The Cyc Blackboard System

Blake Shepard, Ph.D., Doug Lenat, Ph.D., Michael Witbrock, Ph.D.

Cyc's knowledge base and inference engine can efficiently carry out the functions normally comprising a Blackboard System. This document provides an accounting of what such blackboard system features are (Section 0), and how they are supported by extant Cyc system functionalities (Sections 1-3).

0. What's in a "Blackboard"?

A blackboard system consists of three major components:

(1) A **repository** (which one can think of as the blackboard itself) which is capable of storing problems along with contributions toward problem solutions. Often, there are indices into the blackboard; a simple example would be to define a meaning for the x and z axes of a physical blackboard, such as x representing the flow of time and z representing level of abstraction. That additional layering means that knowledge sources (see item 2, below, in this list) may be able to focus their attention on just some region of the blackboard, and may post their contributions (often solving some posted sub-problem, or posting a new one) at a particularly appropriate (x,z) point on the blackboard. This two-dimensional coordinate space was just an example of such an index structure; in principle it could be far richer – and in Cyc's case, it is (see Section 1, below).

(2) **Software specialist modules** ("knowledge sources" (KSs)) which embody diverse and independent techniques for contributing to problem solutions. Typically such knowledge sources will monitor some regions of the blackboard (often supported by an alerting mechanism in the blackboard system), notice postings, respond to some of those, and in the process post new items on the blackboard. For example, a module might start to work on a posted sub-problem, at some point posting a series of sub-sub-problems that are blocking it from further progress, and, when some KSs post the solutions to *those*, this KS might notice that and begin to work again. In many cases, there is some pattern which determines when a knowledge source is relevant. The KS may ask for, or seize, exclusive rights to work on one of the posted sub-problems, possibly for some time period, or there might be multiple KSs working on the same posted sub-problem without even being aware of it – to govern that chaos, it is sometimes useful to have a third component:

(3) [optional] A **control mechanism** which facilitates and rides herd over the interactions between (1) and (2). It is optional because in some applications, the knowledge sources asynchronously monitor, post, and respond to postings, without any need for such a

control. E.g., consider the way EBay works, or a typical WIKI, or the Internet itself, where we humans are the knowledge sources, and the control is heterarchical to say the least.

The following three sections describe the major Cyc components that correspond to these three major components of a blackboard system.

1. The Cyc Blackboard Itself

The blackboard itself, component (1), above, is realized in Cyc by the Cyc Knowledge Base.

The Cyc KB/Blackboard is divided up into contexts (microtheories); the dozen dimensions of context-space induce a multidimensional structure on the Cyc KB/Blackboard. One of those dimensions (predicates interrelating two contexts) does correspond to time, and one does correspond to level of abstraction, but all dimensions are potentially useful, powerful, and relevant, depending on the problem being attacked. Most of the relation between contexts are transitive and hence correspond to the naïve intuitive notion of a hierarchy and of a "dimension" of the blackboard.

External knowledge sources need to be able to post on the blackboard: sub-problems, sub-problem solutions, constraints on the solution, meta-level information, observations, etc. The message may be posted globally, but more often the application derives nontrivial power and efficiency from having the knowledge sources only attending to a tiny sliver of the entire blackboard. That sort of "posting a message at a certain place on the blackboard" is achieved by having an external knowledge source assert something to Cyc in a certain context, via an API (which could be Cyc's powerful but idiosyncratic API, but we can also support the emerging SPARQL/Update standard, given that the poster knows which Cyc ontological terms to use, and the assertions fit within the RDF data-model). Similarly, Cyc supports Java and Lisp APIs, and SPARQL and REST-WebService query access to KB/Blackboard content.

Using a KB supported by a powerful inference engine adds significant functionality to the Blackboard itself. Cyc contains numerous heuristic level (HL) modules that support logical inference over KB content, and they (and KB-stored rule sets) can interact with the KB-as-Blackboard as though they were external knowledge sources, responding to postings and adding new ones.

Via simple API functions, new contexts can be added to the KB/Blackboard (entirely new ones, or ones which extend or restrict one or more already-existing contexts) and existing contexts can be removed from the KB/Blackboard (sometimes merged with other contexts), and individual assertions can be added, removed, or edited from the KB/Blackboard contexts. Because Cyc supports full dynamic update with truth-maintenance, removing postings or even entire contexts/regions from the blackboard is straightforward and behaves correctly.

Arbitrary objects (such as Unicode strings, images, compiled code, etc.) can be embedded in the KB/Blackboard assertions, or connected via assertions referencing file locations, URLs, or other identifiers.

2. External Knowledge Sources

Independent modules can act as knowledge sources on the Cyc KB/Blackboard in three ways:

- First, external modules can register KB/Blackboard event listeners via Cyc's Java API. Once registered, a KB/Blackboard event listener will produce a notification based on an arbitrarily complex logical specification of KB/Blackboard changes. Listeners can be registered to monitor changes to contexts, terms, or collections in the KB/Blackboard. Notably, a listener can be registered on a very general term, collection, or context; and Cyc's subsumption reasoning can be called upon to trigger a KB/Blackboard event whenever there is a change to a more specialized term, collection, or context. External modules can, of course, react to KB/Blackboard events by writing to the KB/Blackboard via the Cyc API or whatever protocol layer (e.g., SPARQL/Update) has been agreed upon.

- Second, external knowledge sources can be made available to Cyc inference via "Semantic Knowledge Source Integration" (SKSI). Once its schema has been explained to Cyc (mapped into Cyc via SKSI), all of the data in an external source is seamlessly available to Cyc (i.e., it becomes virtually part of the Cyc KB/Blackboard. Typically, this method of connecting external modules to Cyc makes sense for knowledge sources whose schemas are relatively static, such as SQL databases and RDF triple stores and stable websites (such as the US Weather Service website, for obtaining forecasts).

- Third, external modules can be rigidly connected to Cyc's internal inference modules. During Cyc inference, a problem can be solved through appeal to a rule, or by application of a dedicated inference module that can directly prove the problem true or false or provide a binding for a variable. In the simplest cases, such dedicated inference modules in Cyc appeal, for example, to natively implemented math functions. In more exotic cases, dedicated inference modules could call out to the results of tests conducted in wind tunnels or particle accelerators. Knowledge sources can also be semi-rigidly attached, by embedding custom adaptors as inference modules that communicate with the knowledge source over a high-speed or low-overhead channel.

These means of connection between knowledge sources and the KB/Blackboard are in addition to the internal reasoning modules that support inference over Blackboard-stored knowledge. Cyc currently maintains over 1000 specialized reasoning modules used to efficiently support a wide variety of specific kinds of logical inference; these comprise its HL (Heuristic Level).

3. The (optional) Control Mechanisms

A number of control mechanisms exist that can affect the way external knowledge sources interact with the Cyc KB/Blackboard. As discussed in (3), above, in Section 0, these are not strictly necessary but can contribute significantly to efficiency and throughput. Some of these control mechanisms already in Cyc include:

Agenda Processing

Optionally, operations performed on the Cyc KB/Blackboard (e.g., assertions, edits, and unasserts) may be passed through the Cyc Agenda. The Agenda serves to serialize inputs to the KB, which prevents multiple operations from simultaneously operating on a single KB/Blackboard object. Although the other monitoring mechanisms generally suffice, API access to the Agenda is possible, providing a particularly powerful way for knowledge sources to monitor BB changes.

Syntactic Well-Formed Formula and Logical Conformity Checking

Any assertion added to the Cyc KB/Blackboard can (but needn't always) be run through one of several available levels of logical checking in addition to syntactic well-formed formula ("WFF") checking.

Assertions added by trusted sources are typically "waved through" by giving assertions the benefit of the doubt and augmenting each *prima facie* logically incomplete assertion with whatever is required to make it assertable.

Less trusted sources can be subjected to a stronger version of checking which will reject assertions whose predicate arguments, for example, are not provably of an appropriate type.

Task Scheduling and Planning

As well as the agenda and assertion-correctness checking and repair, Cyc provides additional infrastructure that can be combined as appropriate with custom code to produce a Blackboard Controller:

- Cyc supports scheduled KB events, which, together with strong support for forward-chaining inference, can make scheduled or planned changes to the KB/Blackboard to trigger actions by the external knowledge sources.
- Cyc integrates a SHOP planner, which can be applied directly to KB/Blackboard content, to plan and execute sequences of KB modification or external knowledge source invocation. This planner is integrated with general inference, which can be used in the course of constructing a plan.