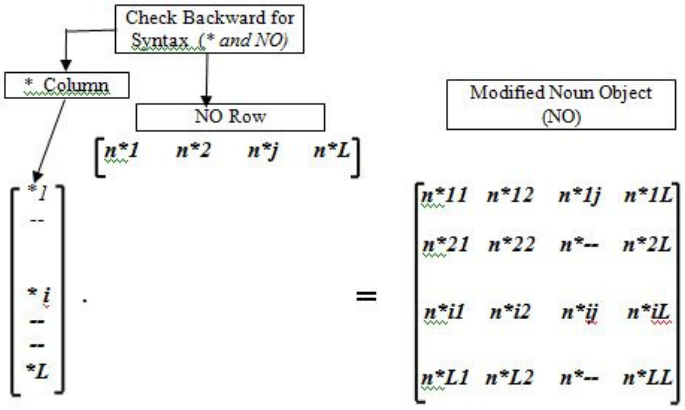

Figure 6 : Convolution ⊛ operation on the original row of noun objects NO and the resulting matrix of modified NOs .

In general, knowledge application programs (KAP) accomplish major functions in the knowledge domain. When a KAP is compiled and assembled, a series of steps in the smaller knowledge assembly-language program (KALP) are generated after syntactic check and semantic check for all the knowledge objects in the KAP. These steps in the KALP follow a series of well-defined fetch-decode-execute (FDE) cycles as the program is executed. Any knowledge level program/subprogram/mini-program, micro-program etc., can be written as a series of fine steps of the nature VF⊛NO.

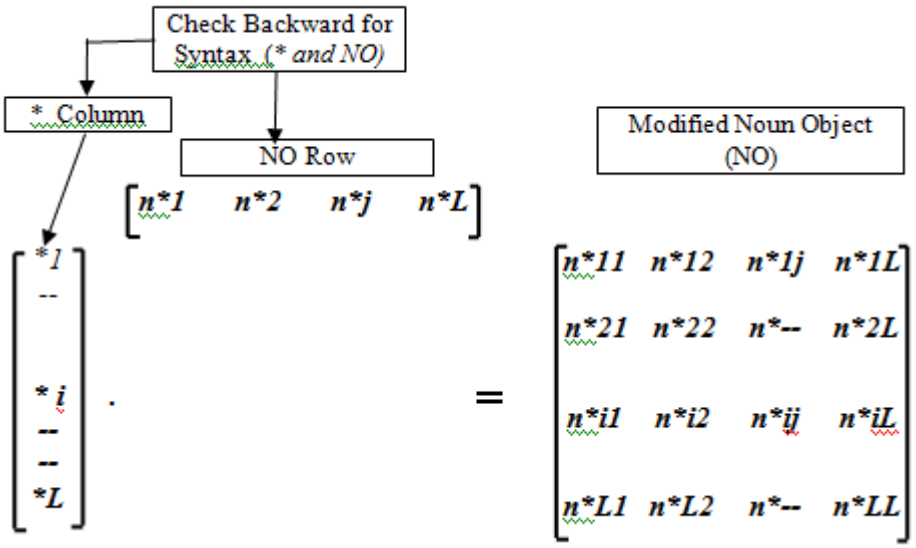

Figure 7 : Generation of the New Object NO' and the mathematical representation of a micro knowledge operation (μkopc) as it is executed on an old noun object NO to evolve a new noun object NO'

3.2 Effect of Grouping of Noun Objects

Noun objects generally consist of other noun objects and they are represented as trees or as matrices. When an action occurs on an object NO, the other noun objects in the group also get affected. To represent this effect, a matrix multiplication is suggested where the VF operation (L x L) matrix is multiplied by (L x L)

The law of fragmentation permits the breakdown of VF, ⊛ and NO. The algebraic representation of KAP is shown in Figure 8. It becomes essential to perform a syntactic and semantic check before any KAP can be executed in a computational environment. The syntactic check ascertains that the process requested can be performed in the HW and SW constraints of the computer and the semantic check ascertains that the process requested is indeed valid and legal in context to the noun object upon which the process is being executed.

4 .ARCHITECTURE OF A CONVOLVER TO PERFORM * FUNCTION

Knowledge processing is more complex than data or text processing and the implication of each step of knowledge processing needs scientific and/or economic justification. In addition, if a KAP calls for an optimization, then the utility of each of the steps of the KAP needs to be evaluated and tallied against the resources required for that particular step or procedure. When only one VF or vf_i, is convolved with one noun object NO or no_k then a simple "convolver" (see Figure 9) will suffice[3] to bring about a (VF⊛NO) or ($vfi \circledast_j no_k$). Other architectural variations, such as one *vf*, multiple *nos* (equivalent of SIMD, or MISD, or MIMD) configuration, etc., can be derived.

In a wider sense, the utility of the process (VF⊛NO) needs to justified. The derived utility is determined solely by the natures of (VF, ⊛, and NO) or by the natures of (vf_i, $\circledast_j$, and no_k). In reality, however, a series of no_k's are involved since the process yields different utility for each no_k. Similarly, the utility is different for each possible variation of vf_i , $\circledast_j$ and NO_k. When these options are available, the utility of each combination is evaluated to find which combination or solution yields the best utility or results. Three different caches for vf_i , $\circledast_j$, and no_k are necessary and shown in Figure 10. Other architectural variations (such as one *vf*, multiple *nos*, multiple *vfs*, multiple *nos*) of the micro knowledge processor will require multiplicity of stacks for *VFs,* ⊛*s,* and *NOs*.

The computation or the estimation of the expected utility of any *μkopc* or micro-knowledge function becomes necessary if the process involves optimization or the selection of the best utility that will result from one or more combinations of vf_i , $\circledast_j$, and no_k . A series of vf_i , $\circledast_j$, and no_k are stored in their respective caches and the process then selects the best combination. of vf_i , $\circledast j$, and no_k . The resulting utilities are illustrated in Figure 11.

$$VF * NO \equiv \sum_{i=1}^{i=I}(vf_i) \bullet \prod_{i=1}^{i=M}(*_i) \bullet \sum_{\ell=1}^{\ell=N}(no_\ell)$$

$$[VF \equiv \sum_{i=1}^{i=I}(vf_i)], \quad [* \equiv \prod_{i=1}^{i=M}(*_i)] \text{ and } \quad NO \equiv \sum_{\ell=1}^{\ell=N}(no_\ell)$$

Figure 1 : Algebraic representations any knowledge process ranging from a major knowledge program to a Micro knowledge operation (μkopc). It implies that any verb function (VF) can be decomposed into a series of smaller (verb) functions, any process (⊛) consists of a series of micro processes, and the effect of the process on noun object (NO) is the combined effect of the micro processes on each of the elements (no's) that constitute the main noun object NO.

[4]The term "convolver" is designated as HW that enforces a verb function on a noun object. It is akin to an adder or multiplier. In the knowledge domain, at least one verb and one noun is implied in every statement that modifies an object. The implication is that an active verb modifies a noun or alters its state. The action of the verb varies dramatically depending on the verb and the noun and become a convolution rather than a simple add, subtract, multiplier, divide, etc., or any logical, vector, matrix, etc., function. Hence the symbol is chosen as a * and written between *VF* and *NO*. The convolver in its simplest form would be an ALU, vector, graphic, signal, etc., processor depending on the syntax and semantics of its usage. In more complex form it could be a digital amplifier, a sensor or a synthesizer.

If the knowledge program KAP calls for combining the three elements (*VF,* ⊛, and *NO*) from local knowledge bases (KBs) with the corresponding elements from the WWW KBs to find the best solution for the KAP, then the coordinates of each of the elements in the local and the WWW KBs need accurate tracking. The utilities also need to be computed and tracked. The effort can be time and resource consuming. Hence the tacking of the local sets of (vf_i , ⊛$_j$, and no_k) modified by the corresponding and documented WWW sets of (vf_i , ⊛$_j$, and no_k) are stored in a 3 D matrix of the coordinates of the three elements of knowledge for the final human evaluation of the derived utilities. One of the possible configurations of this type of μ knowledge processor is shown in Figure 10. Fine-tuning of the final solution is thus postponed to human judgment in view of committee of joint decision.

One of the bye-products of this type of micro-management of the composition of knowledge is the enhanced creativity in the optimal combinations of the two sets of local and WWW (vf_i , ⊛$_j$, and no_k). New VFs, ⊛'s and NO's will be evolved to maximize the utility in one or more directions. When the criterion for selection is social benevolence, the machine yields the best social innovations. When the criterion for selection is best molecular structure for drugs (allergies), the machine yields possible sets of drugs innovations (for allergies), etc.

5. CASE STUDY: MICRO-MEDICAL PROCESSING

In the medical domain, if a medical assembly language program (MALP) that is generated after compiling, assembling, loading and linking the medical application program (MAP), is represented[4] as "summation" (or ∑) of basic medical instructions (BMI)

Typically, a series of nok's are involved since the process yields different utility for each nok. Similarly, the utility is different for each possible variation of vfi and for ⊛$_j$. When these options are available, the utility of each combination is evaluated to find which combination or solution yields the best utility or results.

$MALP = \sum_1^{nm} BMIs =$
or

Any Medical Application Program = $\sum_1^{nm} \{MVF \circledast CorrespondingMNOs\}$,

then, the FDE repetitive cycle during the execution of the MALP follows the steps outlined for the KLP. In addition, the detailed description, identification and authorization of all medical objects, the particular patient, the doctor(s), the staff, the drugs, the therapy, the instruments, the equipment, the diet, the test results and the security and privacy issues become the "medical nounobjects" or *MNOs*. The procedures, the instrumentation, therapist's treatments, etc., become the "medical verb functions" or *MVFs*.

[5]In a conceptual framework, the following equations simply state the any major medical treatment or procedure for any patient is an organized collection of minor processes or procedures. The sequential or parallel nature of these minor functions is embedded in the symbol ∑ that forces the medical machine to complete the major medical treatment or procedure in view of the limitation of the resources of the medical facility, staff, and the knowledge bases of the specialists, doctors, and staff. The medical machine expands these series of minor medical processes into executable program code that can be executed in the medical facility by the staff and medical resources.

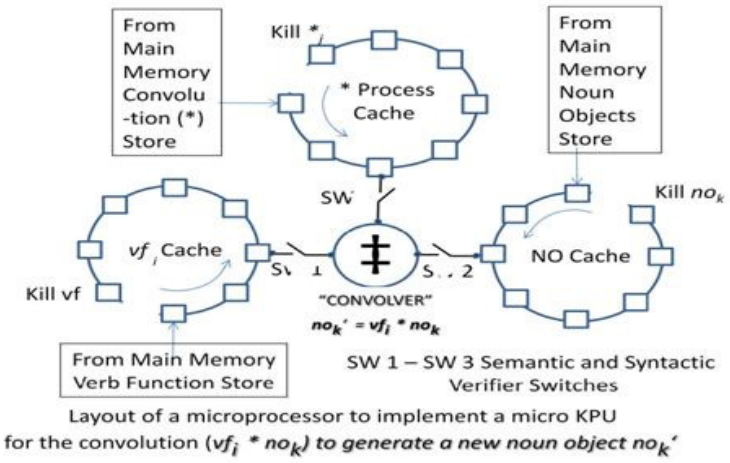

Figure 9 : Location of a "CONVOLVER" in a μKPU. The "convolver" forces the completion of a micro knowledge operation (μkopc) on one or a series of possible no's.

The coordinated execution of the instruction (*MPF ⊛ MNO*) corresponds to the nano- or a micro- medical step in the curing or healing of the patient (pat.). Largely, the doctors (MDs) and medical staff being intelligent and trained perform these long strings of micro-medical functions quite adeptly. However, the medical machine can be and needs to be programmed to perform these procedures efficiently, dependently and securely subject to the available resources (Res.). Accordingly, the flow chart for simplest medical computers, depicted in Figure 12, needs to be enhanced to reflect the additional constraints (added security, privacy, preferences, special needs and features, etc.) necessary in the medical field [25].

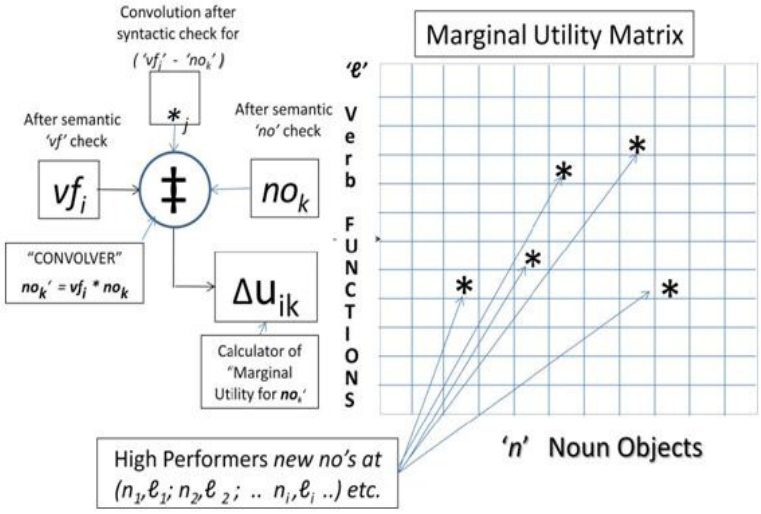

Figure 10 : The plot identifies the combination of vfi ,⊛j , and nok yielded the highest utility from convolving various VFs, ⊛s and NO's available from the local knowledge and expertise. These combinations are analyzed for the utility of the highest performers in a pushdown stack that stores the top choices of new objects NO's. When the choices are limited, an exhaustive search may be performed. When the search is expanded to WWW knowledge bases, as shown in Figure 12, the search algorithms are made intelligent and self-learning to reach the best solution(s).

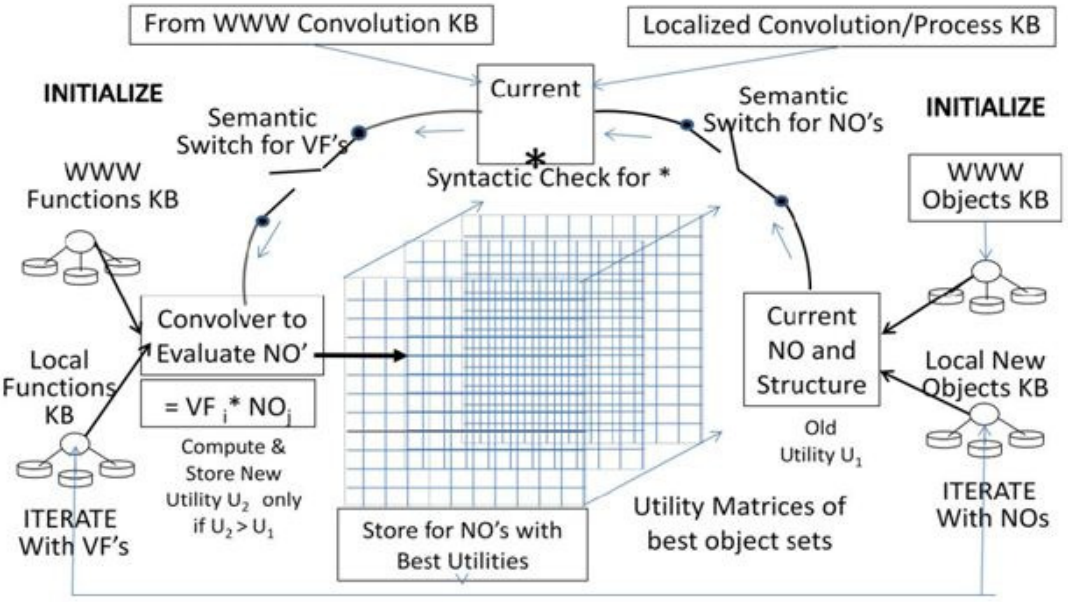

Figure 11 : Verb Functions and Noun objects are drawn from both local knowledge expertise and KBs, and verified against the Internet KB to ascertain the utilities derived from the newly synthesized noun objects. The marginal enhancement of utility is tallied against the marginal cost to make the most valid economic choice for the newly generated object NO'. The derived knowledge centric object (KCO) is likely to be a flawless and perfected KCO based on the constraints imposed on the choices of VFs, ⍟s and NOs.

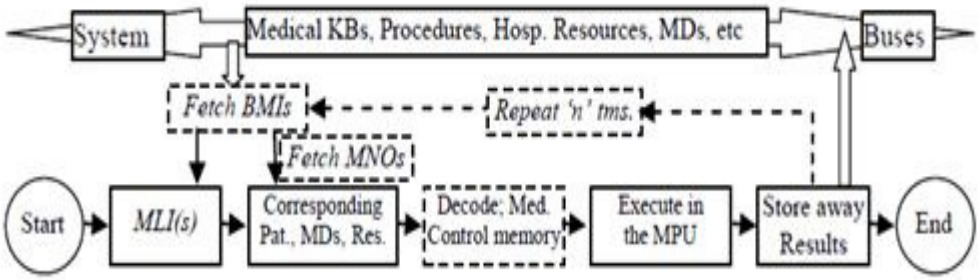

Figure 12 : Flow Chart of the simplest medical computer.

The training of the medical staff is based on medical science, device technology and accurate implementation. The rigorous training infused with intelligence and great skill leads to one sideof aspect, dependability, and predictability of the fetch, decode and execute (FDE) cycle in theMPU[5]. The entire process becomes a small part of the overall process of treating the patients.

The architecture of the central processing unit (CPU) of any moderate speed computer with enhanced memory capabilities can be forced to process the medical procedures for patients at a (very) rudimentary level. Enhanced memories will hold executable programs for the micro-procedures assembled onto the standard procedure for the patient in the special hospital, medical center, or even the country in which the procedure is going to be performed. These sub-procedures will be intelligently assembled based upon knowledge bases that use artificial intelligence (AI). For this reason, we expect the high-level language software (the compiler-assembler-loader-linker software chain [6, 7]) of the IMS to be significantly different from that of traditional computer systems. We also foresee that adapting the set of assembly level procedural instructions will be tailored to the specific medical facility, thus calling for facilities-dependent assemblers. This type of adaptation is routinely made for computer systems at installation time, depending upon the hardware in that particular system.

Dual processors and elaborate error checking offer dependability and security of data. The error checks not only verify the processing, but also the validity of the results based upon the AI-based, expected outcome. Any unexpected findings are referred to the physician team on duty. Results of procedures will be entered into a patient database. Opinions and subjective comments will be entered onto a voice-activated message retrieval system. Pictures, X-ray, and CAT scans will be entered onto a visual database of the computer system, making the IMS a truly multimedia system.

6. CONCLUSION

Processor revolution has been an ongoing and continuous process over the last four decades. The explosion in the types and capacity of the newer and specialized processors is enhanced by the need of faster, more compact and complex applications in the gaming, medical, movie, and other object-oriented applications industry. In this article, we have presented the dramatic changes that have accrued in the supply of processor chips from the Intel 1971, 4004 processor chip to the 2014, fifth generation core processor with Iris® Pro (Broadwell) chip sets. The potential and the emerging applications in the human and social domain are also presented. Two new concepts the designs for HW based object oriented processing and for knowledge processing become essential to build intelligent computer systems to address human, social and medical needs within the society. These designs presented in this article are extensions of the architecture of HW rather than an additional layer of software, object-ware, or knowledgeware to support the humanist aspects in computing. In reality, such HW based system will perform faster than layered software systems. The concept of the processor power should be measured as KNOPS (knowledge operations per second required for the Next generation of processors) rather than FLOPS (floating-point operations per second). It is our contention that GFLOPS and TFLOPS machines and super computers

[6]The processes within FDE cycle occurs within the CPU hardware environment repeatedly [8] during the execution of most computer programs. The Fetch function brings the next executable instruction from the Memory CPU via the switched Buses in the computer system. The Decode function takes the instruction part of the instruction to the Instruction Register or IR where it is decoded as a series of control pulses in the CPU to trigger the hardware operations on the Operand(s). This phase constitutes the Execute part of the FDE cycle. The resulting operands are left in the CPU registers or stored in specific memory locations. The FDE cycles thus repeats for every executable instruction in the program. (Pat.) The individual traits of the doctor and the doctor-patient relationship add different variations of care, concern, and individuality to the entire process of treating patients.

should be tamed to operate as MKNOPS humanist machines that serve society and public [23].

In this article, we have also introduced the concept of microprogramming knowledge, sensor networks, and medical based operations. Fragmented Verb Functions (μVFs)are introduced in the conceptual framework for the basis and design of micro knowledge processors (μKPUs) and micro medical processors (μMPUs). The philosophy is borrowed from the design of RISC computers and micro-programmable CPUs or (μCPUs). Such machines will bring the design and programming of knowledgeand medical machines in the domain of software and knowledgeware engineering.

The design of medical machines thus enters the domain of medical-ware to drive the MPUs and the μMPUs. Fragmentation of Medical Noun Objects (MNOs) is performed by Internet search of the morphemes objects that surround the main medical noun object (such as a patient, an organ, or an ailment, etc). Fragmentation of Knowledge Noun Objects (KNOs) is performed by Internet search of the morphemes (such as institutions, universities, or shrines, etc) that surround the main Knowledge Noun Object (such as a discipline, a scholar, or an expert, etc). Fragmentation of Medical Noun Objects is equally feasible by searching for the basic fragments of biology and physiology that constitute any medical object. When micro-procedures are convolved with micro medical noun objects, a micro medical process is invoked. Multiplicities of such processes occur any medical events/function (such as falling sick, healing, surgery, recovery, etc.). The analogy can be extended in the knowledge domain. The basic core HW remains essentially the same, but the layers of knowledge-ware and medical-ware are adjusted to suit the environment.

REFERENCES

[1]. A. H. Maslow, "A theory of human motivation", Psychol. Rev. 50, pp 370–396, 1943, see also, A. H. Maslow, Motivation and Personality, Harper & Row, New York, 1970, and A. Maslow, Farther Reaches of Human Nature, Viking Press, New York, 1971.Maslow

[2]. S. V. Ahamed, "An enhanced need pyramid for the information age human being", Proceedings of the Fifth Hawaii International Conference, Fifth International Conference on Business, Hawaii, May 26–29, 2005, see also, "An enhanced need pyramid of the information age human being," paper presented at the International Society of Political Psychology, (ISSP) 2005 Scientific Meeting, Toronto, July 3-6, 2005.

[3]. H. S. Stone et al., Introduction to Computer Architecture, Computer Science Series, Science Research Associates, New York, 1980. See also, J. P. Hayes, Computer Architecture and Organization, 2 ed., McGraw Hill, New York, 1988.

[4]. S. V. Ahamed, Chapter 2, in Computational Framework for Knowledge: Integrated Behavior of Machines, John Wiley and Sons, Hoboken, New Jersey 2009.

[5]. J. von Neumann, First Draft of a Report on the EDVAC, Contract No., W-670-ORD 4926, Between United States Army Ordnance Department and University of Pennsylvania, Moore School of Electrical Engineering, June 30, 1945. Also see A. W. Burks, H. H. Goldstine, and J. von Neumann, U. S. Army Report Ordnance Department, 1946. See also

[6]. G. Estrin, "The Electronic Computer at the Institute of Advanced Studies, Mathematical Tables and Other Aids to Computation", Vol. 7, IAS, Princeton, NJ, pp. 108–114, 1953

[7]. A. Clements, The Principles of Computer Hardware, Oxford University Press USA, 2006.

[8]. R. C. Detmer, Introduction to 80x86 Assembly Level Language and Computer Architecture, Jones Bartlett Publishers, 2001. Also see, W.G. Rudd, Assembly Level Programming and the IBM 360 and 370 Computers, Prentice Hall 1976.

[9]. Ennaji, R.; Boulmalf, M., "Routing in wireless sensor networks," Multimedia Computing and Systems, 2009. ICMCS '09. International Conference on , vol., no., pp.495,500, 2-4 April 2009, doi: 10.1109/MMCS.2009.5256646

[10]. Konrad Lorincz, David J. Malan, and at al. “FIRST RESPONSE Sensor Networks for Emergency Response: Challenges and Opportunities”; PERVASIVE computing Published by the IEEE CS and IEEE ComSoc Web accessed on November 1, 2014

[11]. Ahamed, Syed and Rahman, Syed (Shawon); "Architecture and Design of Medical Processor Units for Medical Networks"; International journal of Computer Networks & Communications (IJCNC), Vol.2, No.6, November 2010

[12]. Todd, S., Margie and Rahman, Syed (Shawon);“ Complete Network Security Protection for SME’s Within Limited Resources”; International Journal of Computer Networks & Communications (IJCNC), ISSN : 0974 – 9322 [Online] ; 0975- 2293 [Print])

[13]. Rahman, Syed (Shawon) and Ahamed, Syed; "Intelligent Network Applications for Medical Systems"; The Second International Workshop on Ubiquitous Computing (UbiC-2010),(In conjunction with WiMo 2010) - June 26- 28, 2010, Ankara, Turkey

[14]. Loukaka, Alain and Rahman, Shawon; “Discovering New Cyber Protection Approaches From a Security Professional Prospective”; International Journal of Computer Networks & Communications (IJCNC) Vol.9, No.4, July 2017

[15]. Rader, A., Marc and Rahman, Syed (Shawon); “Exploring Historical and Emerging Phishing Techniques and Mitigating the Associated Security Risks”; International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.4, July 2013

[16]. Opala, John, Omondi and Rahman, Syed (Shawon);“Corporate Role in Protecting Consumers from the Risk of Identify theft ”; International Journal of Computer Networks & Communications (IJCNC), Vol.5, No.5, September 2013

[17]. Todd, S., Margie and Rahman, Syed (Shawon);“ Complete Network Security Protection for SME’s Within Limited Resources”; International Journal of Computer Networks & Communications (IJCNC), Vol.5, No.6, November 2013

[18]. DeVoe, Charles and Rahman, Syed (Shawon); “Incident Response Plan for Small to Medium Sized Hospital”; International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.2, March 2013

[19]. Rahman, Syed (Shawon) and Lackey, Robert; “E-Commerce Systems Security for Small Businesses”; International Journal of Network Security & Its Applications (IJNSA), Vol.5, No.2, March 2013

[20]. Henderson, James and Rahman, Syed (Shawon); " Working Virtually and Challenges that must be overcome in today’s Economic Downturn"; International Journal of Managing Information Technology (IJMIT); ISSN : 0975-5586 (Online) ;0975-5926 (Print)

[21]. Dreelin, S., Gregory and Rahman, Syed (Shawon);“ Enterprise Security Risk Plan for Small Business”; International Journal of Computer Networks & Communications (IJCNC), ISSN : 0974 – 9322 [Online] ; 0975- 2293 [Print])

[22]. Donahue, Kimmarie and Rahman, Syed (Shawon); “Healthcare IT: Is your Information at Risk?”; International Journal of Network Security & Its Applications (IJNSA), Vol.4, No.5, September 2012

[23]. Neal, David and Rahman, Syed (Shawon); “Video Surveillance in the Cloud?”; The International Journal of Cryptography and Information Security (IJCIS), Vol.2, No.3, September 2012

[24]. Lai, Robert and Rahman, Syed (Shawon); “Analytic of China Cyberattack”; The International Journal of Multimedia & Its Applications (IJMA), June 2012, Volume 4, Number 3

[25]. Halton, Michael and Rahman, Syed (Shawon); "The Top 10 Best Cloud-Security Practices in Next-Generation Networking"; International Journal of Communication Networks and Distributed Systems (IJCNDS); Special Issue on: "Recent Advances in Next-Generation and Resource-Constrained Converged Networks", Vol. 8, Nos. ½, 2012, Pages:70-84

[26]. Dees, Kyle and Rahman, Syed (Shawon);“Enhancing Infrastructure Security in Real Estate”; International Journal of Computer Networks & Communications (IJCNC), Vol.3, No.6, November 2011

[27]. Hood, David and Rahman, Syed (Shawon);“IT Security Plan for Flight Simulation Program”; International Journal of Computer Science, Engineering and Applications (IJCSEA), Vol.1, No.5, October 2011

[28]. Schuett, Maria and Rahman, Syed (Shawon); “Information Security Synthesis in Online Universities”; International Journal of Network Security & Its Applications (IJNSA), Vol.3, No.5, Sep 2011

[29]. Jungck, Kathleen and Rahman, Syed (Shawon); " Cloud Computing Avoids Downfall of Application Service Providers"; International Journal of Information Technology Convergence and services (IJITCS), Vol.1, No.3, June 2011

[30]. Slaughter, Jason and Rahman, Syed (Shawon); " Information Security Plan for Flight Simulator Applications"; International Journal of Computer Science & Information Technology (IJCSIT), Vol. 3, No 3, June 2011

[31].Bisong, Anthony and Rahman, Syed (Shawon); "An Overview of the Security Concerns in Enterprise Cloud Computing "; International journal of Network Security & Its Applications (IJNSA), Vol.3, No.1, January 2011

[32].Rahman, Syed (Shawon) and Donahue, Shannon; “Convergence of Corporate and Information Security”; International Journal of Computer Science and Information Security (IJCSIS), Vol. 7, No. 1, 2010

[33].Rahman, Syed (Shawon) and Ahamed, Syed; "Intelligent Network Applications for Medical Systems"; The Second International Workshop on Ubiquitous Computing (UbiC-2010),(In conjunction with WiMo 2010) - June 26- 28, 2010, Ankara, Turkey

[34].Ahamed, Syed and Rahman, Syed(Shawon); "Design Constructs of a Knowledge Machine"; 2010 IEEE International Conference on Systems, Man, and Cybernetics (SMC2010); 10-13, October 2010, Istanbul, TURKEY

[35].Mullikin, Arwen and Rahman, Syed (Shawon); "The Ethical Dilemma of the USA Government Wiretapping"; International Journal of Managing Information Technology (IJMIT); Vol.2, No.4, November 2010

[36].Benson, Karen and Rahman, Syed (Shawon); "Security Risks in Mechanical Engineering Industries", International Journal of Computer Science and Engineering Survey (IJCSES), Vol.2, No.3, August 2011

AUTHORS’ BIO

Dr. Shawon Rahman : Dr. Shawon S. M. Rahman is an Associate Professor in the Department of Computer Science and Engineering at the University of Hawaii-Hilo, USA. Dr. Rahman’s research interests includeInformation Assurance and Security, Software Engineering e ducation, Software Testing & QA, Cloud Computing, Mobile Application Development, and Web Accessibility. He has earned his Ph.D. in Software Engineering and MS in Computer Science from the North Dakota State University, Fargo, ND. He has published over 100 peer-reviewed articles and is a member of many professional organizations including IEEE, ACM, ASEE, ASQ, ISACA, and UPE.

Dr. Syed V. Ahamed : Dr. Syed V. Ahamed is a Professor Emeritus at the City University of New York Staten Island, New York, NJ. Dr. Ahamed obtained his Ph.D. from Victoria University of Manchester in the United Kingdom. Dr. Ahamed also holds the Doctor of Science (D.Sc.) degree from Victoria University of Manchester as well as an M.B.A. in economics from New York University. Dr. Ahamed, a Fellow of IEEE, has a broad spectrum of teaching and research interests including, Intelligent Sensing Systems, Computer Architecture, CPU and KPU Designs, Telecommunications,Intelligent Networks, Knowledge Systems, Management Information Systems, and Economics. He has received many European and US patents and published many books and hundreds of articles.