Use of Database Organization in the Consul System[1]

William Mark
USC/Information Sciences Institute

1. Introduction

A fundamental aspect of the Consul system is the way the organization of the database defines the system's operation and unifies its components. Accessing and maintaining the database is the focus of all system activities, from parsing English to executing programs.

2. The Consul System

The purpose of the Consul system is to provide cooperative interaction between users and a set of online tools (for text manipulation, message handling, network file transmission, etc.). "Cooperative" interaction includes natural language requests for system action and explanation of system activities. In order to provide this kind of interaction, Consul must have detailed models of both the user's needs and the system's requirements--and must be able to translate back and forth between them.

The processes and divisions of knowledge in Consul are shown in Figure 1. The system's basic operating mode is as follows: the user types a request into the system; the request is parsed, i.e., rendered into the system's knowledge representation; inference rules are used to map the representation of the user request into a description of some system action; if this action is a tool operation, the operation is executed by the interpreter, thus fulfilling the user's request; if the action is explanation, the user model is used to generate the appropriate response. There is also a mode of system operation in which Consul incorporates a new tool, as described by the tool builder, into its database. In Consul, all of these system activities--parsing, mapping, interpretation, explanantion, and acquisition--are defined by their interaction with the database.

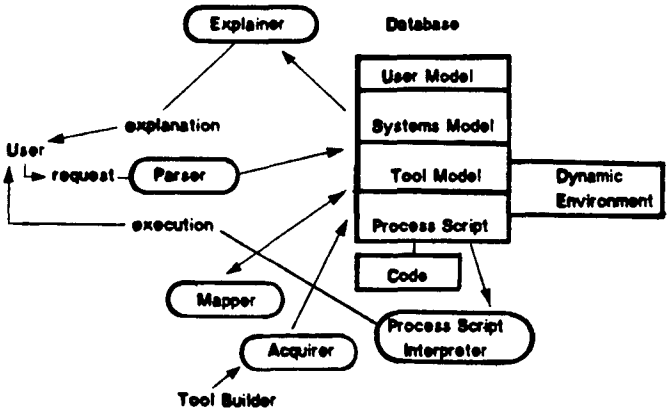

Figure 1: The Consul System

3. The Database

Consul's database is a hierarchical network structure implemented in the KL-ONE formalism[2]. This formalism is used to represent the knowledge required for communication with the user about interactive system activities: descriptions of tool functions (at both general and tool-specific levels) and descriptions of user characteristics (including the actions and objects known to the user, and the English constructs used to express them). In order to understand how the database influences system operation, it is necessary to have a more detailed idea of its contents and connectivity.

3.1. The functional model

Consul contains an abstract model of the actions and objects that are found in interactive tools. This "systems model" consists of descriptions of basic operations such as deletion, scheduling, and display, along with the data structures these operations work on.

A "tool model" is a particularization of the systems model to the actual operations and data structures of some interactive tool that is implemented in Consul. This tool-specific information is built up as part of the process of incorporating a new tool into the system (see Acquisition below).

3.2. The user model

Understanding user requests and explaining system activities requires a detailed model of the user's world: his basic needs (e.g., "change the current environment"),

[1]This research was supported by the Defense Advanced Research Projects Agency under Contract No. DAHC15 72 C 0308, ARPA Order No. 2223. Views and conclusions contained in this paper are the author's and should not be interpreted as representing the official opinion or policy of DARPA, the U.S. Government or any person or agency connected with them.

[2]Ronald Brachman, *A Structural Paradigm for Representing Knowledge*, BBN Report No. 3605, 1978.

his actions and objects (e.g., "remove an object", "message"), and his English language expressions of those actions and objects (e.g., "Get rid of message 6").

3.3. The dynamic environment

In order to set up appropriate execution sequences and construct certain explanations, Consul must have a model of the time course of events. Therefore, all system and user activities are modelled as events in time, i.e., invocations of the actions defined in the user and tool models. This event model serves as a dynamic environment for expressing the *behavior* of the user and the system.

3.4. Connections

It should be clear from the above discussion that the various aspects of Consul knowledge are closely interrelated. The database organization must reflect these relationships in such a way that the various components of the system can utilize relevant knowledge and see its impact on overall system operation.

The primary connectivity in the database is (as is often the case) inheritance relationships defined by a hierarchy. • That is, all operations which are kinds of delete operation inherit the characteristics defined for "delete operation" (see Figure 2). This inheritance relationship serves not only to concentrate shared characteristics in a single location; it also serves to *enforce* the defined pattern of characteristics on all elements in the lineage. That is, each element must explicitly account for all inherited characteristics, showing that the pattern of inheritance is satisfied by the features of its own structure. This explicit use of inheritance relationships to establish the legality of classification is the basic operating principle of the Consul system, as we will see below.

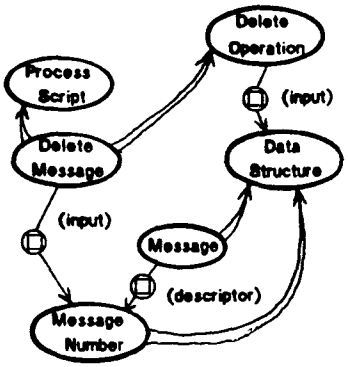

Figure 2: Inheritance Relationships

The second kind of connectivity is implicit linkage through inference rules. Rules specify the possibility of transforming one data structure into another. The transformation is expressed as an explicit linkage between *classes* of information. The implicit linkage occurs between objects that are in those classes, as determined by the object's position in the network database. For example, "Rule 1" in Figure 3 expresses the possibility of redescribing requests to remove lists of things as a sequence of delete operations. When "Smith" types in a request to "Get rid of messages 4, 9, and 11", the resulting parse, shown as the structure headed "RemoveRequestAct.1" in Figure 3, is positioned in the network as shown (the details of this process are beyond the scope of this paper[3]). The fact that RemoveRequestAct.1 is then seen to be in the class RemoveRequestAct (shown by double lines), MessageNumberList.1 in the class List, and so on, allows Rule 1 to be applied as shown, resulting in the redescription shown at the right of the figure.

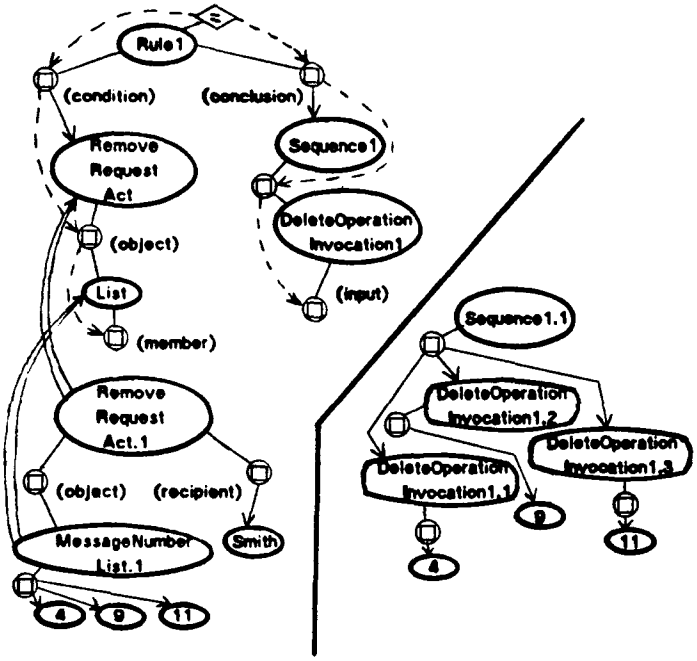

Figure 3: Rule Application in Consul

The final kind of connectivity in the Consul database has already been mentioned: explicit relationships in the dynamic environment. The events in the dynamic environment represent occurrences in the real world of the system, that is, the *act* of making requests, the *invocation* of operations. As shown in Figure 4, these events give rise to a completely separate (non-inherited) set of links between data structures based on the position of those structures in time.

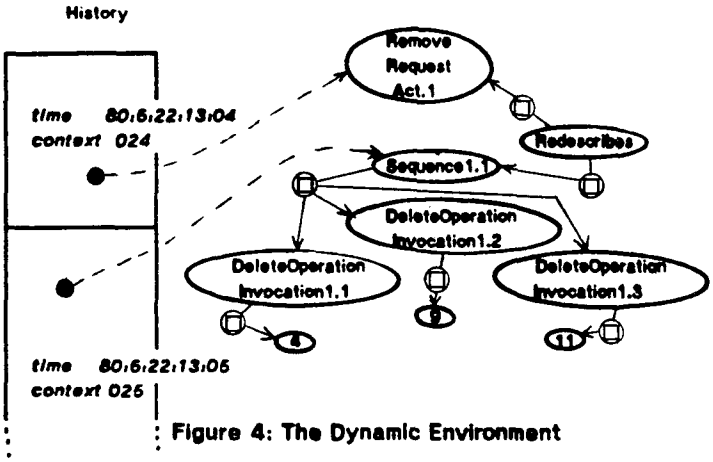

Figure 4: The Dynamic Environment

4. Using The Database

The database organization is basic to the methodology of all system processes; mapping, explanantion, and

[3] See William Mark, "Rule-Based Inference in Large Knowledge Bases", *Proceedings of the National Conference on Artificial Intelligence*, 1980.

acquisition will be described below. The parser methodology also depends heavily on the database organization, but the details of this dependency are beyond the scope of this discussion[4].

4.1. Mapping

Mapping in Consul is the process of taking an existing description in the database, redescribing it in accordance with an inference rule, and then reclassifying it in the database. A key aspect of Consul mapping is that inference rules are also represented in KL-ONE in the database, as shown in Figure 3; the same process of classification that determines the "legal position" of a data structure also determines the applicability of a rule[5]. Rules are applied until some redescription is found that can be classified as a tool operation or an explanation action. For example, rules would be applied until the structure headed "Sequence1.1" in Figure 3 could be described as a sequence of the actual tool operations which delete messages--rather than a sequence of unknown "DeleteOperationInvocations" as shown in the figure.

The mapping process therefore determines rule applicability by finding class relationships, and applies rules by redescribing the satisfied condition in terms of the conclusion. The entire process consists of the utilization of database connectivity: inheritance links to find opportunities for redescription, and rule links to actually create redescription[6].

4.2. Explanation

Explanation is the process of using connections between the functional model and the user model to construct answers to user questions. This involves use of hierarchical inheritance, inference rule, and dynamic environment links to establish the relationship between the concept the user is questioning (some action, object, or event), and the system's representation of that concept. The same sort of links must then be used to find data structures in the user model related to the system's model of the concept in question--to serve as the basis for the answer to the user. For example, if the user asks "Which messages were just deleted?", Consul must find in terms of system effects what it thinks the user means by "deleting messages"; find the kinds of system events that give rise to those effects; find the particular system events that caused the particular effects in question (i.e., recent message deletions); find the particular user-initiated event (i.e., the user's last relevant request) that caused those system events; find the particular effects (in this case, message deletions) ultimately caused by that particular event; redescribe those effects in user terms. All of this involves mapping back and forth between the user and system models, the relevant objects being determined by links in the dynamic environment (see Figure 4), and the relevant explanatory actions being determined by mapping rules and inheritance relationships.

4.3. Acquisition

Acquisition is the process of incorporating knowledge about an individual tool into the existing model of system functionality. The knowledge of what any individual tool can actually do (send a message, move a block of text, etc.) is entered into Consul by the tool-builder in terms of the *process script* programming formalism. Process scripts consist of two parts: a procedure and some descriptive information about that procedure. The descriptive part consists of a small number of procedure characteristics that Consul has predetermined to be relevant. A simple process script to implement a "message deletion" operation is shown in Figure 5.

Consul's acquisition mechanism uses inheritance relationships to classify an incoming process script description as some known description of an operation. As mentioned earlier, this can also be viewed as enforcement of the constraints of Consul's systems model. A simple example is shown in Figure 2: if the process script DeleteMessage is to be understood as a "Delete Operation", its input specification MessageNumber in Figure 5 must be describable in Consul's database as a valid input for a "Delete Operation". When other aspects of the DeleteMessage process script (the other descriptors and certain features of the procedure body) are appropriately classified, Consul will have the DeleteMessage operation "modelled"--i.e., understood in terms of its built-in model of interactive systems. Thus, in Consul, acquisition is tantamount to classification, and is strictly a database access and update methodology.

```
ProcessScript DeleteMessage;
        Input              mn:MessageNumber;
        Output             None;
        Preconditions      None;
        SideEffects        MessageDeletedSV(mn) := True;
        Undo               UndeleteMessage(mn);
        Error Conditions   Error1,Error2;
        Body
          begin
            x := FindFilePointer(mn);
            if Null(x) then fail with Error1;
            if not Deletable(x) then fail with Error2;
            MarkDeleted(x);
          end;
```

Figure 5: A Process Script

5. Conclusion

In the Consul system, the database unifies the activities of the system components--not simply because they all use the same database, but because their behavior is actually defined by the organizational characteristics of that database. To the extent we have found possible at this stage of development, Consul's know-how is in the objects of the database and their connectivity. The active processes that supply system facilities are straightforward programs based on a "database user" methodology.

---

[4] See Rusty Bobrow and Bonnie Webber, "PSI-KLONE: Parsing and Semantic Interpretation in the BBN Natural Language Understanding System", *Proceedings of the 1980 Conference of the Canadian Society for Computational Studies of Intelligence*, 1980.

[5] Compare Richard Fikes and Gary Hendrix, "A Network-Based Knowledge Representation and its Natural Deduction System", *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, 1977.

[6] See also Mark, *op. cit.*.