CSAMOA: A Common Sense Application Model of Architecture

Jason B. Alonso
Tangible Media Group
MIT Media Laboratory
20 Ames St E15-354
Cambridge, MA 02139 USA
+1-617-452-5617
jalonso@media.mit.edu

ABSTRACT
It has become apparent that many human-computer interface applications of common sense reasoning, particularly those built on theOpenMind Common Sense corpus , make use of similar computational tools (spreading activation, for example) in addition to the corpus itself. Meanwhile, new representations, new methods of reasoning, and new applications are being introduced without a clear foundation for understanding their interrelationships. In this paper, I describe my goals and progress in the design of a model of architecture for expressing theinter-operation of common sense tools developed as parts of different efforts.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Theory and Methods.

General terms: Standardization, Theory

Keywords: Common sense, application programming interface, software architecture

INTRODUCTION
It has become apparent that many applications of common sense reasoning built on the OpenMind Common Sense (OMCS) corpus [9] make use of similar computational tools. In particular, most of these applications make use of semantic networks and perform most of their user interface magic using a small set of operations on these semantic networks. These reasoning tools, however, are usually intimately intertwined with their applications or are part of a monolithic software library,ConceptNet [6] , which is a rigid framework from a software engineering perspective. In either case, new applications are bound to the development of new techniques, which hinders the research and development of both.

These tools are diverse, including such disparate components as raw English language corpora and spreading activation algorithms. Meanwhile, new representations, new methods of reasoning, and new applications are being introduced without a clear foundation for understanding their interrelationships. There also exists common sense knowledge representations that compete with semantic networks. As the field of intelligent user interfaces using common sense reasoning advances from isolated experiments toward deployed applications, the need for a standard model of architecture will become ever greater. With this in mind, I set forward this draft of a Common Sense Application Model of Architecture (CSAMOA).

CSAMOA divides the software architecture of common sense applications along conceptual lines, permitting concise discourse on the roles and contributions of any given common sense reasoning tool as well as the development of standard Application Programming Interfaces (APIs) along conceptual boundaries.

Evolving common sense tools
The OMCS family of common sense tools has been evolving and is continuing to evolve, leaving behind the applications that have built upon it. In Lieberman andLiu's Annotation and Retrieval Integration Agent (ARIA) [3], a photograph annotation and retrieval system is built upon Commonsense Robust Inference System (CRIS), a semantic network system that served as a precursor to theConceptNet platform [6]. CRIS was superseded by OMCSnet , which is used in GOOSE, a goal-oriented search engine assistant [4] [5]. Eventually, a suite of knowledge tools were integrated into a single library known asConceptNet [6].

Future directions of the use of the semantic network, as derived from OMCS, include OpenMind Commons [10]. Interestingly, this upcoming project, as proposed, reuses many of the ideas in ConceptNet, but the new structure focuses on using inference in real time, which contradicts the ConceptNet design choice of batch processing all assertions.

Competing knowledge representations
At the present, there is a remarkable spectrum of substantially incompatible tools in the space of common sense reasoning. These include the OpenMind Common Sense project [9], which uses natural languages as knowledge representation, and the fundamentally more ontological projects Cyc [1] and ThoughtTreasure [7]. Of the intelligent user inter-

face applications of common sense surveyed by Lieberman et al in [4], most are based on the OpenMind family of platforms, one interesting exception is Common Sense in a Disk Jockey's Assistant (CSDJ), which made use of ThoughtTreasure instead.

An Analogy with the Development of Informatic Networks

The early development of computer networks, both experimental (like ARPANET) and proprietary, lacked an established "interconnection architecture" to unify diverse pieces of computing equipment. The International Standards Organization (ISO) formally recognized this in 1977 and created a subcommittee known as "SC16" for "Open Systems Interconnection." The subcommittee established, as its highest priority, "the development of a standard Model of Architecture which would constitute the framework for the development of standard protocols." The product of this discussion was the "Reference Model of Open Systems Interconnection," also known as the "OSI Model." [11] This model organizes the entire gamut of communications protocols, from cabling standards to object serialization.

The history of interfaces built on the OMCS platform has clearly demonstrated that many of the components of common sense reasoning platforms are still being developed. Indeed, theOpenMind Commons knowledge elicitation platform is expected to make radical departures from the previous platforms. Furthermore, there exist resources other thanOMCS for common sense, like Cyc and ThoughtTreasure, that have the potential to be useful in the same applications, but there is a great disconnect among Cyc, ThoughtTreasure, and OMCS.

As it stands, the development of new applications is being hindered by the limitations of lack of architecture underlying the software tools for using common sense.

DESIGN CRITERIA

The following design goals were borrowed/adapted from the goals leading to the OSI Model [11]:

1. do not create so many layers as to make difficult the system engineering task describing and integrating these layers;
2. create a boundary at a point where the services description can be small and the number of interactions across the boundary is minimized;
3. create separate layers to handle functions which are manifestly different in the process performed or the technology involved;
4. collect similar functions into the same layer;
5. select boundaries at a point which past experience has demonstrated to be successful;
6. create a layer of easily localized functions so that the layer could be totally redesigned and its protocols changed in a major way to take advantages of new advances in architectural, hardware, or software technology without changing the services and interfaces with the adjacent layers;
7. create a boundary where it may be useful at some point in time to have the corresponding interface standardized;
8. create a layer when there is a need for a different level of abstraction in the handling of data, e.g., morphology, syntax, semantics;
9. enable changes of functions or protocols within a layer without affecting the other layers;
10. create for each layer interfaces with its upper and lower layer only;
11. create further subgrouping and organization of functions to form sublayers within a layer in cases where distinct communication services need it;
12. create, where needed, two or more sublayers with a common, and therefore minimum, functionality to allow interface operation with adjacent layers; and
13. allow bypassing of sublayers.

THE CSAMOA MODEL

At the time of this writing, my working draft of the stack has four layers: application, realm, representation, and corpus.

Corpus

The first layer of the CSAMOA stack is the Corpus layer. It is the most basic component of the CSAMOA stack and is dedicated to preserving the original, human representation of common sense knowledge, irrespective of any particular application, and any notations required to preserve source information. This includes, for example raw natural language statements, elicitation frames, and source data (including user profiles) on the stored natural language statements.

It is important that much emphasis is placed on preserving accountability and human-readability for knowledge in this layer. This emphasis is in place to allow for ready debugging regardless of the knowledge representation (KR) in place– this, in turn, is an attempt to address the fundamental limitations of any KR as described in [2].

In the OpenMind family of projects, this role is filled by the OpenMind Common Sense corpus itself. In systems that have knowledge representations other than natural language, like the maps and other representations from Eric Mueller'sThoughtTreasure [7] , this layer is inclusive of these representations. One can even consider a body of CycL assertions and appropriate navigational routines to comprise the Corpus.

Representation

The Representation layer is the second and most structurally complicated layer in the CSAMOA stack. This layer is dedicated to the abstraction of a particular knowledge representation into a machine-interpretable form. The defining attribute of this layer is the form it gives to the knowledge it contains. Formally, this means that this layer defines both data structures and APIs for the computational manipulation and/or navigation of common sense knowledge. By virtue of

design goal 12, I subdivide this layer intosublayers to allow a particular representation layer to be built upon a variety of corpora. These sublayers include, in rising order: Parsing/Encoding, Reasoning, and Presentation.

In the case of the OpenMind family of projects, the Representation layer would have been CRIS, OMCSnet, or ConceptNet [6]. In my following discourse on the sublayers of the Representation layer, I shall show how ConceptNet specified the most developed set of interfaces out of these three examples.

Presentation The Presentation sublayer defines the theoretical form of the Representation layer, providing the data structures for the machine representation of knowledge as well as a set of interfaces for navigating within the machine representation of the connected Corpus layer, including the creation of working contexts. It should be expected that not all knowledge represented by the Presentation sublayer should come from the layer. This is to say that individual applications of the Presentation layer may introduce knowledge that is not preserved by the Corpus layer.

It should be noted that CRIS, OMCSnet, and ConceptNet are all semantic networks, which would be a valid specification of the data structures in Presentation sublayer. Navigational interfaces may include methods for finding optimized paths between nodes in the semantic network or for filtering for nodes and/or relationship types germane to a particular application domain.

Reasoning The Reasoning sublayer is responsible for refining knowledge for the Presentation layer and for deriving new pieces of knowledge, particularly with abductive reasoning, from the existing body of knowledge. Note that this can result in the creation of knowledge that is not preserved by the Corpus layer. The knowledge used by the Reasoning sublayer may come from the Corpus, the application by way of the Presentation sublayer, or from both.

Upon close examination of CRIS, OMCSnet, and ConceptNet, it can be seen that these projects were distinguished to some extent by the number of relationship types between network nodes and the degree to which reasoning was used to generate succinct representations. Of these examples,ConceptNet was the most advanced for introducing various forms of klines in what would be the Reasoning sublayer.

Parsing/Encoding The Parsing/Encoding sublayer is responsible for taking the knowledge from the Corpus layer and converting its form to comply with the rest of the Representation. As there may be cases where knowledge must be converted back into a Corpus-compliant form or into natural language, this layer is responsible for performing this encoding operation as well.

ConceptNet directly included MontyLingua [6], a natural-language processing engine for English, filling the role of the Parsing/Encoding sublayer.

Realm

The Realm layer is the most vaguely defined component of the CSAMOA stack: it is responsible for translating the knowledge from the Representation layer, or otherwise perform operations on it, to make it useful from the perspective of application development. Like the Presentation layer of the OSI model [11], design goals 3 and 4 demand that a variety of general-purpose routines be collected into a layer just below the Application layer.

This is probably best understood with the simple example of spreading activation. In ARIA as well as Shen, Lieberman, and Lam's *What Am I Gonna Wear?* scenario-oriented fashion recommendation system [8], spreading activation is applied to a semantic network. The parameters used to guide spreading activation differ in these two applications...

In particular, operations such as spreading activation and realm filtering, wherein certain relationship types in a semantic network are filtered out or otherwise weighted to give them critical nuances for other operations (especially spreading activation), belong in this layer to benefit multiple applications. It should also be noted that some operations, like realm filtering, are conceptual expansions of navigational interfaces from the Representation layer.

Application

At the top of the CSAMOA stack is the application itself, which is responsible for all user interactions and for the ultimate processing of the knowledge made accessible by the rest of the hierarchy. This layer defines the experience of the user when interacting with an intelligent user interface, insofar as the processing of natural language and common sense can be left to the lower layers.

FUTURE DIRECTIONS

CSAMOA is still a work in progress. Next, I expect to develop a set of interfaces to fill each of these roles. At that point, I shall adapt existing implementations ofConceptNet and/or OpenMind Commons to fit these interfaces, demonstrating the usability of this model of architecture.

REFERENCES

1. Inc. Cycorp. How does cyc reason? Cyc web site, 2006. http://www.cyc.com/technology/technology/whatiscyc_dir/howdoescycreason.

2. Randall Davis, Howard Shrobe, and Peter Szolovits. What is a knowledge representation? *AI Magazine*, 14(1):17–33, 1993.

3. Henry Lieberman and Hugo Liu. Adaptive linking between text and photos using common sense reasoning. In *Conference on Adaptive Hypermedia and Adaptive Web Systems*, LNCS 2347. Springer, 2002.

4. Henry Lieberman, Hugo Liu, Push Singh, and Barbara Barry. Beating common sense into interactive applications. *AI Magazine*, 25(4):63–76, 2004.

5. Hugo Liu, Henry Lieberman, and Ted Selker. Goose: a goal-oriented search engine with commonsense. In *Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*, LNCS 2347, pages 253–263. Springer, 2002.

6. Hugo Liu and Push Singh. Conceptnet: a practical commonsense reasoning toolkit. *BT Technology Journal*, 22(4), 2004. Kluwer Academic Publishers.

7. Erik T. Mueller. Thoughttreasure: a natural language/commonsense platform. Signiform web site, 1998. http://www.signiform.com/tt/htm/overview.htm.

8. Edward Shen, Henry Lieberman, and Francis Lam. What am i gonna wear: Scenario-oriented recommendation. In *IUI '07: Proceedings of the 12th International Conference on Intelligent User Interfaces*. ACM Press, 2007. Submitted.

9. Push Singh. The public acquisition of commonsense knowledge. In *Proceedings of AAAI Spring Symposium: Acquiring (and Using) Linguistic (and World) Knowledge for Information Access*. AAAI, 2002.

10. Rob Speer. Openmind commons: A new framework for acquiring common sense knowledge. M.Eng. thesis proposal, 2006. http://torg.media.mit.edu/rob/index.php/Research.

11. Hubert Zimmermann. Osi reference model–the iso model of architecture for open systems interconnection. In *IEEE Transactions on Communications*, volume 28, issue 4, pages 425–432, April 1980.