Software Cell

Hugh Ching October 1990

Biological And Software Cells: **A Logical** Accident Of **Similarities**

Software is fully documented. Without documentation most of today's software can no longer function. The human baby is born without any documentation, yet it can function quite well. The main difference between software and a living organism is in the degree of automation. The "software cell" is based on a new concept in software automation. It is originally *known* as the Self-generating Software System (SSS) which, like the human baby, can operate without any documentation; yet, like all software, is always fully (but not necessarily well) documented.

The fact that the software cell, which is derived artificially and independent of any knowledge in biology, and the biological cell, which is created by natural forces, have nearly identical logic is the main motivation for this proposed research. The two cells show so many similarities (See Table) that it can be speculated that (1) the biological cell might be used to guide the future development of the software cell (In fact, the biological cell could have provided a guide for the discovery of the Self-generating Software System, or vice versa.) and (2) in turn, being fully documented, the software cell might fill in the missing links in the logic of the biological cell.

The Self-generating Software System is a computing environment in which all the programs are generated programs, in contrast to most current software programs which are written by programmers. All the programs in SSS are generated by the initial program generator which must be able to generate itself in order to have a totally automated software environment.

Software is the bottleneck in computing. There are two major difficulties in software, namely (1) communication between programmers or between the user and the computer (inadequate documentation) and (2) communication between programs of different versions or of different operating systems. Before the discovery of SSS, the software difficulties stated above are not resolved. SSS, in essence, has solved the problem of software.

The Self-generating Software System is designed to separate users (problem specification) from computer technology. SSS produces software by feeding problem specifications, which are produced by answering questions in human language, into program generators, which are in computer language. The forms of the specifications are designed so that they are understandable to the computer. In SSS even the program

generators have their problem specifications; the generators can generate themselves. When all the programs have problem specifications, users will be dealing only with the problem specifications and not with the inside codes of the programs. The major advantages of SSS are that it can self-improve, automatically document and update. SSSshould eventually eliminate all high-level computer languages and replace them with the human language of the user.

The Self-generating Software System is by itself a breakthrough in knowledge, because software will become the dominant form of knowledge when knowledge experts without the knowledge of computer programming can interact easily with the computer. Software automates knowledge. Software automation automates software. Finally SSS automates software completely.

There is one fundamental aspect commom to both the software cell and the biological cell. Both cells are among the most complex systems in software and in biology respectively. The software cell, like the biological cell, includes all the machine instructions (amino acids) and their generating and utility programs (protein), and all the problem specifications (DNA). Thus, instead of building complex system from simple components, as is done in most current software systems, the software cell builds complex system from an equally complex system, the software cell, just as is the case in biology. SSS currently is devoted mostly to execution. The cell spends mostly on overhead.

The software cell might open up a new field of technology based on the concept of the cell. Most of our current technology can be considered non-cell-based technology which is characterized by starting from the simple and building into the complex. The human is made of cells, and the cell contains all the information of the human. The technology based on the cell is from the complex to the complex, which may have the capability of reproducing the complex. In SSS each program can be a software cell; a wordprocessor, a spreadsheet, or a databased management program can regenerate itself in a form suitable to the user because it contains all the capability of the cell.

The new technology based on the concept of the cell is already blessed by the biological cell which can act as its guiding star. It should provide alternative approaches and new impetuses to the research in artificial intelligence, artificial life, software automation and robotics, which is software and hardware automation and should be one of our ultimate goals.

Research Plan: A One-To-One **Mapping** Of The Two Cells

The basic operation of the cell involves selecting the messages in DNA (specifications) and sending them through a protein generator(program generator) to produce protein(program). Ideally, direct copying should not be allowed in the generation processes. However, DNA can only replicate semi-conservatively, and the software cell has the same problem of not being able to generate with complete flexibility; somewhere partial copying is done. The codes (DNA and specifications) for generating the generator can be expected to be inflexible in both cells.

The program generator and the generator of the program generator in SSS seem to correspond respectively to the ribosomeand the nucleolus in the biological cell. The nucleolus appears to generate itself along with the DNA through replication during mitosis. To test the software cell, the initial phase of the proposed research will be devoted to make a detailed correspondence between the two generators in the two cells. SSS can be transformed into self-reproducing primitive bacteria (to study the origin of life ?). A point-by-point check of the corresponding logic of the two cells can be attempted; it is inconceivable that the biological cell can escape its parallel logic in the software cell, if there is an item-by-item match between the two cells. (SSS has already been demonstrated and a commercial version has been designed and under development.).

Investigators

Hugh Ching (S.B., M.S., Sc.D. MIT, Postdoctor Courant Inst. of Mathematical Sciences, Research Associate UC Berkeley. Generalized Fluid Description, Quantitative Theory of Value, SSS)

Some biologists and engineers from UCB will participate.

Professors Dwight Egbert and Sami Fadali of University of Nevada, Reno have proposed to develop a Self-generating Singleuser Disk Operating System, an MSDOS clone, based on SSS.

Nature Packs All Its Valuables Into A Speck Of DNA

Multicelled organisms are necessary to study artificial intelligence, which leads to many interesting and pertinent questions. Is the human being a biological computer ? Is life an illusion created out of complexity - an exhibition of nature's subtlety (and vivacity) ? Will the distinctions between living and non-living substances become unnecessary in the face of new theories on life, as the theroy of evolution

erases the differences between the origins of species ? The investigation into the perception of pain will help us understand the origin of value. Most importantly, we want to find out for the following reason whether improvements on the biological cell can be made.

It appears that intellect originates from the human brain. However, the brain, as most living things, is developed from the cell. Thus, the cell is the source of intellect. Strictly speaking, as soon as the cell comes into existence, intelligence derivable from the cell and all the problems solvable by this intelligence should be considered already a part of our destiny; a new order can be established only when intelligence passes a certain threshold such that evolution can be replaced by design.

Ultimately, only fundamental improvements on the cell can be considered real progress. It would not be too far fetched to speculate that when civilization reaches a steady state (when we know enough about our world), it will be realized that the rational purpose of existence, or of life, is to improve on the cell - a continual process of nature to make itself more intelligent and beautiful, or simply to increase its own value.

Table: Corresponding Items for Biological Cell and Software Cell

Biological Cell	Software Cell (? = Uncertain)
Amino Acid	Instruction; Programs (Protein) are made of instructions (Amino Acid). Analysis (Decoding cell) Synthesis (Coding software cell) AT TA CG GC (base of 4) 0 and 1 (Binary: base of 2) ATP Allowed computer time or memory
Cell	Self-genarating Software System (All Specification Files + All Generators + All Programs + Allowed computer time or memory)
Chromosomes	All Specification Files
Codon (64 = 4^3)	6-bits byte (8 bits-parity-RNA/DNA check)
DNA	Specification File (which have to be retrieved into program for execution) Enzyme Program (utility)
Evolution (Undocumented)	Design of artificial life (Fully documented) Gene Specification File (partial)
Homeobox	Specification File for regulating other specifications

Human (?)	A highly complex cell-based computer
Life (?) cell (?)	Artificial Life based on software
Meiosis (?)	Splitting of software cell (?)
Membrane in the directory of a software cell	Program checking for allowable files
Mitosis (?) program generator; semiconservative Self-generation	Feeding all specifications through
Nucleus Generators + Programs for regulating allowable files	Main Specification Files + Main
Nucleolus (?) generator (rRNA)	Generator of program
Pain (?) feedbacks and interpretations (?); Question to **ask is:** How to make a machines feel pain ?	Error Messages with complex
Protein specifications	Program (for applications)
Replication	Semi-conservative generation of
Reproduction	Self-generation
Ribosome (rRNA) (specification+program)	Program generator
mRNA, tRNA one set)	Specifications in program (tRNA =
Transcription	Generation of specifications (RNA)
Translation polypeptide)	Generation of program (protein or
Virus specifications Not computer virus (which can only make direct, instead of semi-conservative, copies)	Incomplete software cell with

****

Software Cell	**Biological Cell ----------**
Generator	Cell, Nucleus, Nucleolus, Ribosome
Instruction	Amino Acid
Program directly executable)	Enzyme, Membrane, Protein, Virus Specification File (not
Homeobox Specifications	Chromosomes, DNA, Gene, RNA

Relating SSS to DNA (2000 Update)

Project Start (Confidential prior to 1/31/2000 except to members Ram, Kunii, Tsai, Wang, Debra, Hugh)

Our associative memory enables us to access almost all the information in our memory. The associative memory is a high-level intelligence (machine logic or arithmetic are examples of low-level intelligence) and, therefore, should not be used in set theory as the foundation of mathematics. The foundation of mathematics should probably be rad theory (rad=radix or base of number system), which deals with integers, because machines without associative memory prefers the manipulation of integer numbers over the identification of objects. SSS (Self-generating Software System) uses the above two distinct preferences by humans and by machines in the design of an Universal User Interface (UUI) for human and machine communication. UUI is simply the familiar tree-structured, numerical multiplechoice question format seen most often in computerized library search systems. The universal nature of UUI indicates that UUI can be used to replace any user interface. The numerical integer choices in UUI are for the machine to handle, and the graphics or the human language is for the humans to comprehend. As a practical example of the application of UUI, the dominant Microsoft Windows user interface is overly friendly to the user because it has not included the numerical choices with the items and, thus, has completely ignored the participation of the machine in the human-computer interaction.

In UUI the numerical key inputs are, if needed, recorded in a program "speciiication file" to be easily manipulated, namely, auto-updated, auto-documented, and self-generated by the computer. Selfgeneration will allow the self-generated software system to completely sever its tie to past technology. For example, the first question of UUI of a program generator is:
Choose (1) Input, (2) Output, (3) Calculate, (4) Print, (5) Others ?
If we choose, say, 4, then the UUI will ask the next question:
Choose (1) Print a number, (2) Print a string, (3) Print a variable,
(4) Others ? If we choose 2, the UUI will ask:
What is the string ?
After we input the string to be printed, the generator will guide the program execution to a GO"1'O statement or call label statement. The GOTO statement number will jump the execution to the address given by the statement number. A Permanent Record System (PRS) is designed to eliminate the need to remember the statement number, which conceptually can be used to represent any technical iniormation. PRS is invented to eliminate from computer usage all technical barriers, which, by the way, include foreign languages.

A PRS record has the general format consisting of three items:
statement number ADDRESS = statement number the record and an actual record in BASIC
5486483 ADDRESS=5486483:INSTRUC"1'ION$="PRINl„'+S"I'RINUs
where the statement number is 5486483 and the record is INSTRUCTION$-
'PRIN"I"'+STRING$. The repetition of ADDRESS=5486483 is to let the program know, when needed, the address 5486483 is the generating statement for printing a string. Thus, using the same associative memory, PRS has made it unnecessary to remember the technical information represented by the record at the address 5486483. For example, when we want to write the statement GOT0 5486483 in the self-generation of the SSS generator, we no longer need to commit to memory the statement number 5486483.

To relate SSS to DNA, we set in the PRS record:
statement number = ADDRESS = the record

The ADDRESS of software corresponds to the lock and key system in molecular biology. To be continued

A Demonstration Self-generating BASIC System (DSBS) has already been completed as a guide for the commercial version of SBS (20% of which has also been completely by June 1990). DSBS is presented in full in the report Software **Automation Demonstration** of Section 3 of this Business Plan. The following is a printout of the actual first few screen displays of DSBS:

```
            Demonstration Self-generating BASIC System (1990)

Anyone over 9th grade can write programs.
You can write a program by answering questions.

What is the name of your program? (Type RETURN key after you
finish.) DSBS

Choose one of the numbers in the next question:
***#¡*#ƒ***~**~*****ƒ**~*~****~~ƒ*~ţ**********#**~**#ƒ**#****#*#**~#*#*ƒ*#~*
***
*(1) Print;(2)Question;(3)Compute &
Logic;(4)Files;(5)Exit;BS;Ins;De1;A;L;J;H ? 1

     ( Answer 1    Statement no= 1010 )
(1)Print a statement on Screen; (2) To Printer. (3) Print
statement=variable on Screen; (4) To Printer. (5) Clear Screen; (6)
Others ? 5

     ( Answer 2    Statement no= 1020 )

*(1) Print;(2)Question;(3)Compute &
Logic;(4)Files;(5)Exit;BS;Ins;De1;A;L;J;H ? 1

     ( Answer 3    Statement no= 1030 )
(1)Print a statement on Screen; (2) To Printer. (3) Print
statement=variable on Screen; (4) To Printer. (5) Clear Screen; (6)
Others ? 1

     ( Answer b    Statement no= 1040 )
Type in the statement to be printed out (Type RETURN key after you
          finish.) Demonstration Self-generating BASIC System


*(1) Print;(2)Question;(3)Compute &
Logic;(4)Files;(5)Exit;BS;Ins;De1;A;L;J;H ? 1

     ( Answer 9    Statement no= 1090 )
(1)Print a statement on Screen; (2) To Printer. (3) Print
statement=variable on Screen; (4) To Printer. (5) Clear Screen; (6) Others ?
1

     ( Answer 10    Statement no= 1100 )
```

```
Type in the statement to be printed out (Type RETURN key after you
finish.) Anyone over 6th grade can write programs using S/A.
```

DSBS is a self-generating program generating; it can be used to demonstrate the self-generating capability of SSS. The following is the first few screen output of a generated program of DSBS trying to self-generate DSBS:

```
          Demonstration Self-generating BASIC System

Anyone over 6th grade can write programs
using S/A. You can write programs by
answering questions. This is a GENERATED
PROGRAM !

What is the name of the program to be generated ? (Type RETURN after
finishing) APPPRO
Choose one of the numbers in the following question:

(1) Print; (2) Question; (3) Compute & Logic; (4) Files; (5)
Others;BS;Ins;Del? 1
(1) Print a statement on Screen; (2) To Printer. (3) Print
statement=variable on Screen; (4) Statement=variable to Printer. (5)
Clear Screen. (6) Others ? 5

(1) Print; (2) Question; (3) Compute & Logic; (4) Files; (5)
Others;BS;Ins;Del?
1
(1) Print a statement on Screen; (2) To Printer. (3) Print
statement=variable
on Screen; (4) Statement=variable to Printer. (5) Clear Screen. (6)
Others ?
1
Type in the statement you want to be printed out?
This is an application program generated by a GENERATED program generator.
```

The above generated program is capable of generatan.0the following application program (in full):

```
This is an application program generated by a GENERATED program What is the
value of the first variable ?
5
What is the value of the second variable ?
7
```

```
The third variable is the product of the The value of the third variable
is 35 What program do you want to go to ? DSBS

                                generator.

                                first and the

                               second variables.
```

SSS can be easily self-modified by users; SSS can be continually self-improved. One important example of the application of this self-modification feature of SSS can be demonstrated using DSBS to translate itself into German (or any other languages - computer or human). The following is the same screen printout of the previous page of a generated program imitating DSBS, but it is in German:

```
         Beweiss Selbst-erzeugen

BASIC System Dieser ist ein ERZEUGT

PROGRAM !


Was ist das name von seinen programm ?
APPPRO
Wahlen ein zahl in das nachst frage:
######################################################################
####### (1) Drucken; (2) Frage; (3) Rechnen & Logik; (4) Akte; (5)
Ander; BS;Ins;Del ? 1
(1) Drucken Schirm erklarung; (2) Druck; (3) Drucken Schirm
erklarung=unbekannt; (4) Druck; (5) Klaren Schirm; (6) Ander ?
5
;#####################################################################
##
(1) Drucken; (2) Frage; (3) Rechnen ~ Logik; (4) Akte; (5) Ander;
BS;Ins;Del ? 1
(1) Drucken Schirm erkle.rung; (2) Druck; (3) Drucken Schirm
erklarung=unbekannt; (4) Druck; (5) Klaren Schirm; (6) Ander ?
1
Was ist seiner erklarung sein wollen druckt aus ?
This is an application program generated by a GENERATED program
generator.
######################################################################
####### {1) Drucken; (2) Frage; (3) Rechnen & Logik; (4) Akte; (5)
Ander; BS;Ins;Del ?
```

(1) Drucken Schirm erklarung; (2) Druck; (3) Drucken Schirm
erklarung=unbekannt; (4) Druck; (5) Klaren Schirm; (6) Ander ?
1
Was ist seiner erklarung sein wollen druckt aus ?
##
####### (1) Drucken; (2) Frage; (3) Rechnen & Logik; (4) Akte; (5)
Ander; BS;Ins;Del ?