Miser Project: Interpretation, Representation, Computation, and Manifestation*

Dennis E. Hamilton (Posted on 2019-02-11-14:24)

Introduction: Miser Project conception starts from reasoning with and in certain mathematical structures. That foundation is used with mathematical thinking in establishing a **model of computation** and subsequent **reasoning about programs** and **their dependable employment for practical purposes**.

Dependable manifestation of the computational model in **implemented operations** of a digital computer rests on **empirical determination** that the computer's **performance yields valid** computation-model **interpretations**.

The Miser Project use of "model" is reserved for a mathematical- logic situation around interpretations of theories, a flavor of modeling (or theory-building) that exposes **the marvelous utility of computers as instruments of human purpose and experience**.

* Mind-map notes and comments by William L. Anderson,
https://orcid.org/0000-0003-3200-7947, 2019,
CC0 https://creativecommons..org/publicdomain/zero/1.0

1. Recap: Computation-Model Stage-Setting†

† I need definitions (examples, or notes?) for:
- model of computation
- effective-computability[1]
- operational computational process
- computational representation
- mathematically-demonstrated correspondence
- manifestation (description of this is hard)
- tacit manifestation (still a bit fuzzy)

A.With regard to computational interpretation, effective-computability is taken as a condition on deductions that are available given a particular mathematical representation of a function or predicate.

B. The transition from an effectively-computable representation in any structure(?) to an operational computational process of any kind is suggested, not demonstrated.

C. It is posited that the interpretation-in-‹ob› approach is sufficient for computational representation of the same effectively-computable mathematical functions as any other recognized model of computation. Demonstration of such universality (i.e., Church-Turing computability) and common limitations of that universality is **one Miser Project objective:** making this aspect of theoretical computing accessible in practical, demonstrable, and understandable terms for computing practitioners and other interested parties.

2. Structure, Interpretation, Representation, Computation, and Manifestation†

Some informal notions are adapted and restricted in the context of the Miser Project to assist in capturing the essence of computation and its practical application.

Structure: mathematical characterization of theoretical entities, in terms of domains, primitive notions, and applied logical theories.

Interpretation:
- *mathematical:* arrangement by which one mathematical structure is modeled in another using a mathematically-demonstrated correspondence(?)
- *linguistic*: interpretation of a text in a (standard) semantic domain
- *empirical*: achievement of engineered mechanized operations confirmed to satisfy a model of computation
- *social / scientific:* identifying objects in physical/social reality as evocative of theoretical entities, and sometimes *vice versa*

Representation:
- *mathematical*: formulation of mathematical functions in a given structure's theory language
- *conceptual:* the entities and functions of one structure that serve as interpretations for those of another, thereby representing that other structure
- *manifold*: a representation may provide multiple entities that are equivalent as valid interpretations of a single entity in another structure

Computation:
- *conceptual*: performance of (algorithmic) procedure p such that representation of some entity x, the operand/data, is transformed into a representation of y, satisfying specified functional relationship y = f(x).
- *informal*: the result of a computation, e.g., "the square root of x", or, interactively, "achievement of the end-game in *Tomb Raider*"

Manifestation:
- *general*: perceptual object of attention recognized in some other object or activity
- *operational*: act of producing/presenting objects or activities of such perceptual attention
- *interactive*: with computer systems, the appearance evoked in the representation of computational results/operation on some medium (e.g., the perceived images on the surface of a computer-operated graphical display)
- *tacit*: objects perceived as being manipulated via interaction with a computer system operating in keeping with manifestation fidelity [???]

rename Representation: *reflective* to be Representation: *manifold*?
Rationale: definition indicates multiplicity of representation. could use *multiplicity*

3. Review: More About Structure

Structures evoke theoretical entities. There is no suggestion of material or natural existence. Whatever conceptual notions one might harbor about the subject of a structure, consideration and discussion of the properties of such entities is confined to what can be reconciled entirely via expression and deduction in an applied-logic theory, along with any informal statements that restrict and/or inform the constructions and valid deductions.

References:
1. M. Davis, *Computability and Unsolvability*, Dover Publications, NY, NY, 1982, ISBN 0-486-61471-9, pp. xv-xviii.

4. Interpretation: Variations on a Theme

4+. Concerning Representational Equivalence

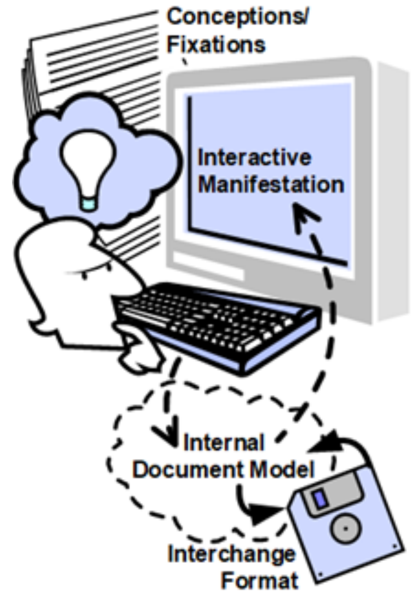
5. Choreography of Computation

6. Manifestation and Tacit Awareness

There is a point with computers where the experience of using the computer is as manifestation of something that is evoked by the exhibition of computational-processing activity.

Tacit recognition is important in how manifestations are evoked with so much fidelity that the manifestation is perceived as being manipulated and preserved by computer interaction. The accomplishment is a wonder of engineering of subordinate processes to achieve such fidelity without interference (except when it doesn't).

One case of tacit understanding will arise early. It is easy, as a computer programmer, to recognize a formula that has some mathematical appearance as expression of a computation; this mistaken attribution is encouraged by the borrowing of mathematical notation for programming-language purposes.


Conceptions/
Fixations
Interactive
Manifestation
Internal
Document Model
Interchange
Format