Building evolving systems to meet dynamic operational requirements: major design issues

Allen Brewer, Ph.D.

Deterministic Inductive Logic (DIL) is a novel approach to formulating and leveraging description to enable aggregating descriptive statements to reason about aggregates (e.g., categories) in dynamic and situational contexts.

The two DIL contributed truth-values enable representing a semantic element as: (1) “unknown” and (2) existentially quantified, a state in which two or more facets of a mutually exclusive and collectively exhaustive logic term are present or have been asserted in a situation, context, category, etc. DIL enables operations to combine(), compare() and contrast() truth-value feature vectors (e.g., profiles). DIL facilitates testing a subsumption relationship between two profiles, where those profiles may be used to describe statements, cases, classes, cohorts, etc. By assigning truth-values to semantically consistent concepts, linking phrases (e.g., predicates), propositions and systems of propositions, the description logic can be used to express very complex conceptual models. The approach can be used to reason about descriptive profiles and assess whether one profile is included or excluded by another. A profile that is not subsumed is logically NOT of the the class/type represented by the profile to which its subsumption was evaluated. A semantic truth-value description profile may be used as a query formulation which, when implemented with a compatible repository, may be used to retrieve statements, cases, classes, cohorts, rules, etc., expressed in the description system, which may be combined into a quantified model. DIL's computable semantics enables a novel information retrieval system for storing experience, rules, decision models, descriptive models, etc., that can be leveraged for data mining, decision making, situational analysis, etc.

Dynamic accommodation and adaptation involves both the operational cycle and the system's maintenance and improvement cycle.

The two most central issues involved in systems dynamically adapting, accommodating and learning are the: (1) operational mode modification/accommodation cycle, and (2) software maintenance and improvement cycle. A system must be capable of functioning within a change cycle that is shorter than the rate of operational change or the decisions it makes will be made using outdated information and decision models. The maintenance and improvement cycle of a system must be shorter than the rate at which requirements are discovered or changes are identified or the system will be forever behind the acknowledged needs for system functionality. These two cycles apply to a system's descriptive capabilities and to the needs to modify descriptions sufficiently quickly and flexibly to enable the system to be incrementally improved so that it can become progressively more complete, correct and specific over time, at a rate sufficient to meet the incremental improvement demands for new/different descriptive requirements.

The pacing factors for descriptive systems are the completeness, correctness, specificity and extensibility of the description system.

In a description based system, the specificity of the system is a function of the specificity of the description supported by the system and the specificity of the description assigned to each case/object/instance. One can generalize a specific but one cannot refine a generalization to become more specific. The capability to construct and deploy very specific descriptions is critical to achieving precision performance. The capability to add terms and descriptive patterns to meet specificity requirements is critical to evolving a system's information or knowledge representation. To achieve higher sigma (>1) information retrieval and filtering performance, descriptions should be as complete, correct and precise as practicable. For example, a specialist's description of a retina may be usable by a general ophthalmologist and may be computed into a generalization for a general practitioner or into a user friendly form for patient communication and education. Retaining the precision of the retina specialist assures that information is not lost permanently by its generalization. In accounting, for example, the accountant's general ledger retains the specifics of all transactions which are reported at levels of generalization that are matched to user information needs to facilitate managements' understanding and to enable targeted information use.

Data driven semantic infrastructure design enables parallel description and description logic improvements.

By constructing the description system using a data driven design, changes in the description logic (e.g., how descriptions are formulated) can be made without interrupting a system's operation. The proposed semantic infrastructure enables the addition of terms, the truth-values associated with those terms, and the patterns for describing measurements, observations, rules, situations, etc., to be modified in parallel with the operation of a system. For example, rules that are used to drive rule-based systems can be formulated, added and modified in a rules repository without stopping a system's operations. During the next model construction cycle, if the size of the logic vector has changed its length can be updated to accommodate an extended feature vector. New rules and descriptions that use new descriptive elements can be aggregated during the next combine() operation to include enhancements and extensions in the next model built. The data driven design approach can reduce the software maintenance cycle required to add terms, add rules, etc.

Semantic elements are logic elements not linguistic expressions.

The semantic infrastructure is designed to register concepts and conceptual models. It assigns a universally unique semantic identifier (uuSID) for each *term* that is defined and for which applicability and use rules have been specified to regulate the assignment of truth-values to that semantic element. Semantic elements can be related by connecting them using linking phrases. For example, thesauri can be constructed by linking synonymous elements, related elements, broader terms, narrower terms, etc. Elements might be replaced by creating a proposition like term1--<replaced by >--term2. Preferred terms can be identified using propositions like term1--<preferred term>--term2, etc. These types of propositions can be used to manage a terminology, translate descriptions by substituting preferred terms, remove replaced terms, etc.

The subsumption test enables situational analysis of descriptive feature vectors.

The subsumption test works on any two semantically compatible truth-value vectors where, for example, one of the vectors may be a situational rule and the other might be used to represent the situational condition of an environment. If the situational rule is subsumed by the situation's environment the subsumption relationship can be used to build a model and/or trigger a decision or action. In such situations, logic elements that are universally quantified that fulfill trigger conditions may be viewed as discriminants with respect to a rule or a situation. Discriminants may be conceptualized as “actionable” information or knowledge elements in a relativistic situation—action or situation—decision context. When elements are unknown but may be required to trigger an action, a system might choose to acquire or resolve those element values that enable it to make decisions or initiate actions. A system might prioritize diagnostic activities to resolve values based upon their relative importance, value, and/or cost of resolution to focus system resources on making relevant, useful and valuable decisions.

Information retrieval and model building in the semantic truth-value space.
The information retrieval functionality of the semantic truth-value space enables a situational profile to be used as a query formulation to retrieve, for example, all rules that are subsumed by a situational profile, to, for example, define a set of decision alternatives that can potentially be ordered or processed to make a decision or to select an action. A subset of alternatives based upon a selection criteria can be ordered based upon ordering principles like value, cost, etc. An example might include building a model of rules in which an *authority rule* may be defined in a rules repository where that authority rule is subsumed by a situational context (e.g., profile). Such a process might be used to model what may be permissible in a situation described by or bounded by the semantic elements and their truth-values as represented by a situational profile. Comparable models might be constructed to describe situational *obligations*, *instrumental capabilities*, *action discriminants*, etc.

System elements that are common and reusable in a descriptive system include the underlying representation and any code that leverages descriptions for reasoning about decisions and actions.
The capability to partition pieces of description and executable code that can be shared by different systems that may have overlapping requirements, situations, etc., enables one to define "common" elements that can be "re-used." For example, a hospital robot that has the capacity to navigate a hospital and provide telepresence, may need additional instrumentation and software to also provide clinical procedure support, such as providing an ECG capability. The navigational knowledge and capability (e.g., KI aspects of the AOKI agent model) may be shared across all robots that navigate within the same territory, where the obligations and authorities (e.g., AO aspects of the AOKI agent model) may be a function of the characteristics of a specific robot/user/system/etc. For example, there could be rules that operate at higher levels of generality that can be reused if they are complemented with rules that add necessary specifics to enable a system to function in specific situations or environments. For example, contamination rules may apply throughout a hospital to all robots, where housekeeping robots may respond differently in situations related to entering patient rooms than might clinical instrument robots.

Clinician as intelligent agent—instrument operating in an OODA loop to deliver patient care.
In the clinical encounter documentation project, a clinician is conceptualized as a type of "instrument" that engages in observation, orientation, decision and action. In addition to information and knowledge activities, a clinician may also perform diagnostic and interventional procedures (actions) in conjunction with clinical instruments. The clinician interfaces a clinical evidence system by documenting observations, orientations, decisions and actions in clinical encounter documents that are formulated using defined, pre-specified semantic patterns to express clinical encounter evidence, plans, actions and outcomes. These observations can be used to build a patient history, to model timeframes, to enable difference analysis, gap analysis, outcome research, etc. The capability to use clinical documentation to build custom tailored models of patients, for example, can enable intellectual access and navigation of what is known about a patient and can, if used in conjunction with an encounter experience repository, be used to leverage what is known about similar patients' experience to enable healthcare delivery systems to be patient centered and to learn from experience. The semantic systems architecture described and enabled by Deterministic Inductive Logic (DIL) has been proposed for clinical documentation that can be used productively for research and patient care.

(Notes: (1) The GAHMJ paper describes the description logic for clinical evidence and provides ophthalmic examples of clinical evidence description.

(2) The Affordable Care Act (ACA) set up the Patient Centered Outcomes Research Institute (PCORI) to fund patient centered outcomes research.
(3) The International Terminology Standards Development Organisation (IHTSDO) (see: http://www.ihtsdo.org) which manages the Systematized Nomenclature of Medicine (SNOMED) (see: http://www.ihtsdo.org/snomed-ct/what-is-snomed-ct) has adopted an architecture that does not support the construction of quantified models (FOL) or negation. It has a very long lead time for adding new terms.

