

Second Order Metaprogramming

David Dodds
Open-Meta Computing Inc
324-549 Dansey Ave, Coquitlam, B.C. Canada
david_dodds_2001@yahoo.com

Abstract - Metaprogramming, in the sense used in this paper (rather than the programming template sense) is the use of a collection of programs which orchestrates other (possibly collections of) programs. One of the requirements of the metaprogramming code is that it must collectively develop a sense of what the situation is and what should be done as a result. Situation awareness is an example technique used to interpret / analyze sensor data into a structured data set which indicates the "awareness" of the situation at that time. What is needed is the ability to functionally adjust both the trigger pattern of input and the resulting behavioural response. This is done through learning. Systems which are able to adjust their own metaprograms are called self-metaprogramming, and learning is a key feature of these systems.

Keywords: Second Order Metaprogramming, Conceptual Metaphor, Cognitive Fusion

1.0 Introduction

Metaprogramming is programming about programming. Used in this sense Lilly [1] describes a metaphor for what metaprogramming consists of and what capabilities it brings to a system possessing it.

"The built-in programs survived as a basic underlying context for the new levels, excitable and inhibitible, by the overlying control systems. Eventually the cerebral cortex appeared as an expanding new high-level computer controlling the structurally lower levels of the nervous system, the lower built-in programs. For the first time learning and faster adaptation to a rapidly changing environment began to appear. Further, as this new cortex expanded over several millions of years, a critical size of cortex was reached. At this new level of structure, a new capability emerged: learning to learn.

When one learns to learn, one is making models, using symbols, analogizing, making metaphors, in short, inventing and using language, mathematics, art, politics, business, etc. At the critical brain (cortex) size, language and their consequences appear.

To avoid the necessity of repeating *learning to learn, symbols, metaphors, models* each time, I symbolize the underlying idea in these operations as *metaprogramming*. Metaprogramming appears at a critical cortical size-the cerebral [system] must have a large enough number of interconnected circuits of sufficient quality for the operations of metaprogramming to exist in that [processor].

Essentially, metaprogramming is an operation in which a central control system controls hundreds of thousands of programs operating in parallel simultaneously."

This paper is about some of the components and effects of a computer system used to govern an autonomous robot through the usage of aspects of this kind of metaprogramming. Typically robot system computers have some operating system (software) which contains routines to perform various required tasks such as moving data around, inputting and outputting data, "driving" various devices and other not high-level operations. Using these services are programs which are designed to operate the robot and its sensors and effectors. These programs are largely low level code much like the operating system, differing only in that they use the code in the operating system to carry out wanted activities. Metaprogramming, in the sense used in this paper (rather than the programming template sense) is the use of a collection of programs which orchestrates other (possibly collections of) programs. The orchestration is far above the level of merely calling subroutines by rote. The metaprograms which govern the use or application of other programs are "higher level" or "more abstract" than the program code operating in the operating system (software). One of the requirements of the metaprogramming code is that it must collectively develop a sense of what the situation is and what should be done as a result. Situation awareness is an example technique used to interpret / analyze sensor data into a structured data set which indicates the "awareness" of the situation at that time. Predefined rules can be associated with the various "slots" in the situation awareness data structure, so as to monitor their content and

react to various content found there. This is much like instinct, once a situation has been determined to exist predefined behaviour is unleashed as a result. This is ok if the robot is a spider but for a functional robot in the field this instinct level of behaviour is not adequate.

Minsky, in his "The Society of Mind" described a collection of (software based) "agents" who, individually, were somewhat "dull" but collectively provided an adequate synergistic "intelligence" to govern a robot. This society of programs is not unlike that which Lilly describes in his technology of metaprogramming.

One of the powerful constituent capabilities or "agencies" of metaprogramming is that of making and using analogies and metaphors[2]. These make use of logical processing which is not hamstrung by the need of exact matches, exactly equal values. Lotfi Zadeh [3] has pioneered "Fuzzy Sets" as a means of processing "inexact" (not random) data. In this paper we see fuzzy set technologies used to support the representation and processing of metaphors and analogies. This processing is coupled with that of ontologies to provide a multimodal system able to apply conceptual metaphors functionally to more advanced robot governance.

1.1 Brokering Semi-Disjoint Ontologies

A "context extraction" component scans predicate content from the comment section of ontologies. The process uses Hidden Markov Models and phrase-structure-based templates. It identifies the meaning of predicates occurring in ontology-based geo-spatial descriptions by usage there. By this means the computer can determine what antecedents are necessary to constitute the occurrence of one or more particular ontological items, such as "above-Directly". This approach helps broker functional compatibility between a collection of partially overlapping (semi-disjoint) ontologies. These capabilities are part of the constituency of the metaprogramming ensemble mentioned above.

```
<owl:ObjectProperty rdf:ID="above-Directly">
  <rdfs:comment>(#$above-Directly ABOVE BELOW) means either that
    (1) the volumetric center of ABOVE is directly above some
    point of BELOW, if ABOVE is smaller than BELOW; or that (2)
    some point of ABOVE is directly above the volumetric center
    of BELOW, if ABOVE is larger than, or equal in size to, BELOW.</rdfs:comment>
  <guid>bd58fbde-9c29-11b1-9dad-c379636f7270</guid>
  <rdfs:subPropertyOf rdf:resource="#above-Generally"/>
</owl:ObjectProperty>
```

The Hidden Markov Models provide the metaprogramming system a means of recognizing certain linguistic constructs, in this case descriptions of required antecedent context needed to exist in order that the named, such as above-Directly, ontological entity be recognized as "active" (and "relevant", to the indication of the "situation". It is an element of that context, and is constituent of the "awareness" in a situation awareness data structure).

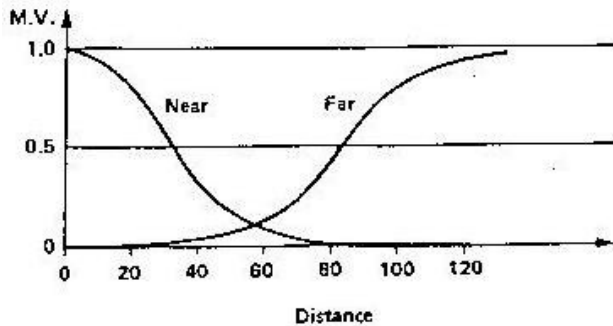
The following ontology fragment, IsNear, shows an example of how a fuzzy concept such as "is near" or "near (to)" is represented in an ontology and by its concomitant process (in this case the computation listed after the ontology fragment).

```
<rdf:Property ID="IsNear">
  <rdfs:comment>has a degree of nearness (by value). g1(x)</rdfs:comment>
  <rdfs:range rdf:resource="#www.open-meta.com/2001/ IsNear" />
  <rdfs:domain rdf:resource="#SvgEntity" />
</rdf:Property>
```

$$\text{IsNear} = g1 = 1 - (1/2 + 1/\text{PI} * \arctan((\sqrt{(x2-x1)^2 + (y2-y1)^2}) - k(1)/k(2))) .$$

The graph illustration shows what the calculation (IsNear and its fuzzy complement, Far) looks like. The Y-axis

is Grade of Membership Value.



Next we see an ontology fragment, Above, followed by its associated or concomitant programming code, also called Above. Each ontological item has its own concomitant programming code. The linkage or concomitancy is explained later in this paper in the section on MultiModeling [4]. The code shown inputs position values for two objects, usually from processed sensor data, and calculates a scalar value which is the grade of membership (“degree”) to which object 2 is above object 1. Zadeh's “linguistic variable” technology can be used to translate that scalar value into a sensible english statement, like “far above”, or “somewhat far above”, etc.

```

<rdf:Property ID="Above">
  <rdfs:comment>centroid of OBJ1 is anywhere above the centroid of OBJ2. Above means a y-axis value for
  OBJ1 which is less than the y-axis for OBJ2. (2D) </rdfs:comment>
  <rdfs:range rdf:resource="#SvgEntity" />
  <rdfs:domain rdf:resource="#SvgEntity" />
</rdf:Property>

```

```

public real Above( int y1, int y2 )
{
  if (y1 < y2) //2D Above
  {
    /* Far() = 1 - IsNear() */
    return 1 - ( IsNear(y2-y1) );
  }
  else
  {
    return 0;
  }
}

```

1.2 Cognitive Fusion

Cognitive Fusion, what is it? Anyone involved with robotics knows that useful robots must be able to sense the environment in which they operate. They must have some sense of the nature of the things in the world they interact with. In the earlier days of robotics the sensors were treated as individual instruments and the governing software of the robot made use of the measurements and data in a regimented if not somewhat disjointed way. It became apparent that better functionality of the overall robot could be obtained by integrating the panoply of sensors into something which increasingly approached the holistic co-ordinated nature of biological systems. We humans use, but are typically unconscious of, a highly integrated use of vision and proprioception. As adults we live integrated lives and never think about it. This integration took some time and effort to develop. When we were babies we thrashed our arms and legs about in the crib. This was instinctual. As we thrashed about we were taking notice of what it felt like, to move limbs. This is known as proprioception, the sense of what a body part "feels like" when at rest or moving. As we thrashed about we also were taking notice of visual stimulation through our eyes. And also we were noticing acoustic stimulation through our ears. Being provided with nervous system and brain which fortunately is able to "learn" we gradually built-up a connectivity between various senses of proprioception and senses of vision. Having two eyes we were able to learn to "see in stereo". The

world as seen from both eyes is identical except that the images are slightly displaced from each other. We learned when we were very young to cognitively combine these two displaced images into a visual sense of depth. This "feeling of depth" or "sense of depth" is created by the brain, just as "sound" is and "colors". In the world there is no "sound", just pressure differences and vibrations. In the world there is no "color" there is just electromagnetic energy of various frequencies and intensities. Our eyes can detect a narrow band of that spectrum and our brain generates the sensation of color from the signals our eyes send to it. Color occurs in the brain and no where else.

The brain also integrates the simultaneous patterns of limb proprioception with images in our vision. just as our brain is able to process, together two simultaneous but slightly different visual images to make apparent three-dimensional vision, depth, as "feeling" or "sense" is generated by the brain to integrate proprioception with vision. the "feeling of space or depth" that we adults have, take for granted, never think about is a generated "sense" or "feeling". Like the sensations of "red", "space" ("depth") occurs in our head not out in the world itself.

One can look at various visual illusions and see that parts of the visual system operate persistently a certain way, even when our knowledge or understanding informs us that what we are seeing (in the illusions) is not accurate or "correct". By integrating the various patterns of the sense of proprioception (of our limbs) with the images we see with our eyes we can cognitively correlate what "locations" and volumes wherein we can kick and thrash "feel like". Our brain learns to combine proprioceptive information and visual information into a "feeling of space" or "sense of space" in our consciousness. In essence we develop an integrated sense of "what space feels like", put another way, "I wave my arm about and see it and at the same time am aware of the proprioception information "in" my arm." One way of talking about that is to say "I feel the space (location, volume) [via the proprioception of my arm] where my arm is waving." This "sense" or "feeling" of spatiality is a product of integrating sense data and thereby providing a "meaning" (of that integration). It might be seen as a noticing and an expectancy built-up from it. Yet we do not have conscious access to the computation / processing / integrating itself.

At the same time we sense images and feel proprioceptive information in our body we also hear sound. The same integrative system allows us to correlate particular sounds with particular images. This allows humans to have a second (vision is first) remote sensing capability which can provide useful clues about the nature of the world around us.

The awareness of these integrated sense-data makes it appear that which is sensed and integrated "occurs" at certain locations. This integrating system location is what provides us with a sense of "where I am", I am located at that place whose "sense" (or "feeling") is that generated by my brain [as a result of the integration.]

This integration of the senses (sense-data) is done by processing below the level of consciousness. Our awareness is handed the result of the processing, we are not privy to "what it feels like" in the midst of this processing.

1.3 Situation Awareness

Robotics systems, when operating in any non-trivial environment must be able to develop status information about the surround. In addition to some rudimentary form of touch robots need distance sensing in order to be more viable in that world. Visual and acoustic sensors available to the robot bring the possibility of receiving data about something which is not already immediately upon the robot. One term for this is situation awareness (SA). In this light SA amounts to applying a context to data sensed from sensors. In this respect SA could be said to be a kind of "making sense" of the sensor data. SA is the development of a structured data set from collections of data from a bunch of sensors. The sophistication of the software which performs the integration of the robot's sensor data determines the quality of the situation awareness. One way of doing SA is to have predefined categories (of situation) which have tests or conditionals which respond to the sensor data in known predictable ways. Loosely speaking it could be said that this approach has the system watching the sensors for data which

has certain particular characteristics or values. This approach suffers from a number of problems. If it is category based then the only situations this system is capable of being aware of is within those categories that are predefined. If other than God-like prescience was used to define those category detectors which winnow out the (identity of) categories from the stream of sensor data then there will be situations that occur that such a system can never be aware of. This (rather likely) lack of complete prescience in the software of this kind of SA is precisely the same dilemma faced by the author of the instruction book in Searle's Chinese Room. These (not entirely prescient) instructions are equivalent in some sense to instinct. Nature long ago seems to have found instinct alone insufficient as a means of governance of systems. Neural (ie processing) sophistication provided a means of improving on fixed instinct by providing an updatable version of instinct in the form of learning. Robotics systems are able to make use of machine learning systems to augment / update their rules / conditionals / categories. So far the machine based learning systems don't hold a candle to those biological systems but we do what we can with what we have figured out so far.

Cognitive fusion is about allowing disparate representations and approaches to interact co-mingle functionally to develop a new synergistic result. In some robotic systems sensor data from range and vision sensors is processed so as to develop a description of the physical surround into ontological / lexical form such as "walls, doorways, obstacles, holes (pits) , fences, ditches, hills" and so on. This integration may also be interpreted into a map or visual form (for human consumption). When some form of machine learning based conditionals are used to perform the filtering of the sensor data there is some likelihood that the integration product will not have the blinders in recognition of situation that early rule-based systems had. Expectancy is nice, it's useful but if there is no learning capability in the expectancy system it is partially blind from birth (biased).

By allowing the robotic system to functionally co-mingle products (results) from different agencies or capabilities it is possible for increased sophistication in governance of the robot.

One of the things necessary to do this is to not design the system based on scarcity thinking. Several four-core or sixteen-core processors should be doing the processing. Using a single uniprocessor computer is simply mulishness.

Cognitive fusion implies using multiple paradigms simultaneously. In the past this was notoriously hard. It is necessary to use multiple paradigms in order to expand the functionality of robotics systems. One of the ways of doing this to examine the means by which functional (not chaotic) collaboration between technologies can be made. One of the ways this has been done in the past was to develop and use of lingua franca, a language or representation into which the various participatory systems of approaches could be cast. Another means was the use of the Blackboard System where the disparate systems wrote their information in their native tongue and the other systems comprising the overall system each had to have translators to interface their representation (tongue) with that of each of the other tongues used by the other blackboard participating systems.

One wants to be able to functionally able to co-work neural nets, genetic algorithms, rules, fuzzy sets, objects and so on, so as to meld the capabilities of each to provide a collective coverage of all the areas of the totality. Each approach brings particular strengths and suitability (competencies). An orchestration of these systems might in some sense be aware of the strengths for each type and use this knowledge to channel data and processing activity (shades of work flow) to the most appropriate technology.

Cognitive fusion is at the core of the ability to process vision sensor, range sensor and proprioception data into a model of spatiality. (No claims of being able to produce "sense" or "feeling" such as humans have.) It is also at the core of the ability to relate particular sounds with particular visual activity.

1.4 Integration of Proprioception Inside a Robotic System

How might this integration of proprioception look inside a robotic system? In most if not all, robotic systems the various sensor modalities are compartmentalized and fusion, if it occurs at all, is done as a separate

compartment. The integration of vision and proprioception might be done in the manner used by the "MultiModeling" approach (University of Florida). Generally speaking, they had each modality operating as usual in a compartmentalized fashion in asynchronous but possibly synchronous time. An ontology was used to bridge pairs of modality; geometric models, dynamic models and a 3D interactive graphics environment. Their system was comprised of Functional Block Model and Finite State machine. One would add other expressions of dynamic modeling, such as System Dynamics Model, Markov Models, Petri nets, Queuing Models and difference and differential equations to such a system. Items in the 3D interactive graphics view had graph structures (links) connecting them to instances of particular items in the various modality modeling systems. A very simple graph structure can be implied which connects an XML structure (which could be an ontology) using a URL with an item in some other structure or file. It is also possible that the same URL link be used to connect with a "router" facility, one which can implement graph structures. An example of such a router is a topic map, a collection of nodes and arcs which overall adhere to a particular (ie topic map) definition of constituency. The single address departure point on the ontology is preserved by this but the depth of the connecting graph (may be) considerably deepened. The details of the connecting graph can be defined by dynamic process which is capable of making informed updates of the graph structure (via the topic map).

Since it is a graph structure links each such part of the ontology to the various modeling items it is possible to have knowledge embedded in the graph or pointing to the graph which can be used to augment the connectivity. Because it is a graph structure it is possible that part of it is used as a semantic network. This knowledge-in-the-connectivity can be used to achieve processing, filter, etc. and used to implement learning (the effects thereof).

SVG has a metadata element designed to provide the ability to embed metadata, including ontology and topic maps, directly into the graphics elements. The SVG Title and Description elements allow namespaces to be used in them and hence ontological and metadata entries there too.

```
<?xml-stylesheet type="text/xsl" href="anim02.xsl"?>
<svg width="16cm" height="14cm" viewBox="0 0 255 201">
<metadata xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/TR/. ..-schema#"
xmlns:daxsvg="http://www.open-meta.com/daxschema/">
<rdf:Description about="#arrowstreamer">
<daxsvg:Ingress resource="#funnel"/>
</rdf:Description>
<rdf:Description about="#arrowstreamer">
<daxsvg:Inside resource="#funnel" />
</rdf:Description>
<rdf:Description about="#funnel">
<daxsvg:Containz resource="#arrowstreamer"/>
</rdf:Description>
<rdf:Description about="#arrowstreamer">
<opencyc:above-Generally resource="#funnel" />
</rdf:Description>
</metadata>
<desc>Copyright 2006 David Dodds Example anim02 – demonstrate deBono diagram SVG animation with
Lakoff spatial metaphor</desc>
<rect x="1" y="1" width="253" height="199"
fill="black" stroke="blue" stroke-width="7" />
<text id="uplabel" x="230" y="20" style="font-family: Verdana; font-size:12.333; fill:blue">UP</text>
<text id="downlabel" x="200" y="180" style="font-family: Verdana; font-size:12.333; fill:blue">DOWN</text>
<g id="leftfunnelside">
<path d="M 99 180 L 99 57"
style="fill:none; stroke:green; stroke-width:10"/> </g>
```

```

<g id="rightfunnelinside">
  <path d="M 153 57 L 153 180 " style="fill:none; stroke:green; stroke-width:10"/> </g>
<rect id="arrowstreamer" x="110" width="3" height="20" >
  <animate attributeName="y" attributeType="XML"
  begin="0s" dur="5s" fill="freeze" from="50" to="170" />
  <animate attributeName="height" attributeType="XML"
  begin="0s" dur="5s" fill="freeze" from="20" to="143" />
  <animateColor attributeName="fill" attributeType="CSS"
  from="rgb(0,0,255)" to="rgb(110,0,0)"
  begin="0s" dur="5s" fill="freeze" /> </rect>
</svg>

```

Since SVG is expressed in XML it is possible for this (graphics) system to be a (visualization) anchor for a "MultiModeling" system. If, for human edification, one wanted to literally implement a "Theatre of the Mind" one could use this SVG system to display visuals resulting from interaction of the ontology, the dynamic modeling and the geometric modeling. Minus the visualization part this same scheme can be used by an autonomous mobile robot to represent / model the geometric and dynamic aspects of the world the machine finds itself in.

The graph based linking in the Florida Multimodel does not provide (itself) the same kind of result we saw when the brain takes input from two (or more) modalities and produces a "mix" (ie the sensation of spatiality). Perhaps an approximation of that "sense" is obtained if one used the SVG to provide an animation of geometric movement through time. For example, let us have an SVG graphic of a (perhaps stick-figure) baby lying in a crib. The animated baby waves his arm in the air, the arm bending etc occurring in the same time that it would occur with a real baby. Now let's reduce what the animation shows to the equivalent view of what a baby might see, just the arm moving around. What we have here is a visualization and a metaphor of the cognitive event "seeing my arm move / wave around" plus the proprioceptive (arm / limb) information. The actual visualization generated by the SVG is arm-appendage visual changing location in the SVG viewing area. The timing of the change of location of the arm visual is the same as that of a real baby. The SVG code which produces this animation is a set graphic elements for "arm-parts", like hand, lower arm, upper arm, shoulder. The position / location of these parts is specified by some scripting such as Ecmascript. The scripting contains time-based iteration resulting in "human-like arm motion". So the SVG shows the visual part of this "cognition" (that the baby might have) and the SVG code has the "proprioceptive" information in it, but that code is not part of what is actually shown in the picture. It is "behind the scenes" as it were.

Well the SVG code is not actual proprioceptive information it is actually a time-based encoding of what "limb position changes must occur in the output". A better rendition of the proprioceptive information can be had by using the "physics computations" [5] [6] [7] [8] such as applied to high end computer game "action-graphics". The better quality computer games use extensive calculations to make the graphics as realistic as possible using "physics computations" (like "mass", "inertia", "gravity", "elasticity", and "viscosity") amongst others. Such computational information to do with this baby arm motion simulation would provide approximate proprioceptive information. Thus this SVG animation provides overall a metaphor for the "received cognitive event" experienced by the baby (brain). Such computational metaphors enable more advanced robotic operations, just like terms "event horizon" and "gravity well" have use to astronomers and physicists.

2.0 References

- [1] J. C. Lilly, *Programming and Metaprogramming in the Human Biocomputer* p. ix The Julian Press, New York: New York 1972.
- [2] G. Lakoff, M. Johnson, *Metaphors We Live By* Chicago, Ill: The University of Chicago Press, 1980
- [3] L. A. Zadeh, "From Computing With Numbers To Computing With Words — From Manipulation Of Measurements To Manipulation Of Perceptions," *Int. J. Appl. Math. Comput. Sci.*, 2002, Vol.12, No.3, 307-324.
- [4] M. Park, P. Fishwick, *Ontology-based Customizable 3D Modeling for Simulation* Dept. of Computer & Information Science & Eng., Univ. Florida.
- [5] D. Bourg, G. Seemann, *AI for Game Developers* Cambridge, Mass: O'Reilly Media, 2004.
- [6] S. Rabin Ed., *AI Game Programming Wisdom* Hingham, Mass: Charles River Media Inc, 2002.
- [7] S. Rabin Ed., *AI Game Programming Wisdom 2* Hingham, Mass: Charles River Media Inc, 2004.
- [8] T. Jones, *AI Application Programming* Hingham, Mass: Charles River Media Inc, 2003.