Why do Large Language Models lack Transparency? -

and a reassessment of the archives of an early transparent human/computer research project

Christopher F Reynolds
chris@codil.co.uk
Draft Discussion Note, December 2023

Abstract

The latest large language models are very impressive but they are they are black boxes which don't really understand the information they are processing. For comparison this paper reviews the archives of the CODIL project (1966-1988). CODIL was a transparent language system modelling human thought processes, but which was abandoned because it was not compatible with the AI paradigms that were fashionable at the time. In addition to demonstrating how to constrict a transparent human-friendly "electronic clerk" the archives also provide information about how research was handled at the time of the ICL merger, and the problems which neurodiverse researchers can face when their "blue sky" research differs from the establishment views. It is possible that, if the merger to form ICL hadn't happened, CODIL, or something like it, could have formed the basis of a human-friendly computer language.

The Problem with the latest AI Systems

Computing and artificial intelligence has made significant leaps forward since Bush[1] published his 1946 essay "***As We may Think***" introducing the idea of the memex. Over the years AI research has been through several overhyped paradigms separated by AI winters[2] when the predicted goals have not been reached, while there have been significant advances in online computing. There can be no doubt that the latest chat boxes using large language models, available over the internet, are very impressive but there are problems because they lack transparency and the difficulties have been recently summarised as:

> *The system … is not interpretable even to the system's developers: hundreds of billions of parameters (numbers), which do not map cleanly to human-interpretable concepts. For this reason, frontier AI systems are "black boxes" to their developers, who can observe their*

[1] As we may think. Vannevar Bush, The Atlantic, July 1945
[2] Review of AI Winters

behaviour but have little understanding of the internal mechanisms that produce them. This lack of mechanistic understanding makes it challenging to know how to change, much less how to predict, the behaviour of an AI system.

AI Safety Summit, November 2023, Capabilities and Risks from Frontier AI

This lack of transparency should not surprise anyone, because the main AI paradigm used to decide whether a computer system is intelligent is the Turing Test[3]. The aim of the test is to produce a black box system which imitates (but not necessarily accurately models) human intelligence. The latest AI research has set out to produce a powerful black box system and, having succeeded, it would appear that the resulting black box solution has real problems. Even the chat box Bing[4] seems to agree:

As the Turing test involves a "black box" is it surprising that AI systems designed to pass the Turing test also behave like black boxes?

That is an interesting question. The Turing test is a way of measuring the intelligence of a machine by comparing its responses to those of a human in a conversation. The test involves a "black box" because the interrogator does not know the internal structure or design of the machine, and only judges it by its output.

AI systems designed to pass the Turing test, such as ChatGPT, also behave like black boxes because they are based on large language models (LLMs) that generate text by predicting the next word based on statistical patterns in billions of online sentences. These models are very complex and difficult to interpret, and often produce outputs that are not consistent with logic, common sense, or facts. Therefore, even though they can mimic human behavior and fool interrogators, they do not necessarily have any understanding or reasoning behind their responses.

So, it might not be surprising that AI systems designed to pass the Turing test also behave like black boxes, since they are both evaluated by their output rather than their input or process. However, some researchers argue that this is not a good way to measure the intelligence or capabilities of AI systems, and propose alternative tests that require more abstract thinking, problem-solving, and explanation skills. These tests might reveal the limitations and blind spots of LLMs, and also help us to understand how they work and how to improve them.

A possible reason for this difficulty is that powerful predictive mathematical models do not always reflect the way the human brain's information processing system works. The classical example, which shows how misleading powerful mathematical models can be, relates to the movement of the planets as seen from Earth. Over two thousand years ago the Greeks[5] explained the observations in term of an epicycle model, and it is now known, thanks to a fortunate archaeological discovery, that they constructed a clockwork "computer"[6] which could predict the planets' positions with acceptable accuracy. Of course we currently understand how and why the planets really move, thanks to the research of Galileo, Newton and Einstein, and it is also recognised that the epicycle model can be extended to explain any orbit simply by increasing the number of epicycles until you get a good fit[7]. A more recent example relates to the sophisticated quantum mechanical calculation of the energy levels in a hydrogen atom which really tell us nothing about what an electron is. Have the current large language models fallen into a similar trap by using a predictive statistical model which involves billions of variables, but which tells us very little about how the human brain works?

---

[3] See Review of the Turing Test

[4] Bing Chat with GPT-4

[5] See Ancient Greek Astronomy

6 See Antikythera mechanism

[7] It is even possible to use epicycles to trace out the outline of a Tyranosaurus rex. Effectively the Greeks were uspng an algorithm which, given enough parameters, will be able to predict any orbit, however unlikely, with no understanding of the underlying physical reality,

Do we really need an AI system that models the brain? The CODIL Project

Perhaps a better approach would be to try and model how the human brain handles complex real-world information. I am currently investigating the surviving documentation of an early human/computer interaction project called CODIL (Context Dependent Information Language)[8] which directly addressed this problem, but which was abandoned over 30 years ago because it did not conform to the then popular (but now no longer fashionable) AI paradigms. CODIL's chief assumption was that if you wanted intelligent interaction between humans and computers the key requirement was two-way transparency and mutual understanding of the information being processed. The research deliberately excluded the use of powerful mathematical (and computationally expensive) algorithms which most potential human users would not understand and which were too sophisticated to have arisen in the human brain through any likely evolutionary routes. There was also no need to take the Turing test approach and hide the fact that the "electronic clerk" was a computer, any more than an accountant in a management team of humans would hide the fact that they were an accountant with access to specialised knowledge. The starting point of the research was that there must be mutual understanding of the task if the "intelligent" computer clerk was to work in partnership with humans on complex real world projects.

The purpose of this re[ort is to identify what the archives tell us about how the CODIL project tackled the human/computer interactive transparency problem.

The Background to the CODIL project.

The CODIL project was almost an accident which arose because I am neurodivergent (autism and aphantasia) – which means that I sometimes come up with unconventional solutions to complex peoblems. Between 1959 and 1971 I was involved with the following very different complex real world tasks:

- A theoretical study (no computers then) of the properties of a group of aromatic chemicals which linked quantum mechanical models (involving the Heisenberg uncertainty principle) with experimental results. (Ph.D. at Exeter University) One problem was communication difficulties between theoretical and experimental chemists and as a result I decided to look further into human-human communication.
- A geomorphology project relating to the development of a river valley during the Ice Ages based on scraps of surviving accessible evidence preserved in caves in the adjacent limestone hills. (A hobby linked to what later became the William Pengelly Cave Research Trust)
- I was employed as an information scientist in a combined mail room and library department where my role was to ensure the smooth flow of research and development information in an international veterinary company (a subsidiary of the Wellcome Foundation). In effect I was employed as a human chat box, providing management information from a paper data base of correspondence and reports. I became interested in the possibility of using computers to provide better management information, but to do this I needed to change employer.

[8] The CODIL project effectively arose from a design study in 1968 and effectively finished in 1988. A final home for the extensive archives has not yet been agreed. The current reassessment is to decide what parts of the archive need to be retained or digitised.

- I moved to work on one of the most complex commercial applications implemented in the 1960s on LEO III computers within a large oil marketing company which had approaching a million customers (Shell Mex & BP). I took a particular interest in the way that most computer "errors" (apart from simple typing errors) were really the result of failures in human-human communication between the sales staff and the computer experts. A design study of the extremely complicated sales contract system formed the basis for the later CODIL project.
- I was head-hunted to do market research into the requirements for the next generation of large commercial computers, which it was hoped would support interactive integrated management information systems (English Electric LEO, later ICL). Prior to the merger, the CODIL project was started with the support of John Pinkerton and David Caminer, but it was one of a number of research projects closed to save money after the merger to form ICL.
- For a year I worked on the human interface of the massive Linesman/Mediator air defence system, where I ended up as personal assistant (unofficially trouble shooter) for the project's programming manager, before becoming Reader in Computer Science at Brunel University where I continued the CODIL research.

During this period I never stopped to define what a complex task was in the context of large organisations, but Donald Rumsfeld's later definition[9] seems appropriate:

> *Reports that say that something hasn't happened are always interesting to me, because as we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns—the ones we don't know we don't know. And if one looks throughout the history of our country and other free countries, it is the latter category that tends to be the difficult ones*

The design factors underlying CODIL

CODIL was originally proposed as an interactive language for supporting future top-of-range interactive commercial management information systems. The intention was to produce an electronic clerk which could work symbiotically with a team of humans working in a complex (i.e. not fully predictable) marketplace. The CODIL model can be considered as an extension of the memex described in Bush's classical "*As we may think*" paper[10]. The goal was tp design an electronic clerk which would process information more accurately than humans, keep accurate records and ***always be able to tell the human user what it was doing using the user's terminology***. Because the system needed to support a wide range of large commercial applications the solution also had to be cost-efficient when run on the computers likely to be available in the mid 1970s.

Bush's paper suggested a special information language might be appropriate for the memex. The problems of implementing a reliable working natural language interface for computers which might be used anywhere in the world, seemed impractical in the mid 1960s. The CODIL language was designed to be provide a simple syntax framework which used the ontology of each relevant task area (whatever the natural language spoken by the human users) and, as all tasks used the same generic framework, switching between tasks should be easy. The supporting system needed to process information in a way that the human user could understand without overloading the user's short term memory. While the system could be simulated on a conventional computer it was suggested that some hardware modifications were to allow faster associative addressing. In fact the hardware aspect was

[9] Donald Rumsfeld speaking at a U.S. Department of Defense (DoD) news briefing on February 12, 2002,
[10] As we may think. Vannevar Bush, *The Atlantic*, July 1945

never explored beyond taking out a patent and for historic reasons the original research only considered keyboard input.

CODIL was unlike conventional programming languages because it was a two-way language[11] which allowed the human to describe all aspects of the task to the computer. It also allowed the computer to describe what it was doing to the human, and allowed the computer to request further information when necessary. In hardware terms CODIL resembled a symbolic assembly language where the human user could quickly interpret CODIL language statements in natural language terms. On the computer side task information was stored as named sets in an associatively addressed network[12]. Every node in the network represents a named set or a partition of a named set, and set membership defined the links between nodes. Fuzzy logic could be used to indicate the strength of the links between nodes and to accommodate learning facilities. There was no formal distinction between "program" and "data" and every node could be used as a command, a logical gateway (IF) or as data depending on the context. However if the user decided to put all the rules in one “program” set and all the “data” in another set, the set generated by matching the “program” and “data” sets would be equivalent to the results generated by a conventional procedural program/data model.

The CODIL decision-making unit (the logical equivalent of a conventional CPU) used a set of activated nodes (The FACTS) as a remarkably small working memory which modelled the human user’s short-term memory and which formed the heart of the computer/human interface. The decision-making unit uses the FACTS to explore the network of interlinked sets to "deduce" what new nodes need to be activated. Because sets can contain sets, and the route through the network changes every time a new node is activated the result is highly recursive and logically very powerful. The system was transparent because every node is meaningful to the human user (if suitably presented as text on a terminal screen) because it represents a partition of a user-named set. In particular the nodes visited form a track through the network that, in stored program computer terms, represents a tailor-made mini-program, generated in real-time for the current task. This explains how CODIL can be used for uncertain and complex tasks, as the computer generates tailor-made mini-programs (in human readable form) for the current detailed task at the time that a decision is needed. This very different to the conventional programming approach where a human programmer has to define a complex all-options-included procedural program in advance – and which results in maintenance problems when the complex real world requirement change.

CODIL as a transparent generic information processing system

CODIL was originally designed to be a language framework which would be easy for both humans and computers to understand, using a task related vocabulary chosen by the humans, and processing information in a way that the human could understand and control. The idea started with a design study involving the sales contracts in a major oil market company, selling products as varied as liquid

[11] I am unaware of any operational computer language where this was a principle design aim.

[12] In the original research CODIL was describes as a table driven system, with a “decision making unit” scanning the table. The current reassessment of the archive shows that the table was best seen as a map of a network, where each line of the table defined a node and associated links within the network. Seen in this way the “decision making unit“ decides which nodes in the network are active and formed part of both the network’s short term memory and the user interface.

petroleum gas, aviation fuel, lubricating oil and tarmac. This very simple example[13] shows how it works.

For any task information is provides in terms of sets and subsets. Thus the CODIL item **CUSTOMER = Smith & Co** identifies a member (**Smith & Co)** of the set **CUSTOMER**. When **Smith & Co** orders **100 widget** the order is recorded in a statement (= a set of linked items) such as:

CUSTOMER = Smith & Co; PRODUCT = Widget; QUANTITY = 100.

which could be one of a number of orders held in the file (a set of linked statements) **TODAYS ORDERS.**

Another set, **PRICE LIST,** contains statements including:

PRODUCT = Widget; QUANTITY >= 50; UNIT PRICE =10.

while another set, **CALCULATE**, contains statements such a

QUANTITY; UNIT PRICE; TOTAL PRICE IS = UNIT PRICE * QUANTITY

where the **QUANTITY** item refers to any member of the set **QUANTITY** and the **IS =** identifies that the arithmetic demon is needed to determine the relevant member of the set **TOTAL PRICE.**

Processing is straight forward. The statements are mapped into a network and the order is placed into CODIL's "short term memory" – which is a set called the **FACTS**. By definition all items in the **FACTS** set are true, and the CODIL decision making unit follows the "true" links through the network to deduce that the items **UNIT PRICE = 10** and T**OTAL PRICE = 1000** tshould be activated to become part of the **FACTS** set which now contains:

CUSTOMER = Smith & Co; PRODUCT = Widget; QUANTITY = 100; UNIT PRICE = 10; TOTAL PRICE = 1000.

The CODIL model differs from the conventional stored program model in the following ways:

- CODIL is ***fully self-documenting*** because all information relevant to the current task is described in the terminology used by the human user and is easily searched or updated because there is no artificial distinction between "program information" and "data information." A conventional application program is effectively a "black box:" which contains "program information" which the user cannot search or amend to meet changes in the complex real world. This means that additional human readable documentation has to be provided for the users and there could be differences between that documentation and the way the actual program works.
- In the stored program model each field represents a single value. A CODIL item can represent the whole set, the null set, a simple partition of a set (a single value, a range or a list), a set containing statements which describes the original set. or a function (called a demon), such as **IS =** , which obtains a value describing a member of the set. This can involve interactions with the human user. For example **QUANTITY (QUERY) = 10** can trigger a request to the

[13] While this is a simple example the original design study included contracts in which some customers had a discount on purchases of bulk fuel oil depending on the name of the coastal tanker used to make the delivery.

user to either confirm the default value or substitute a new value. Because of the flexibility of CODIL items the system is far better handling tasks which include missing or uncertain information or where the rules are likely to change.

- The CODIL language is mapped onto an associatively addressed network, rather than a digitally addressed array of numbers. This resembles the human brain which clearly holds information in networks rather than arrays, and where there is no clear distinction between "program" and "data"
- In CODIL the conventional task-specific application program is replaced with a ***generic (task independent) decision making unit***. The unit assumes that the **FACTS** set (the equivalent of human short term memory) is true and scans the network looking for new items to include in the **FACTS** set. At each step the unit decides if the current item is "true" in the context defined by the **FACTS**, and as a result decides which path to follow through the network.
- The actions of the decision making unit are ***transparent*** because the human user can be told the positive (i.e. "true") path through the network in the form of a mini-program, written in CODIL and tailored to the **FACTS** being processed. This compares with the stored program model where an expensively human generated application program has to be generated in advance and which has to explicitly anticipate all possible task inputs. CODIL is better at handling complex tasks because it delays generation of the "mini-program" until the time it is needed.

Summary of the original research and publications

General Introductions and the Software Interpreters

CODIL started as the symbolic assembly language of an unconventional computer processor design, proposed in 1967, where the aim was to produce a "white box" system which would work symbiotically with people in the field on non-numerical information processing applications. The hardware was never built but four different software interpreters were written which have shown that the approach was feasible, and would support a wide variety of applications. The four versions of the interpreter were as follows:

- The Pilot program: Written in symbolic assembly code to run on a System 4/70 computer (IBM 360 compatible). This was a batch system which was built to demonstrate the feasibility of the idea for a range of small scale test applications. An introduction to CODIL and information on the design is given in ***CODIL: Part 2: The CODIL language and its interpreter*** (1971).
- The ICL 1903A interpreter was written in COBOL (a mistake - but chosen because CODIL had started in the context of very large commercial data processing systems) and while it could be used via a teletype terminal most of the early work was done in batch mode. It was a complete redesign of the Pilot program, incorporating many of the lessons learnt in the previous version, but in retrospect retained some "conventional computing" features which were later shown to be inappropriate. There were no publications explicitly describing this interpreter, but some information is given in papers describing the applications tested. These included several data bases, *Tantalize* (a heuristic problem solver), and interactive ***Teach Yourself CODIL*** lessons, as well as a variety of small scale test applications. The best account of applications, etc., examined with this version is in ***A Psychologicl Approach to language design*** (1978) It was used from 1972 to 1980.
- The Multics interpreter was an interactive upgrade of the ICL version, when Brunel university switched from batch working to glass teletypes. It was used to support a variety of teaching packages and work was done on online interactive educational task. It was used from 1980 to 1988, when I retired and the Multics system was replaced.

- MicroCODIL: The limitation of glass teletype interaction of the Multics system - and the desire to develop a far more flexible user interface - led to a very significant redesign. It was decided to implement a portable version on a small educational computer, The BBC Micro, and the software was test marketed to schools. The paper ***CODIL – the architecture of an information language*** (1990), describes MicroCODIL - and also includes some features of the Multics version. MicroCODIL made it possible to escape from glass teletypes and work on windowed screens including colour. This aspect of the software design is described in detail in ***The use of colour in language systax analysis*** (1987).

CODIL in Context

Prior to the work on CODIL I had worked on the manual processing of complex research and development information in a subsidiary of the Welcome Foundation. I then worked on a very large computerised sales accounting system for Shell Mex & BP, where I got a good insight into the limitations of commercial computer systems as they were in the 1960s. Two of the early papers looked specifically on the importance of recognising that not all information was neat and tidy, or could be predicted in advance. The first was presented at Datafair 71, in Nottingham, in March 1971 and reprinted in a slightly modified form as ***CODIL: Part 1: The Importance of Flexibility*** (1971). The second was ***Designing an interactive language for the pragmatic user*** (1974). This was published when there was a strong movement in the data processing world to more and more formalism - including the introduction of relational data base techniques. This paper points in the opposite direction and argues that you need to start by designing a simple architecture which the human users can understand. Later papers, such as ***Human Factors in System Design*** (1987), discussed the design in the context of MicroCODIL.

The much later paper ***Are we trapped by our training*** (1991) looks at a different problem that emerged in the research. It appears that people who have been trained to use a conventional programming language found it difficult to unlearn the "need" to predefine an algorithm, and hence found it harder to understand CODIL than people with no computer experience.

CODIL and Artificial Intelligence

At an early stage it was realised that the CODIL approach appeared to model how some people thought about practical information processing tasks. The work was being done in a computer manufacturer with a view to developing a commercially profitable system and at the time little thought was given to what the research might say about human intelligence or computer science theory. Once the work had moved to a university environment the wide implications were considered and initial thoughts are given in ***An Evolutionary Approach to Artificial Intelligence*** (1973). It was discovered that CODIL could support a powerful heuristic problem solver, and demonstrate machine learning, and progress on this was outlined as ***Recent Developments with CODIL*** (1976). The paper ***A Psychological Approach to Language Design*** (1978) includes a summary of what had been achieved, including the incorporation of learning facilities. A very powerful problem solver called **TANTALIZE** was implemented and was used to solve 15 consecutive Tantalizer Problems (later called Enigma) from the New Scientist. However for various reasons I decided to concentrate on other possible applications of CODIL and as a result the TANTALIZE problem solver was never transferred to the Multics version of the software.

CODIL & Data Bases

CODIL has been tested on a variety of different topics as can be seen in ***CODIL as an Information Processing Language for University Use*** (1981). From an early date it was used to process patient diagnostic and treatment information in the Cardiac Department at Hillingdon Hospital – ***Using CODIL to handle poorly structured clinical information*** (1978) - but as the information was confidential another data base was set up of family history information. This was chosen not only because it was non-confidential, but also because the information described real life situations and

was often incomplete or ambiguous. Four books containing the genealogical information were produced to demonstrate the use of CODIL for historical data. The two most relevant papers are ***CODIL as a knowledge base system for handling historical information*** (1988) and ***Knowledge Bases for Historians*** (1988). A commercial software package for early personal computers, called Superfile, was a pirated version of the file handling routines in CODIL (but lacked the logic processing routines) and was widely used for archaeology and related applications in the UK in the 1980s. It was also decided to demonstrate that CODIL could handle conventional data base files, and one application was the departmental usage statistics on the university computer. The paper ***Formalism or Flexibility?*** (1978) showed how CODIL could be used to support Relational Data Base files.

CODIL - Online Teaching and Publication

In 1980 the university switched to a terminal-based service and CODIL was used by classes of up to 120 students, both to introduce them to the computer and the course, and for computer-aided instruction. At the same time I was involved in the British Library BLEND[14] project to explore the possibility of having online scientific journals. As a result the CODIL system was used to model an online system (accessed directly via dial-up rather through BLEND's very limited Notepad software, which did not allow such flexible online working) where papers on different applications, including the student activities, could be combined in an online "journal" including the options of including raw data, operational demonstrations, and hyperlinks between papers. This is summarised in ***A Software Package for Electronic Journals*** (1983). This was the only truly interactive paper submitted to the BLEND project, as all the other submitted papers were straight text documents which could be read online.

MicroCODIL

Much of the research in the 1980s concentrated on the development of the MicroCODIL system. The aim was to demonstrate that the basic decision-making unit was so small and fast it would run on a very small home/educational computer and could carry out a wide range of tasks. Papers describing what the system would do include ***A Microcomputer Package for demonstrating Information Processing Concepts*** (1985), ***Human Factors in Systems Design*** *(1987)*, ***Introducing Expert Systems to Pupils*** (1988), ***A Flexible Approach to Local History Daya Bases in the Classroom*** (1988), ***A Psychological Approach to the Computer Handling of Historical Information*** (1990) and ***Moving Information Technology Research from the Laboratory to the Classroom*** (1990). The software was well reviewed – see ***Reviews of MicroCODIL***[15] - but did not fit in with the National

---

[14] *Plans and initial progress with BLEND—an electronic network communication experiment*, B Shackel, International Journal of Man-Machine Studies, Volume 17, August 1982
The Electronic Journal: A User's Perspective. Cliff McKnight, 1993

[15] The published reviews of MicroCODIL

Computers in Education Journal, January 1987
Gabriel Jacobs, ***The Micro User***, February 1987
Keith Chandler, ***Network User***, March/April 1987
Jonathan Evans, ***A & B Computing***, April 1987
Educational Computing, April 1987
Computers in Schools, May 1987
Mike Page, *New Scientist*, 24th September 1987
Jill Phillips, ***Your Computer***, October 1987
Steve Mansfield, ***Acorn User***, November 1987
Jaquetta Megarry, ***Times Educational Supplement***, 6th Nov 1987
R. McDermott ***NEXT*** (Ceefax), 13th November 1987
A & B Computing, December 1987
Francis Botto, ***Disk User***, June 1988
Jean Underwood, *The Psychologist*, September 1988

Curriculum (which assumed more conventional systems) and, probably by trying to do too much, the package was not really robust enough for classroom use.

Modelling the Brain – The role of Culture

To understand the links between CODIL and the way the brain works it is useful to consider whether the human brain is significantly different from that of our nearest ape relatives (genetically they are extremely similar) or whether the difference is related to our ability to share large volumes of information over many generations (i.e. culture) while, when an ape dies, all the information it has gathered during it's lifetime is lost. If we look at the history of scientific research it is taken for granted that our intelligent ideas are arise because we have access to vast quantities of shared information. Sir Isaac Newton effectively recognised this when in 1675 he wrote "*If I have seen further it is by standing on the shoulders of Giants.*"

Both CODIL and the current AI large language models effectively recognise that much, if not all, of our intelligence is buried in our cultural exchanges, and the most important difference between human brains and those of our animal cousins is the ability to store large quantities of cultural information.

The large language computer model collects huge volumes of cultural information. Of course there is a vast amount of duplication in collecting culture from many different sources, and this duplication allows them to make reasonably good probable predictions. This is really no more than a greatly scaled up version of how the Greeks must have analysed very many (in human terms) planetary observations to explain the apparent motions of planets using epicycles. It would seem that the large language models tell us very little about how the brain "understands" what it is processing, just as the Greek's epicycle model tell us very little about a planet is and how it is moving. Perhaps the biggest difference is that by using powerful large modern computers the large language models can using many orders of magnitude more variables that were available to the Greeks over 2000 years ago. Of course the large language big data model is a useful and powerful tool but it would be wrong to assume that it tells us much about how our brain "understands" information, or how we can build user-friendly computer systems that understands people.

If it is agreed that a significant part of human intelligence is embedded in culture we share this means we should reconsider the way we look at artificial intelligence packages – as the algorithms which drive those packages is also based on culture – and it is perhaps inappropriate (in terms of scientific research) to make a rigid distinction between the way the brain uses culture to write very sophisticated "black box" algorithms and the way the brain uses culture to form the knowledge base which the algorithms are based.

How CODIL models the brain

If you want to design an interface to interface two different information processing system you need to understand how both systems work. We have no real control about how the brain works, but we do have control over how we design computer systems to process information – and for maximum mutual understanding the computer should model what the human brain does. This is what CODIL tries to do:

- CODIL maps information onto a network, which fits in with what is known about the physical organisation of the brain. This is in marked contract to the stored program computer model that maps information onto a numerically array.
- The CODIL model uses a two way language (like natural language between humans) using the human's own vocabulary, The vocabulary is based on nouns (the name of objects) or sets (the name of a collection of objects) and giving names to things must have been an evolutionary first step in the development of language.
- In human language any piece of information (**X**) can used as "data" (= I know **X** is true), as a conditional test (= is **X** true?) and as a command (= remember **X** is true). In the same way in CODIL the difference between "data", a "conditional test" or a "command" relates to the context used to access the information.
- The brain evolved in a complex environment where there would be known unknowns and unknown unknowns and to ensure survival it needed to be able to identify known knowns. Because CODIL does not make a formal distinction between "program" and "data" it is relatively simple to modify the links between the nodes in the network to identify the patterns to form useful predictive known knowns. This idea is the basis of the limited learning features that had been tested by the time the project was abandoned.
- The human brain has a limited working storage (short term memory) which can handle about 7 pieces of information at any one time, while a computer can hold a many thousand pieces of information. CODIL is based on the idea that, in order to be understood by humans, the computer must process information at a similar scale to human brain.

When the project was discontinued there were several areas where more work was needed to see if CODIL continued to model the brain:

Handling images and other sensory input. When the project started it was assumed that all input would be from a keyboard and no work had been done on visual input, either of images or as speech or text recognition. It is possible that this omission continued into later research because I have aphantasia – in effect I suffer from a "disconnect" between the logical and visual parts of my brain – resulting in my designing a brain model that also has aphantasia!

Logically there is no reason why an object in a set should not be stored as a binary image rather than text, apart from the fact that the decision making unit would need to be able to decide whether two images represented the same object. The way forward to extending the CODIL model to cover a wider range of sensory inputs would be to consider multiple CODIL systems optimised to handle different types of sensory information. The later version of CODIL provided a route for information to be exchanged with other conventionally coded applications, and there no reason why this external application should not be another AI package.

Natural language. A statement such as:

MURDERER = Macbeth; VICTIM = Duncan;
WEAPON = dagger.

can easily be morphed by a human into a natural language statement. The research on the TANTALIZE task show that CODIL can act as a powerful information processing language, capable of using a knowledge base containing itels such as **VERB = Murder** which coukd then be used to generate the sentence

Macbeth murdered Duncan with a dagger

However no sentence generating knowledge base was set up and tested during the original research.

Leaning. Several different methods of optimising the knowledge base were included in the experiments. The simplest involved a simple reordering the linkages in the network to minimise storage and access times. Another technique, involving feedback, ensures that when TANTALIZE was used to solve problems the answer was found something approaching the minimum number of steps. If fuzzy logic was used the system could look for the most probable answer. However it is clear from the archive that more testing would be useful

The FACTS and current context.
The **FACTS** are the CODIL's system's equivalent to the human short term memory, and defines the current context. To keep the system user-friendly the number of items needed to be kept low. This meant removing items when they were no longer needed. For instance if the decision making unit explored a blind alley in the network the system needed to backtrack and remove items which were no longer relevant. However an attempt to forget items which had not recently been used was unsuccessful. While the default **FACTS** housekeeping algorithm clearly worked no attempts were made to see if CODIL could reproduce human-like forgetting if the number of items became excessive.
TANTALIZE used a larger FACTS set (up to 30 items) and highlighted an interesting point. In solving a number of problems the heuristic problem solver clearly was finding the answer by a human-like route – but following a path through the network where the human would loose track because of short-term memory overload. There is logically no reason, apart from human compatibility, from keeping the number of items in the FACTS low, and it should be possible to use a CODIL-like system with a "short term memory" of several hundred items – which would turn it into a black box system, This raises the question of human-like AI which uses so much information that humans cannot understand what it is doing.

Reassessing CODIL

The aim of the current reassessment of the project is to assess the potential value of preserving the bulky original records and the following points suggest the records are likely to be of continuing interest:

- The project, originally supported by the LEO computer pioneers John Pinkerton and David Caminer, was one of the earliest projects to look at the human-friendly computer interface and this means the research could be of interest to computer science historians. The reasons for its abandonment could be of interest to anyone studying the funding of unconventional "blue sky" ideas.
- CODIL was planned to be a tool for directly mapping commercial applications directly onto an associatively addressed network when all conventional procedural languages map tasks onto a numerically addressed array of numbers. This appears to be a unique feature.
- It appears that the CODIL project unintentionally reverse engineered how the human brain processes information by using its short-term memory. This aspect of the research and it's relevance to the evolution of intelligent brains was not adequately covered in the original research. In addition a review of recent research show that there is still no still no adequate evolutionary model which explains how the brain's network supports intelligence. CODIL suggest such a model, but the project was terminated before several important aspects had been examined.
- To ensure that the system could always explain what it was doing to a human observer the size of the main work area (The FACTS) was deliberately kep small to match the size of human

short-term memory. This appears to be another unique CODIL feature. However if the size of the work area is significantly increased the result could be another blaxk box AY package.

- CODIL proved powerful enough to support a heuristic problem solver which solved over a dozen consecutive *New Scientist* "Tantalizers" as they were published in the weekly magazine. This was a test that I don't think had been attempted at that time by any other AI based problem solvers – where the puzzles to be solved, and described in their papers, were specifically selected by the researcher.
- A practical (but small scale educational version) working transparent system was produced in MicroCODIL. This allowed the user to see how the system was processing a task at any time, just by pressing a function key. which attracted many favourable reviews in comparison with other educational AI tools.
- Because the research was finally abandoned in 1988 none of the major original CODIL interpreters will work on existing computers – and the ability to rerun and extend testing unfortunately restricted the scope of this assessment. Producing a powerful new interpreter might be appropriate and allow further evaluation by, for example, including image processing. In addition increasing the size of the FACTS (the number of nodes that can be active at any one time) could extend the model to handle complex tasks which could not be done by humans because of the limited capacity of human short term memory.
- The possibility of integrating this transparent approach with large language models of natural language has never been considered, but if the large language model could generate CODIL-like statements this would only require adding a few additional recursive pathways.

Was Neurodiversity a factor in the project closing down?

The British Computer Society has recently created a Neurodiversity IT Specialist Group to "*help the industry access the unique talents of neurodivergent IT professionals, support those individuals in their work and raise awareness of their contribution to broader society through their work.*" It is now widely accepted that neurodiversity can be a strength and an asset for creativity, if neurodiverse people are given the support they deserve. Neurodiverse people (with autism, dyslexia, ADHD, bipolar disorder, etc) have different ways of thinking, processing information, and communicating their ideas, and it is relevant to consider whether the fact that I have aphantasia and autism is relevant to the history of the CODIL project.

Neurodiversity can be a strength and an asset for creativity, if neurodiverse people are given the respect, recognition, and support they deserve., as their creative contributions can seem too radical, impractical, or irrelevant by others who do not share their perspective or vision. Neurodiverse people can have difficulty communicating with neuro-normal people who have different cognitive styles or preferences. This can lead to frustration, isolation, or low self-esteem for neurodiverse people, who may feel misunderstood or undervalued. Because their ideas may be seen as too controversial they may face discrimination, exclusion or exclusion. This can limit the access, resources, or opportunities for neurodiverse people, who may face barriers or challenges in developing their skills, talents, or potential[16].

Of course, when I was a child some 80 years ago I would not have been diagnosed as neurodiverse, but at the time I had difficulties in establishing social networks with fellow pupils, and developed a low self-esteem – a problem which is common with autistic children. These weaknesses have

[16] This paragraph is an edited. Version of the replay given by the Bing chatbot when asked about the disadvantages of being neurodiverse.

continued, to some degree, and undoubtably explain why I was not a good manager of the CODIL project and made several unfortunate decisions.

There were also two external non-technical events which had a negative effect on the development of the CODIL project. The first was the merger to create ICL shortly after the project had begun, but before any significant result had been obtained. The archives throw light on how the new ICL board closed down some interesting research projects that were not immediately relevant to the development of the 2900 series. The second major setback was the painful illness and death of my eldest daughter, which left me extremely depressed and led to my taking early retirement – and the closure of the CODIL project.

What is more relevant are the reactions I got to CODIL because the approach was "outside the box" and it is well known that neurodiverse people have difficulty in getting their creative ideas acceptrd. To me (possibly because of the abnormal way my neurodiverse brain is "wired") the ideas underlying CODIL seemed pretty obvious. Early on I realised, from other people's reactions, that the approach was new - but it took me several years to realise why many computer programmers AI experts assumed that the approach too good to be true, so simply rubbished it without looking at the details. One incident sums up many of the early criticisms – and is, I understand, typical of how neurodiverse ideas are often dismissed. When ICL was withdrawing support I arranged a meeting to try and get approval for some funding, and took an example of CODIL to show how the language might be used in a human resources department. The computer expert who was interviewing me said that he didn't understand how the example worked and when I suggested we should ask the people in the human resources department what they thought of it his reaction was "*but they are not computer experts so their opinion would be totally useless."*

Later in the research I produced the TANTALIZE problem solver and found it difficult to get papers published. One paper I submitted described how TANTALIZE worked, together with details of actual problems solved, including comparative timings for problems described in other AI papers. The anonymous rejection report claimed that the system I describe was too theoretical ever to work, completely ignoring the fact that I had submitted details of the system actually working. Another attempt to publish was rejected because all good AI research is implemented using the Pop-2 programming language. It would seem that the language in which a computer package was written was more important that whether the system actually worked.

Shortly after this I had the opportunity to show some of the working problem solving listings to one of the leading AI gurus of the time and while he showed some interest he carefully explained that while my system might be relevant to commercial systems, it had nothing to do with AI because I had not demonstrated that it could be used to play chess. As a result I carefully drafted another paper which I submitted to a leading American journal. It came back with four reviews - two were scathing, one was favourable, and one simply said they did not understand it. I was so depressed I gave up most of the CODIL research targeted at AI – and it was only years later, when going through the archives, that I read as far as the final paragraph of the covering letter. The editor, who would have known the authors of the scathing reviews, urged me not to abandon the research because there might well be something in it because it annoyed some people so much.

At this stage I felt there was little point in trying yet again, if anonymous reviewers kept on rejecting papers because the system could not work because it did not conform to their preconceived ideas. The answer was to shrink the system so that I could enclose a working copy on a floppy disc, along with any submitted paper. This work was disrupted by the aforementioned family suicide but the result was MicroCODIL, which runs on a BBC computer. When it was trial marketed it attracted

many very favourable reviews by named reviewers[17], presumably because the reviewers could easily see that MicroCODIL was a friendly transparent system. However I came to the conclusion that it would be better to move it to a more powerful personal computer, which would allow bigger tasks. In fact the only criticism I had of MicroCODIL was fatal to the future of the prpject. A newly appointed head of department, who considered himself an expert in main stream AI, decided that AI meant applying for massive research grants for powerful computers. He made it very clear that having a member of staff who had developed an AI package that would run on a tiny computer brought the department into disrepute and the sooner I left the better. Probably because of my post-traumatic stress disorder I agreed to take early retirement, after having written an article in the New Scientist[18] about the problems of doing unconventional research. It was no great relief, because the CODIL project had already been closed, when, a couple of years later, there was a union enquiry into other related cases of serious bullying at my former employer[19].

After the project having been closed there was one final public criticism. Gerah Voldman wrote a review of the final CODIL paper[20] in ACM Computing Reviews when he said "*This paper aims to investigate the design of a small multi-purpose language, but the number of demons used in the project makes it ultimately unrealizable.*" To be fair he was an American writing a review of UK based research and concluded that the project was unrealizable because I had difficulty in fitting the system into a “toy” UK built computer with 32K bytes of memory at a time when most AI-linked research used computers with a megabyte or more direct access memory backed with generous hard discs, rather than a small floppy disc. On the other hand, was he saying this all looks too good to be true so I must find a plausible sounding reason to dismiss it. After all, British Computer Society set up their Neurodiversity Specialist Group because it realised that neurosdivere individuals were likely to come up with interesting creative which were lost because of unfair critism.

The Future of CODIL

There are many areas where more research could usefully be done but there are limits to what I can currently do from my bedroom office as I am now 85 and a carer who is officially in deep retirement. Because of my age I am not planning to return to full-time academic research, but I am still able (but for how long because of my age) to discuss and expand on the original research or to make arrangement for the safe archiving of the detailed original documentation. A detailed paper is currently being prepared on the network interpretation of CODIL and the light it throws on the development of transparent intelligent systems and the possible evolution of intelligence. If you think you can help, or have any queries or suggestions as to how research on the CODIL model can be restarted I would be interested to hear from you. I can currently be contacted at chris@codil.co.uk.

---

[17] Reviews of MicroCODIL

[18] Why "Blue Sky" research is so difficult, *New Scientist,* 14 July 1988

[19] UK universities must break their silence around harassment and bullying, David Batty, The Guardian, 18 April, 2019. This article clearly is not directly related to the enquiry into bullying that was held at my former university shortly after I took early retirement. As I had already left my case, as far as I know, was not even mentioned in that enquiry and if there were any NDAs involved I was never asked to sign one.

[20] CODIL. The architecture of an Information Language, Computer Journal 1990