

Metaphors of Intent

Colin Potts

College of Computing, Georgia Institute of Technology, Atlanta, GA 30332-0280

potts@cc.gatech.edu

Abstract

Current research in cognitive linguistics questions demonstrates that metaphor is pervasive in the understanding and communication of abstractions of all kinds. This has wide-ranging implications for how we describe actual and envisaged information artifacts. Although it is not machinery or mind or mathematics or matter, requirements engineering and design methods must take some metaphorical stance toward their subject matter. This paper investigates two types of fundamental metaphors that recur throughout requirements engineering: (1) reification of abstractions as material substances and containers; (2) spatialization of abstractions as locations, trajectories and spatial relations; and (3) anthropomorphisms, some of which have recently been codified by Jackson as problem frames.

Consider the following task: You are to describe the requirements for an information artifact that does not yet exist. This is the problem of requirements description. For the sake of simplicity, we can assume that the requirements are whatever you say they should be, thus finessing such difficult problems as deciding who the customers are, extracting requirements from them, negotiating requirements, and validating that the requirements are feasible. Also we will ignore any issues of existing or required implementation infrastructure that constrain what can be done. Thus, by “information artifact” (IA) we could mean a stand-alone software application running on a general-purpose desktop computer, a suite of programs running on a distributed architecture, a handheld information appliance, a system of environmentally embedded programmable devices, or a service available through a network¹. What these technologies share in common is what is important for RE, and that is that they are intentional technologies that require some explicit description of purpose. Of course, all designed artifacts are intended to serve some purpose (although it may not always be clear to users what it is). What is different about information artifacts is that *implemented purpose is all they are*. A material infrastructure and mechanisms will be necessary to

implement the requirements, just as they are necessary to implement a car. In the case of the car, any description of function failing to mention wheels or brakes and merely describing patterns of desired movement and acceleration would be hopelessly abstract. But in the case of an IA, we would like to abstract away from unnecessarily constraining mechanisms if we can. And since computation and information manipulation are not grounded directly in the material world there is much more freedom to do so and to describe the intended purpose free of unnecessarily concrete restrictions.

Describing IAs is not just a matter of imagining something that does not exist. To illustrate this, consider a second task. You are now to describe an existing IA that you can manipulate and experiment with freely to somebody who does not believe that it exists. This second task is still requirements description, but removes the uncertainty caused by the future tense. It is requirements description if we accept with a big grain of salt that the artifact was implemented in accordance with customer requirements. Our task is to use the artifact to help us describe those requirements, not its implementation.

Is this second task possible? If it is not possible to describe the requirements for an existing artifact, it must surely be impossible to perform the more difficult first task of describing something imaginary. The grain of salt assumption that the existing artifact did in fact meet its requirements is irrelevant, since the task of describing it, though not the properties so described, would be the same whether it met its requirements or not. Since we are performing thought experiments, we can equally imagine in the case of an artifact that did not meet its original requirements a perverse user, not an original customer, who likes the artifact just as it is and whose requirements it therefore meets fortuitously.

Thus requirements description (in contrast to requirements engineering as a whole) addresses the problem of describing complex required phenomena, and whether these are actually achieved and for whom are not the issues at stake. Requirements are the properties that should hold in the real world that we want the IA to help bring about, not the properties of the IA itself. Moreover, the phenomena that a planned IA brings about generally do not yet exist, although they may. They are phenomena toward which a customer has an attitude, such as wanting them to exist or not rather than merely acknowledging their existence or possibility. And they are phenomena

¹We will assume that technology only is to be found within the boundary of an IA, not “designed” human activities, so a business organization may require and use IAs but cannot become one.

that are to be under the control of the IA and toward which the developer therefore also has an attitude, one of commitment. How is this act of description done, and how can it serve these different needs? How can it be done? Since the descriptions are always created by and for people, what do we know about human descriptive and interpretive abilities and tendencies that suggests or constrains ways in which to describe requirements in practice?

Answers to these questions go beyond what is conventionally regarded as human factors to the heart of what requirements descriptions are. The human factors issues surrounding requirements engineering have been addressed by previous research and by the notational and tool support in standard methods and specification languages. Human factors obviously affect how customers and developers understand requirements documents written in natural language, semi-formal graphical and pictorial notations, and formal languages, and how they understand examples presented through the means of storyboards, scenarios and animations. These are important practical concerns, but relegate the significance of human involvement in requirements description to a role that requires descriptions to be made palatable through syntactic sugar, pictorial views, and filtering.

In this paper, we take a more radical position than a concern with making descriptions palatable for customers. This position, which emerges from recent developments in cognitive science that have been driven by evolutionary psychology, neuroscience, phenomenology, and comparative linguistics, not by symbolic AI, formal linguistics, or analytic philosophy, questions whether describing should be considered symbolic representation at all. Rather than symbolic representation, the basis for all understanding is taken to be metaphorical mappings from a domain to be understood onto embodied perceptuo-motor schemas. Perception and action, are taken to be paramount, not static representations. Since this can come as a shock to those who take semantics to be truth-functional by definition, the next section, is a gentle tutorial on embodied cognition and the relevance of grounded metaphors to requirements description. The following three sections introduce and illustrate three major types of metaphor in requirements descriptions: the reification metaphor that abstractions are material objects, the spatialization metaphor that abstractions are locations or paths in space, and the anthropomorphic metaphor that procedural abstractions or required features are homunculi or mindlike machines to be understood in terms of commonplace human actions and communicative patterns. In all three sections, we are concerned with the role that a dominant metaphor plays in suggesting requirements for or enhancements to an information artifact and, conversely, how it closes off other possibilities. For example, the spatial metaphors that underpin formal models of security policies, and the roles that these metaphors play in surfacing or sublimating related requirements, such as information privacy, affect which

requirements are easiest to reason about and which are regarded as most legitimate. We conclude with a discussion of the broader implications of the new cognitive science for requirements description, requirements engineering in general, and the design of information technology in a human context.

1 Grounded and Figurative Metaphor

In psycholinguistics and the philosophy of language it was long taken for granted that literalness was basic to language understanding and meaning and that figurative uses of language such as metaphor and metonymy were colorful but inessential supplements to more basic mechanisms. Most computer scientists are familiar with the figurative sense of the word “metaphor” as it is applied non-linguistically to systems of signs on the user interface, such as the “desktop metaphor”, to the conceptual design of applications when following some user-centered design methods, or even to the understanding of such rich systems as the Internet, which can be construed for some purposes as a library, market, community, or whatever [18]. Such metaphors are deliberately and artificially created to promote creative understanding and design, and are certainly not the normal basis for how developers think about application requirements or features.

1.1 Ubiquity of Grounded Metaphor

In the past twenty years however, a growing pace of evidence from comparative linguistics, phenomenology, experimental and developmental psychology, and even neuroscience has pointed to a much more fundamental role for preconscious visuospatial and motoric mechanisms in the understanding of symbolic information that look just like these metaphors except for their fundamental nature and posited ubiquity. This evidence is summarized by linguist George Lakoff and philosopher Mark Johnson [12]. In a series of books, Lakoff and Johnson have argued persuasively that cognitive science should rethink the fundamental assumption that to reason is to construct and manipulate symbolic representations in a language of thought and should adopt instead a naturalist stance which takes evolutionarily adaptive perceptual and performance systems to have been co-opted as the ground for symbolic and abstract conceptual thought [9-13]. In his two most recent works [12, 13], Lakoff takes aim at the apparently most abstract edifices of human thought, Western philosophy and mathematics, respectively, arguing that systems of reasoning in both field rest on elaborate blends of image schemas and perceptuo-motor mappings. That is, the role of metaphor (in this sense of the word) is to provide a structural environment for reasoning, while the

role of symbolic representation is to reason formally and precisely within the constraints of that environment.²

Some of Lakoff and Johnson’s claimed metaphorical mappings from image schemas to abstract concepts are shown in Table 1. All take the form of “X is Y”, where “X” is an abstract concept and “Y” a summary of more primitive perceptuo-motor schemas. So familiar are some of these that it takes some effort to see the “X’s” as abstract at all and the X/Y mappings as more than self-evident definitions. Nor are the mappings arbitrary cultural constructions. For example, there are reasonable ontogenetic reasons why very early childhood experience should lead to the Important is Big mapping rather than any equally well-formed but semantically inappropriate mapping such as, say, Important is Smelly. Similarly, while the desktop icons for folders and trashcan may rely on cultural knowledge and therefore qualify as traditional figurative metaphors, the more basic intuition that these concepts can be treated as if they were spatially localized containers that may be opened or closed and into which objects may be placed (usually causing the objects no longer to be where they had been previously) relies on metaphors that are grounded in mappings from universal perceptuo-motor experiences to abstractions about information classification.

Naturally, to point out these mappings is to explain very little about abstract thought as they are merely the foundation on which higher-level metaphorical blends are constructed. For example, Lakoff & Nunez [13] spend over fifty pages describing several basic metaphors for arithmetic (arithmetic or number manipulation as object collection, object construction, use of measuring sticks, and motion on a path). Thus, as one proceeds upward from the foundational sensori-motor schemas, one constructs and encounters more complex blends of schemas as well as social and interpersonal schemas that themselves become the ground for still higher level metaphors. These mental and social schemas are still primitive by adaptive standards, as a finely tuned awareness of interpersonal and social reality is almost as adaptive for a gregarious animal species like ours as basic object perception. Thus what is fundamental or nearly so from the perspective of embodied cognition may look quite elaborate from the viewpoint of symbolic cognition or systems of formal reasoning based on an exclusively mathematical or physical ontology.

Johnson and Lakoff [9, 12] address moral theories as metaphorical. Although discussing ethical theory would take us far from practical requirements engineering issues, most ethical discourse revolves around the acceptance and ontological status of norms and principles. Johnson and Lakoff argue that apparently sophisticated moral theories rest on fairly simple metaphorical mappings that would apply equally to more mundane policies and norms. For example, a system of principles can be treated as

constraints or forces that act on one’s trajectory through life, or one’s conscience can be regarded as an homuncular observer and critic of conduct. These metaphors may seem fanciful when applied to information technology, but a run-time requirements monitoring feature is essentially a mechanical “conscience” implemented to ensure compliance of system behavior not with moral codes, of course, but with policies.

Table 1: Example metaphorical mappings from perceptuo-motor image schemas to abstract concepts (selected and adapted from [12]).

Metaphor	Abstract Concept	Concrete Schema
Important is Big	Significance in situation	Seen/felt size
Difficulties are Burdens	Obstacle to intention	Felt weight
More is Up	Quantity or degree	Seen or felt elevation
Categories are Containers	Classification	Seen or felt containment
Similarity is Closeness	Diagnostic/predictive similarity	Seen or felt proximity
Help is Support	Assistance toward intention	Felt firmness underneath
Time is Motion	Passage of time	Seen/felt movement
States are Locations	Situational equivalence	Seen/felt place
Change is Motion	Variation over time	Seen/felt movement
Action is Self-Propulsion	Autonomous activity	Intentional movement
Purposes are Destinations	Intention	Seen/felt places
Purposes are desired objects	Intention	Reinforcing object
Causes are physical forces	Causes and origins	Felt pressure & weight
Relationships are enclosures	Relational dependency	Seen/felt enclosure
Controlling is Being Above	Causal dependency	Vertical alignment
Seeing is understanding	Knowledge	Objects seen
Understanding is Grasping	Knowledge and comprehension	Objects actively felt

² This characterization was suggested to the author by one of the reviewers.

1.2 Grounded Metaphor in Design

This shift³ from representation of abstractions in a symbolic code to their embodiment in perceptual and performance schemas has profound implications for cognitive science and has generated some controversy. More relevant for our purposes here, however, are the equally thought-provoking implications for requirements engineering and applied computing generally that we may need to reevaluate fundamentally or at least problematize our taken-for-granted reliance on representations of system and real-world properties and behaviors. If all cognition, from everyday preconscious thinking to abstract professionalized mathematical reasoning is grounded in perceptual and performance schemas rather than arbitrary symbolic codes, perhaps we should consider the ways in which such schemas can be used in professional descriptions of desired and actual computing and information artifacts rather than reducing such natural descriptions to a schema-neutral logic.

Although the paradigm of literally interpreted symbolic representation is deeply rooted in software engineering and information technology research and practice, it is hard to get away from the metaphor that some abstract phenomenon is a physical object or region with geometric, mereological (part/whole) and topological (connectivity) structures. Consider the following examples (to name but a few):

- the separation between system and environment, or machine and domain [7];
- the state “space” of a system’s behavior;
- the network of goals relevant to a system [19];
- the web of agency and interdependent responsibilities in which a system operates [21];
- the sequencing of episodes within scenarios [14].

In all these approaches, the depiction of states as places, of processes as vertically extended structures, of classifications as trees, and so on, is merely a presentation of a more formal model. Even in requirements methods inspired by a contextual or naturalistic rather than a strict abstractionist or representational world view [15, 16], we find several representations of abstract phenomena such as Contextual Inquiry/Design’s flow, sequence, and culture models [2] which lack a formal semantics but which also rely on embodied mappings. In the cultural model of Contextual Inquiry, for example, organizational conventions or mores are depicted as extended amorphous objects pressing down on the hapless stakeholder, a mapping from tacit social awareness through a visual depiction of size and proximity to basic tactile or kinesthetic experiences.

³ The text (“shift”) illustrates the pervasiveness of physical metaphors in discourse about ideas. The reader is invited to pick any page in these proceedings and find further examples.

Furthermore, many abstract concepts from ontologically intransigent areas of experience also appear in reasoning about software and information appliances. These include such apparently philosophically elevated ideas as mind, self, agency and morality. In contrast to the success of first-order truth-functional formal systems for reasoning about context-free truths, such as those found in the axioms of arithmetic, propositional attitudes such as knowledge, desire, and obligation have not proved amenable to formalization in modal logics or economic models of choice. And yet abstract concepts like knowledge, preference, autonomy, norms and policies, are difficult to dispense with when thinking about information technology.

For example, we do not need fancifully to impute moral attitudes onto technology for issues concerning right action, obligation, responsibility, the absoluteness or flexibility of policy, and norm-following behaviors to become relevantly problematic. Any system designed to enforce some explicit organizational policy becomes a rule-following and thence moral agent. And we find it extraordinarily difficult not to talk about such systems and system components without recourse to concepts like agency, priority, intention, norms and fairness. We do not need to mean by these language uses, although one day perhaps we may, that a designed technology is morally significant in the same way as a person. But it is naive to argue to the contrary that such uses are mere figures of speech. They color the concepts that we can subsequently employ and the inferences that we can or choose to draw from them.

Similarly, in an architecturally distributed system, a federation of system components or a community of agents, issues of who knows what, what each component can do, and even what each is trying to accomplish become first-class concepts to reason about. Just as we habitually think about states and change in terms of physical location and motion, so it is almost impossible to conceptualize component interaction except in terms of anthropomorphic metaphors in which the components or agents are person-like to the extent that we can talk coherently about their competence, knowledge and motives.

Thus, there appear to be at least three types of design-relevant metaphorical mappings that affect our descriptions of IAs in requirements engineering: Metaphors that reify or anthropomorphize abstractions as (1) physically extended and bounded objects, (2) regions of space, and (3) mental or communicative processes or agents of change. All three kinds are pervasive and subtle in their reach. We discuss two examples of each category.

2 Reification: Abstraction as Object

Consider the following perceptuo-motor mappings proposed by Lakoff and Johnson:

(1) If importance is mapped to size, the important object must be thought of as a big object. Big objects are generally heavier than small objects; you cannot see around them as easily; they are more difficult to miss and are easier to see in the distance. Objects that are big are big for everyone; so importance, when mapped onto size, does not lend itself naturally to relative judgments. However, big objects far away may seem no bigger than small objects up close, so different judgments of importance could be construed as differences in position or the basis of comparison. Some requirements are more important than others and are therefore metaphorically more massive and burdensome.

(2) If “more” is mapped to “up”, the something that there is more of must be stackable and not compressible. Quantities of this kind usually have an inherent floor or zero value, but do not necessarily have a maximum or ceiling unless artificially constrained by a container (i.e. extrinsically by something else). Priorities among requirements are frequently described through sorting, alignment and ranking.

(3) If purposes are desired objects, then they can be manipulated the same way that any objects can be. They may be grasped or relinquished, traded or stolen. Unlike size, desirability is in the eye of the beholder, and there is no need for a given purpose to be shared. Perhaps they can be divided or segmented into smaller objects that still have value, or perhaps they cannot. An object remains the object it is unless something happens to it; its identity and unity does not depend on whether we think or care about it. Viewed as an object, a purpose endures without the need for constant attention. This is a different view of purposes than goals as points in a dependency network.

(4) If knowing is grasping, then an object of knowledge is a object that may be grasped or let go of. Because objects are physically bounded, and grasping encloses a region of space, you cannot grasp something independently of somebody else grasping it; either you hold it on your own, you have to collaborate to hold it because it is bulky or heavy, or you are struggling for it. Viewed as grasped objects, knowledge is difficult to share or disseminate. It is more like a possession or item under one’s exclusive control. We will encounter an example of the Knowing Is Grasping mapping below when considering information security and privacy requirements.

3 Spatialization: Abstraction as Place/Path

Many of the metaphors that Lakoff and Johnson discuss are rooted in spatial experience, either in the notions of locations or regions of seen or felt space or in the notions of trajectories or positions along paths.

3.1 Classification as Placement

One of the most basic spatial mappings is that Categories are Containers. This of course is the basis of

simple visualizations like Venn diagrams or the enclosure semantics of Higraphs. But the geometric and topological constraints of physical space impose strong constraints on how concepts may be placed in spatialized categories. First, a container is a closed region with a well-defined boundary. Thus an object in the sense of object-oriented analysis, when viewed as a physical object is either an instance of a given class or it is not. It is possible for an object to have many classes if its container (class) is nested inside others. It is also possible for containers to overlap, but this requires interpenetrating boundaries, which we do not encounter in everyday experience of physical containment, and hence multiple inheritance gives rise to a weakened sense of classification as placement.

Philosophers, linguists and communications researchers have pointed out the problems with rigid category boundaries [3, 20], but most writers on the foundations of requirements engineering, from 1960s information systems analysis down to the object-oriented analysis and ontology modeling methods of today still cling to the Category is a Container metaphor.⁴

3.2 Features/Services as Occupancy

Not only can the objects of an information artifact’s subject matter be regarded as objects placed in fixed containers: the features or services can be regarded as spatially situated also. This mapping is behind the notion that an information artifact fits into “its” problem domain. But where is the problem domain? It is taken to be a frame of reference independent of the artifact itself with spatially coherent internal boundaries. Thus we could imagine a map of domain space in which the domain of “word processing” contains “paragraph formatting” much as the USA contains the state of Georgia. Word processing is a neighbor to “page layout” (or perhaps an overlapping region, sharing some common subdomains). Word processing and page layout are closer to each other and to “presentation preparation” than they are to “payroll”, although payroll is close to (or overlaps with) “taxation.”

Raymond [17] uses the term “noosphere”⁵ to spatialize the abstract concept of a space of possible applications or uses of IT as a kind of territory. He develops a telling and elaborate metaphor to explain the dynamics of application evolution in new application territories. Simply by moving into, colonizing or “homesteading” a region of the noosphere, a program stakes a claim to that region and makes it much more

⁴ A notable exception is Jackson [1995].

⁵The term noosphere was introduced by the theologian Teilhard de Chardin to describe the putative realm of the human spirit (in contrast to the biosphere). Raymond’s more prosaic use of the term names the “realm” of information artifacts that distinguishes it from physical reality but which nevertheless connotes a *region* of experience.

difficult for competing programs that come along later. Homesteading is of course a figurative metaphor, but underneath this is the more basic metaphor of domain applicability as occupancy of a region of space, a region that may contain, overlap, and be near to or far from others. And this is the pervasive metaphor of domain analysis: that the space being carved up into regions really exists independently of the technologies that occupy it, that a specific feature or domain concept is either in a sub-region or it is not.

The central assumption underlying all location- and trajectory-based metaphors is that the frame of reference, the space, that the objects of interest occupy and in which they move is itself relatively fixed and stable over the timescale of interest for the development and evolution of artifacts. Enhancing the features of an application program is viewed as growing its boundary to “annex” nearby domain regions (as, for example, when a word processor is augmented with new page-layout requirements). An application’s requirements are complete if they occupy all of a relevant domain and they are “bloated” or involve “gold plating” if they occupy too large or diffuse a region or have tendrils of functionality extending into comparatively unrelated domain regions. Conversely, the development of the artifact itself does not affect the topography of the underlying domain.

But the domain-as-region metaphor breaks down if we allow information artifacts and their contexts or environments to co-evolve over approximately the same timescale. The domain of giving and viewing presentations did not exist as a stable region of the “noosphere” waiting for Powerpoint and other applications to colonize it. What we think of as a presentation today is partly determined by our past experiences of Powerpoint use. Applications change their use contexts just as use contexts determine the requirements for and evolution of artifacts. Thus domains are not stable frames of reference for product families and information artifacts but co-evolve with them and are partly defined by technology features.

3.3 Security/Privacy as Spatial Constraints

Security and privacy policies provide a fertile area for investigating how tacit metaphors affect how rights and obligations over information are construed (see also [1]):

Intellectual Property: Commerce in Facts.

Information is a stuff that can be bought, sold and stolen. When I own it, I have it. When you buy it, you have it. If something is mine, I can do what I want with it within reason.

This metaphor valorizes the role of the owner of the information and ignores the logically independent role of information subject. You do not necessarily own information that concerns you. The intellectual property metaphor therefore devalues privacy and accuracy as it affects the information subject.

The Magic of Names: Identifiers as Surrogate Entities.

Names are sometimes surrogate objects that have as much value as the object itself. “Identity theft” is a peculiar concept when examined closely. The term suggests not just that one’s identifier may be misused but that in a real sense it may be stolen and therefore no longer be yours.

Sunshine and Shadow: Disclosure and Secrecy.

Information can be seen or not depending on how clear the view is and whether it is hidden. This metaphor emphasizes the desired obscurity or hiddenness of sensitive information. Information misuse is now not stealing but snooping. Firewalls and similar security features act as camouflage.

Rights and Obligations: Commitment as Force.

An agreement not to disclose personal information is conditional on one’s continued ability to honor the obligation. This is not feasible if that ability is preempted by change in ownership (e.g. a takeover, or bankruptcy) or when overruled by legal action such as a court subpoena. This metaphor therefore treats privacy and security policies as forces that act on information. Sometimes one policy will overpower others so that they are no longer applied.

4 Anthropomorphism: Mechanism as Mind

A third major group of metaphors in requirements descriptions are anthropomorphisms. These involve the positing of a kind of homunculus or agent, or purposeful machine, as an operational definition of or locus for teleological responsibility. The language of early symbolic AI has permeated computing with such frozen metaphors. Operating systems have “demons”, user interface toolkits provide “listeners”, distributed systems involve the coordination of “bots” or “agents”. Processes “listen” on channels, issue “requests” or “commit” to actions. How much of this is a figure of speech, and how much does it reveal an underlying anthropomorphic frame of reference? According to the view explored in this paper, these are not mere figures of speech but reflect a deeply rooted need to refer to elements of mindlike or personlike responsibility in the description of a complex artifact. Hsi and Potts [5] refer to these units of purposeful function as “teleons” to distinguish them and the principles that govern their individuation and composition from corresponding principles for software implementation or user-interface components. Thus the “spell checker” is a well-delineated teleon concerned with the checking of spelling in a text processing artifact such as a word processor. What makes the spell checker a teleon, that is a delineated unit of purpose at the requirements level, needs to be formalized more carefully than Hsi and Potts do, but certainly the fact

that there is some C++ code called “spell_check.cpp” is not part of that definition.

4.1 Anthropomorphic Problem Frames

As an illustration of the pervasiveness of anthropomorphism in requirements engineering, consider the concept of problem frames [7], which are canonical patterns of subproblem that appear in describing the requirements for information artifacts. Each problem frame or concrete subproblem description involves a “machine”, an abstraction of a planned responsibility to be allocated to the information artifact being specified. Its responsibility is always stated with respect to the state of one or more real-world domains, with the attitude toward this state being variously creating it (in the case of a workpiece problem such as document editing), truthfully reflecting its state (in the case of an information display problem such as monitoring temperatures or responding to queries about an enterprise), bringing some state about (in the case of a required or commanded behavior problem, such as traffic light control), mapping from one domain to another (in the case of a transformational problem such as language translation or data analysis), or bridging from one domain to another (in the case of a connection problem such as telephony call management).

In our terminology, Jackson is essentially claiming that there are a number of basic types of teleon, such as workpiece composers, information display oracles, behavior controllers, symbol transformers and resource matchmakers. A complex artifact can be decomposed into a superposed set of such homunculi, some of which operate at a meta-level with respect to others (e.g. a security intrusion detection teleon that is an oracle for the state of an underlying connection teleon, or a customization teleon that is a composer of workpieces used by an underlying workpiece or controller teleon to modulate its behavior).

Jackson avoids obvious anthropomorphism by calling the requirements-implementing artifacts “machines”, but he means by this abstract or virtual machines that may exist notionally only for the purpose of orthogonal description and that may not exist as delineated architectural units in the implementation of the composite artifact (machine) at all. But a gauge (machine) is very similar in its implications to an oracle (homunculus), a workpiece editor (machine) is a mechanized amanuensis (homunculus), a controller or governor (machine) is an automated operator (homunculus), a translator can be either mechanical or human, and a connection or switch (machine) is a kind of matchmaker (homunculus). Indeed, when one starts to consider more elaborate variants or flavors of the basic problem frames, the machines become even more agent-like. For example, a controller for a “biddable” domain (i.e. one that is autonomous and that the machine can influence but not control) is more like a reminder or a nag than an operator. All it can really do is cajole and monitor whether its recommendations have

been followed. The requirements of the ensemble formed by such an information artifact and the biddable entities (generally people or groups) that it is intended to influence can only be met if the non-mechanical part of the ensemble behaves within some constraints.

4.2 Anthropomorphic Services

Jackson and Zave [8] have proposed an architecture for telecommunications systems that maps “features” as telephony professionals generally understand them onto well-delineated architectural components. In Jackson’s terminology, there is a clear correspondence between software components or subsystems and problem-level virtual machines. This proposal must be seen as an empirically verifiable design hypothesis: that modularizing the implementation similarly to the modularization of the requirements leads to effective (e.g. resource-efficient, easily modifiable) implementations. It is not a definition of what a feature necessarily is.

Nevertheless, many systems, particularly those that inhabit information-rich, event-dominated domains, such as electronic mail, can be thought of as confederations of feature-providing teleons. Features are not just something that the system provides; they are the components it is made from. Thus e-mail can be construed as an overlapping set of teleons or anthropomorphic roles such as anonymous remailer, mail lists, the address book, the message arrival listener, and so on [4].

5 Discussion and Conclusion

We have shown a number of very different ways in which requirements description is perfused with metaphor in the embodied cognition sense of the term employed by Lakoff and Johnson. Given the richness and pervasiveness of perceptuo-motor and anthropomorphic schemas lurking under the surface of requirements descriptions, it is difficult to maintain an assurance that requirements documents are symbolic representations of desired properties in schema-neutral ontology. Our language and basic cognitive mechanisms and predispositions force us to describe required behaviors and properties in object-like, place-like, path-like and ultimately mindlike terms. Artifacts are designed to help customers reach goals and overcome obstacles, they are designed to follow paths of operation that branch and loop, their behavior is divided into modes or regions of relevant circumstances, some of which are central to their purpose and some peripheral, they make decisions, communicate with users, monitor, control and allocate.

One way to interpret these claims is to acknowledge the need for better modes of communication with customers and other non-specialists, to make specification languages and requirements analysis and animation tool interfaces more palatable to non-technical readers and viewers. This argument makes use of a distinction

between performance and competence. Humans as writers and readers of system descriptions and as stakeholders in the systems described therein have bounded rationality that limits their full understanding of a formal, symbolic representation. The formal semantics of a specification represents the normative meaning that an ideal reader could grasp in principle. Metaphors suggest ways to translate the authoritative specification into more accessible, local descriptions.

But this misses the more significant challenge that embodied cognition raises for requirements engineering and applied computing generally; that to describe an information artifact is not to represent its purpose in a symbolic code but is in some sense to enact the perceptuo-motor or interpersonal communicative schemas that give the artifact's purpose its meaning. Far from being a dilute translation of a formal, symbolic description into a fanciful or even gratuitously confusing rendition, according to this perspective descriptions that exploit these metaphorical mappings convey the only meanings that a specification can have. This is a radical and untested suggestion, but it deserves serious attention in future research and practice in the specification of application requirements, information usage policies, and infrastructure standards.

Acknowledgments. The author wishes to thank several anonymous reviewers for their constructive suggestions.

6 References

- [1] A. Anton, J.B. Earp, C. Potts and T. Alspaugh, The Role of Policy and Stakeholder Privacy Values in Requirements Engineering, *Proceedings RE'01: International Symposium on Requirements Engineering*, Toronto, Ontario, 2001 [these proceedings].
- [2] H. Beyer and K. Holzblatt, *Contextual Design: Defining Customer-Centered Systems*. San Francisco, CA: Morgan-Kaufmann, 1998.
- [3] G. C. Bowker and S. L. Star, *Sorting Things Out: Classification and its Consequences*. Cambridge, MA: MIT Press, 1999.
- [4] R.J. Hall, How to Avoid Unwanted Email, *Comm. ACM*, March 1998.
- [5] I. Hsi and C. Potts, Studying the Evolution and Enhancement of Software Features, *Proceedings International Conference on Software Maintenance*, San Jose, CA: IEEE Computer Society Press, 2000.
- [6] M. Jackson, *Software Requirements and Specifications*. New York, NY: Addison-Wesley, 1995.
- [7] M. Jackson, *Problem Frames: Analyzing and Structuring Software Development Problems*. New York, NY: ACM Press/Addison-Wesley, 2001.
- [8] M. Jackson and P. Zave, Distributed Feature Composition: A Virtual Architecture for Telecommunications Services, *IEEE Transactions on Software Engineering*, vol. 24, pp. 831-847, 1998.
- [9] M. Johnson, *Moral Imagination: Implications of Cognitive Science for Ethics*. Chicago, IL: University of Chicago Press, 1993.
- [10] G. Lakoff and M. Johnson, *Metaphors We Live By*. Chicago, IL: University of Chicago Press, 1980.
- [11] G. Lakoff, *Women, Fire, and Dangerous Things: What Categories Reveal about the Mind*. Chicago, IL: University of Chicago Press, 1987.
- [12] G. Lakoff and M. Johnson, *Philosophy in the Flesh: The Embodied Mind and its Challenge to Western Thought*. Basic Books, 1999.
- [13] G. Lakoff and R. E. Nunez, *Where Mathematics Comes From: How the Embodied Mind Brings Mathematics into Being*. Basic Books, 2000.
- [14] C. Potts, ScenIC: A Strategy for Inquiry-Driven Requirements Determination, *Proceedings RE'99: International Symposium on Requirements Engineering*, Limerick, Ireland, 1999
- [15] C. Potts and I. Hsi, Abstraction and Context in Requirements Engineering: Toward a Synthesis, *Annals of Software Engineering*, vol. 3, pp. 23-62, 1997.
- [16] C. Potts and W. C. Newstetter, Naturalistic Inquiry and Requirements Engineering: Reconciling their Theoretical Foundations, *Proceedings RE'97: Third IEEE International Symposium on Requirements Engineering*, Annapolis, Maryland, 1997
- [17] E. Raymond, *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly Linux, 1999
- [18] M.J. Stefik and V. Cerf, *Internet Dreams: Archetypes, Myths, and Metaphors*, Cambridge, MA: MIT Press, 1997
- [19] A. van Lamsweerde, R. Darimont, and E. Letier, Managing Conflicts in Goal-Driven Requirements Engineering, *IEEE Transactions on Software Engineering*, vol. 24, pp. 908-926, 1998
- [20] L. Wittgenstein, *Philosophical Investigations, Second Edition*. Oxford, UK: Basil Blackwell, 1958.
- [21] E. Yu, Towards modeling and reasoning support for early phase requirements engineering., *Proceedings RE97: Third International Symposium on Requirements Engineering*, Annapolis, Maryland, 1997.