



Context-mediated behavior for intelligent agents

ROY M. TURNER

Department of Computer Science, University of Maine, Orono, ME 04469 USA.
email: rmt@umcs.maine.edu

Humans and other animals are exquisitely attuned to their context. Context affects almost all aspects of behavior, and it does so for the most part automatically, without a conscious reasoning effort. This would be a very useful property for an artificial agent to have: upon recognizing its context, the agent's behavior would automatically adjust to fit it. This paper describes *context-mediated behavior* (CMB), an approach to context-sensitive behavior we have developed over the past few years for intelligent autonomous agents. In CMB, contexts are represented explicitly as *contextual schemas* (c-schemas). An agent recognizes its context by finding the c-schemas that match it, then it merges these to form a coherent representation of the current context. This includes not only a description of the context, but also information about how to behave in it. From that point until the next context change, knowledge for context-sensitive behavior is available with no additional effort. This is used to influence perception, make predictions about the world, handle unanticipated events, determine the context-dependent meaning of concepts, focus attention and select actions. CMB is being implemented in the Orca program, an intelligent controller for autonomous underwater vehicles.

© 1998 Academic Press Limited.

1. Introduction

You do not ride a bicycle on a train. The idea never crosses your mind, even if you have a bicycle with you, even if you are an avid bicyclist. Why not? Because it is not appropriate in that context. If asked, you could immediately reply that this is not something one does on a train. How do you know? You just do—the knowledge of what is appropriate is automatically available to you when you are in or are thinking about the context.

Humans are exquisitely attuned to their context. There can be no serious objection to the statement that for humans and other animals, behavior is context-dependent. Animals that behave inappropriately for their context will likely not survive to pass on their genes; this is part of evolution. Humans who behave inappropriately for their context can face the same problem; at best, they risk being labeled insane. Indeed, the term "appropriate" in relation to behavior is meaningless without recourse to some context. There is no such thing as context-free appropriate behavior.

Artificial agents must also exhibit context-sensitive behavior. This, too, is not a contentious point. All successful artificial intelligence programs are context-sensitive, though it may take a great deal of effort and the range of contexts may be severely limited. An artificial agent that does not behave appropriately for its context is useless.

Then the interesting question is not whether context-sensitive behavior exists or is useful, but rather: how can an intelligent artificial agent take context into account as effortlessly and automatically as humans and animals do?

Over the past few years, we have developed an approach to context-sensitive behavior for artificial intelligence systems as part of our research in schema-based reasoning (Turner, 1994). We call this approach *context-mediated behavior*, because attention to context underlies almost all of the agent's behavior.

We use the term *context* to mean any identifiable configuration of environmental, mission-related and agent-related features that has predictive power for an agent's behavior. The term *situation* is used to refer to the entire set of circumstances surrounding an agent, including the agent's own internal state. Context is thus the elements of the situation that should impact behavior.

Context-mediated behavior (CMB) is based on the idea that an agent should have explicit knowledge about contexts in which it may find itself, then use that knowledge when in those contexts. In our approach, this knowledge is represented as *contextual schemas* (Turner, 1989ft, 1994). Each contextual schema (c-schema) contains both descriptive knowledge about a particular context and prescriptive knowledge about how the agent should behave in that context. C-schemas are organized in a conceptual, content-addressable memory. Features of the current situation are used to find matching c-schemas, which are merged to create a coherent view of the agent's context. Prescriptive contextual information is then used to bring the agent's behavior in line with its context.

This approach to context-sensitive behavior has several desirable properties: first, it is efficient. When the situation changes significantly, a decision is made about what the context is. From then until the next change, the agent does not need to reason about the context; instead, it simply uses the information from the current set of c-schemas. This is in contrast to standard decision-making in planning, rule-based systems, and most other artificial intelligence systems, in which every decision made involves considering what the current context is. Contexts are often long-lived, with many decisions being made while the agent is in the context. Consequently, CMB will be more efficient than reasoning about the context each time decision-making is necessary.

Second, context-sensitive behavior is *automatic* once a context is recognized. This means that behavior selection can be fast, since the agent's decision-making from that point on uses the contextual knowledge already present. The context can pre-set reflex behavior, for example, so that little or no decision-making is required at the time the reflexive behavior is needed.

Third, CMB aids perception and understanding. Once the context has been identified, information from the corresponding c-schemas can be used as a source of predictions about unseen features and suggestions of how to disambiguate perceptual data. The predictions from c-schemas can also help the agent predict the occurrence of unanticipated events and to recognize them when they occur. In addition, many concepts have context-dependent meaning. For example, the concept of one agent being close to another has a different meaning when two ships are attempting to dock than it does when they are attempting to avoid colliding with one another at sea. By storing such context-dependent meanings in c-schemas, CMB fosters context-sensitive understanding.

Finally, by representing the context explicitly, CMB opens the way for agents to adjust their contextual knowledge over time based on their own experience. It also allows many novel contexts to be handled, since the agent can blend together knowledge about similar contexts it knows about.

This paper describes a computational model of context-mediated behavior. This model was begun in the MEDIC project (Turner, 1989a, 1994) and is currently being developed in the Orca project (Turner, 1994, 1995). MEDIC was a medical diagnostic reasoner, and Orca is an intelligent controller for autonomous underwater vehicles (AUVs). CMB is being implemented as Orca's ECHO (embedded context-handling object) context management module. Though we are focusing first on AUVs, CMB is meant to be a general approach to context-sensitive behavior for intelligent artificial agents.

We first discuss our application domain and the Orca AUV controller. Next, we examine the question of which aspects of behavior should be affected by an agent's context. We then describe our model, context-mediated behavior. The paper ends with a discussion of related work, then conclusions and plans for future work.

2. Autonomous underwater vehicle control

An AUV is an untethered submersible robot. Figure 1 shows an example AUV, EAVE-III (Experimental Autonomous Vehicle). EAVE is a short-range, experimental vehicle built by our collaborators at the Autonomous Undersea Systems Institute (Blidberg, Turner & Chappell, 1991). Over the years, the EAVE series of vehicles have performed many tasks related to ocean science and engineering. There are many other AUVs currently being built or fielded. Some, like EAVE, have been built for AUV development purposes, while others are beginning to perform real tasks for ocean science, industry and the military (see Blidberg *et al*, 1991, for a review).

AUVs can perform many tasks important for mariculture, the petroleum industry, mining, communications and other industries. These tasks include laying cables, inspecting and repairing oil rigs, doing underwater construction and prospecting. Humans can do these tasks, but the ocean is a hostile environment for humans. Except for military vessels, long-term human presence is mainly limited to the uppermost layer of the sea. Even here, storms and high wind can limit the surface-based support most undersea activities require. AUVs can overcome these problems. They can operate without human presence or surface support, and can perform hazardous tasks and operate in any sea

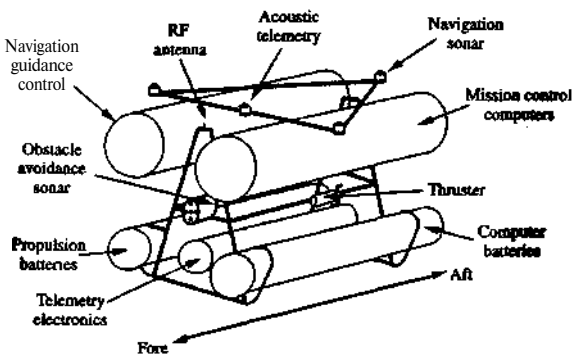


FIGURE 1. The EAVE-III autonomous underwater vehicle.

state without risking human life. They do not have the drawbacks of remotefoperated vehicles (ROVs), since they do not have tethers to become entangled, nor do they require a support vessel on the surface.

AUVs also have great promise for use in oceanography and other marine sciences. Understanding the ocean is important in itself, but it is critical to understanding and predicting global environmental change. Yet it has been said that we know more about the surface of Neptune than about our own planet's oceans (Blidberg *et al.*, 1991). The ocean is mostly impervious to airborne or spaceborne sensors. Satellites are essentially limited to observing the topmost few meters of the sea's surface, and, even here, "ground-truthing" their data requires measurements to be taken on-site. AUVs are ideal for studying the ocean. They can carry sensors to where they are needed, and they can move to characterize phenomena of interest. They do not suffer from the motion limitations of moorings or drifters, nor do they require human presence, as do ROVs.

AUVs have many military applications. In addition to the most obvious ones, AUVs can gather information in hazardous areas, they can be used to clear mines without risking human life, and they have promise for use in rescue missions.

The AUV control domain provides a challenging problem for artificial intelligence (AI). An AUV operates in a world about which we have very little knowledge, hence uncertainty is high. Sensors for underwater use are notoriously noisy and plagued with uncertainty. In addition, the underwater world is not static; indeed, it can change drastically over a small period of time or a small distance. Complicating matters further, many important missions have long duration. For example, it has been proposed (Blidberg *et al.*, 1991) that AUVs function as "underwater satellites", remaining on-station to return data over long periods of time. In such missions, the likelihood of equipment failure and errors arising from uncertainty increases dramatically.

Context-sensitive behavior is critical to AUVs, as it is for any other reasoner. An AUV controller will need to behave appropriately in many different contexts. Very often, an AUV will be used for many different missions over its lifetime, and it will carry out those missions in many different kinds of environments: in harbors, in the littoral (near-shore) zone, in the deep ocean, under ice and so forth. Even a single mission will often involve operation in several different contexts. An AUV controller needs knowledge about the different contexts in which it will be operating so that it can adjust its behavior appropriately.

But just having contextual knowledge is not enough. The AUV must be able to rapidly find and apply the knowledge, preferably automatically, to decide how to behave. Decisions about what to do in a given situation must often be made very quickly, or else the AUV risks its mission or itself. For example, when power fails, the AUV must decide at once what to do. If it is in a harbor, then landing on the bottom and releasing a buoy may be the best response; if it is in the open ocean with kilometers of water beneath it, this is likely not the appropriate thing to do, and instead it should surface and radio for help. There will be virtually no time to decide what to do when the event occurs.

At the present time, widespread use of AUVs is limited as much or more by the lack of sophisticated control software as by any other factor. AUVs have been capable of performing a variety of simple tasks for years. Before they can reach their full potential, however, they need intelligent controllers that can carry out complex missions and handle unanticipated problems that arise.

Orca (Turner, 1994, 1995) is an intelligent mission controller for autonomous agents being developed at the University of Maine. We concentrate mostly on AUVs undertaking oceanographic missions. Orca's role is to free ocean scientists and other users from needing detailed knowledge of AUV control. Users will describe their missions to Orca in terms meaningful to them, then it will plan and carry out the mission, while handling any problems that may arise.

Orca is a schema-based, adaptive reasoner. It consists of several co-operating modules, as shown in Figure 2. Schema Applier (SA) is responsible for finding and applying a p-schema to achieve the goal or goals currently in focus. Agenda Manager (AM) is responsible for maintaining Orca's focus of attention. Event Handler (EH) handles all input from outside the program, and it is responsible for handling any events, unanticipated or anticipated, that it detects. We assume Orca will control the agent's hardware (e.g. sensors and effectors) indirectly, so input will be from lower-level control software. Long-term memory is responsible for retrieving p-schemas and c-schemas in response to "probes" from the context manager and SA, and working memory holds information for all the modules.

CMB is implemented by the context manager, ECHO. As can be seen from the figure, ECHO plays an important role in Orca. It is responsible for maintaining Orca's

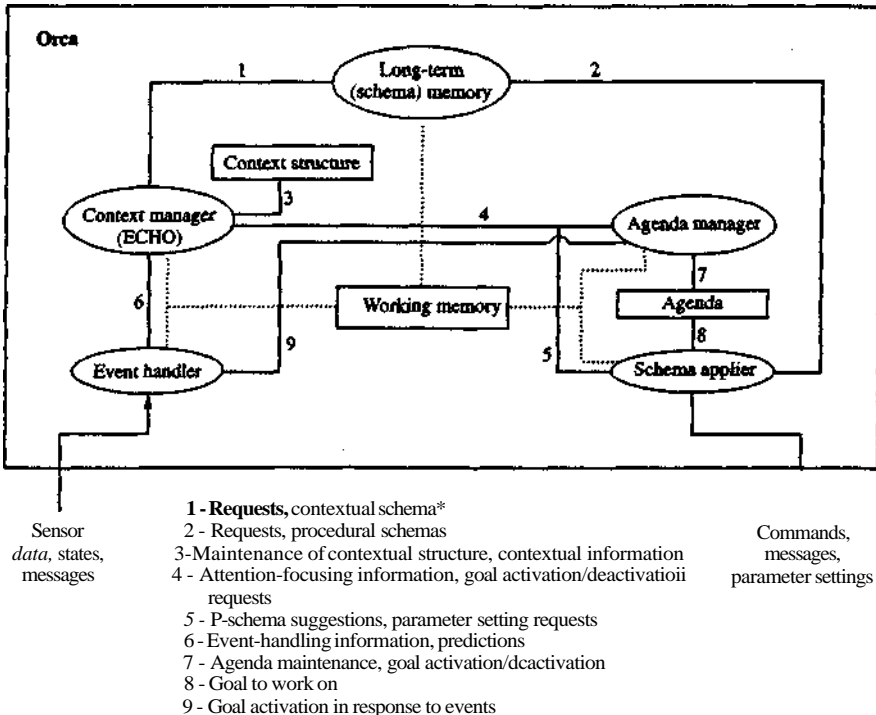


FIGURE 2. Internal structure of Orca.

representation of its current context (the *context structure*) and for disseminating contextual information to the other modules.

Orca is implemented in Allegro Common Lisp and CLOS on a Sun workstation. ECHO is being implemented as a CLOS class with its own Lisp process. Orca is currently being developed and tested in our SMART simulation testbed, with the goal of eventual fielding aboard AUVs. Our initial target vehicle is the EAVE-III vehicle, described above. Ultimately, we anticipate Orca controlling long-range AUVs and AUVs involved in multi-AUV systems. The current version of Orca is being used by other projects focusing on such systems.

3. Context

Given that an agent's context impacts its behavior, the question naturally arises: what is "context"? Most often, the answer is somewhat vague: context is the current situation, the environment, etc. Recent workshops and conferences (e.g. Brezillon & Cavalcanti, 1997) have focused some attention on this question, though unfortunately without much agreement.

Perhaps by restricting our consideration here to the domain of controlling intelligent autonomous agents such as AUVs we can have better luck. We repeat here our definition of context:

A *context* is any identifiable configuration of environmental, mission-related, and agent-related features that has predictive power for behavior.

This means that a configuration of features is only considered to be a context if it predicts something about how the agent should behave when those features are present. Thus, the set of contexts is a drastically reduced subset of all possible feature configurations. For an AUV, docking with a support ship is an important context. Features of this context include the presence of the AUV and ship close to one another; the AUV has the goal of docking; both vessels are equipped for docking; and so forth. This context subsumes a very large number of situations that vary from one another along features that have no impact on the agent's behavior: color of the ship, day of the week, position of Mars, etc. The restriction "identifiable" further reduces the set of contexts. If the agent cannot identify the context, it makes no sense to store a representation of it.

Let us look at a few examples of contexts in the AUV domain. Consider an AUV operating in the open ocean. Here, if it needs to determine where it is, it can surface and take a GPS (global positioning system) fix without much chance of collision. When there is a catastrophic failure, it should attempt to surface and radio for help, since landing on the bottom here would likely mean going below its crush depth.

Other environments have different predictions for how the AUV should behave. Consider "in a harbor". Since there is likely to be traffic on the surface and possible clutter on the bottom, the AUV's depth envelope should be tightened automatically when entering the harbor. Similarly, should the AUV need to determine where it is, it should first try means that do not require it to surface, since that would risk a collision; if it does have to surface, it should do so carefully, after trying to locate any approaching vessels. Should a catastrophic event occur, the AUV should land on the bottom and

release a buoy rather than surfacing; knowing that it is in a harbor should allow it to make this decision quickly.

Though this context can help an AUV behave appropriately in a wide range of harbors, some harbors have different properties that affect behavior. For example, Portsmouth Harbor (NH) has extremely strong tidal currents and Bar Harbor (ME) has a sandbar that completely blocks part of the harbor at low tide. An AUV operating in these harbors must know about these specializations in order to behave appropriately in the contexts.

Missions can also define contexts. The range of actions available to an AUV while on a mission is one thing that is constrained by such a context. For example, the context "on a sampling mission" might restrict the way the AUV moves so as to avoid disturbing the sites from which data is to be collected. A context such as "on a rescue mission" affects the behavior in other ways. In that context, for example, the AUV should focus its attention on the main mission task, even at the expense of other goals that might otherwise be quite important, such as "determine location". In the context of "on a cooperative mission", the way goals are achieved can be different than in other contexts. For example, in that context, the AUV can ask other AUVs to achieve goals for it. In addition, which goals it focuses attention on may be partially determined by the goals other AUVs are working on.

Properties of the AUV itself can also define contexts. An AUV changes over time. One way this happens is when the AUV is equipped with different sensors and effectors for different missions. For example, the context "equipped with a CTD"^t has implications for what kinds of things the AUV can do. In the context of having a piece of equipment that is pressure-sensitive, the AUV should know not to exceed that depth. An AUV also changes when there are electrical or mechanical failures, and these changes give rise to contexts that affect behavior. The context "low power", for example, impacts behavior by changing the way goals are carried out (e.g. choose shorter paths, move more slowly, etc.) or even eliminating some goals from consideration; the context "power failing" should immediately cause the AUV to take the appropriate actions, for example, by activating a goal to abort the mission.

Requiring contexts to be identifiable and predictive reduces the number of contexts an agent must know about, but even so, the space of possible contexts necessary may be huge. For example, an AUV may find itself in the contexts: in a harbor; in a harbor with low power; in Bar Harbor at low tide; in Bar Harbor at low tide with low power; and so on. It would be unwise to attempt to represent all possible contexts, especially since many of them can be seen to be composed of simpler contexts.

We suggest that the following principle guide the selection, either by humans or the agent itself, of which contexts to represent:

A context should be represented as a c-schema only if (1) it cannot be represented by merging the knowledge contained in existing c-schemas or (2) if such a merger fails to prescribe the correct behavior for the context.

Thus, the context "in a harbor during incoming tide" might be adequately captured by merging two representations, one for "in a harbor" and another for "incoming tide", for

^t An instrument for measuring conductivity (and indirectly, salinity), temperature and depth.

a wide range of harbors. However, the context "in Portsmouth Harbor during incoming tide" would likely require its own representation, since the strong currents in that harbor would not be adequately predicted by merging existing representations.

4. Behavior affected by context

Context impacts almost all facets of behavior. In humans, for example, perception is heavily influenced by top-down predictions, making it more difficult to detect out-of-context objects than familiar ones, and there are numerous studies showing the priming effect of one concept on another (see e.g. Glass & Holyoak, 1986). Pevtsov and Goldstone (1994) suggest that the categories a person has learned affects what features of an object he or she perceives. Context also impacts decision-making and action. Classic studies by Tversky and Kahneman (1974) show the effect of context on the estimation of probabilities; for example, in the "gambler's fallacy", after a string of heads in a coin toss, the subject's estimation of the probability of the next toss being tails deviates from 1/2. Preference measurements are also context-sensitive (Mellers & Cooke, 1996). Learning is affected by context, as studies as far back as those of Pavlov have demonstrated. Context has been studied extensively in language use, usually with "context" meaning the history of prior utterances (e.g. Ferstl, 1994), but also including other kinds of context. Holtgraves (1994) has found that the status of the speaker relative to the hearer affects whether the literal meaning of an indirect request is activated. Kraemer and Piwek (1997) have found that non-linguistic context is used in filling pre-suppositional gaps in utterances. Social interaction is also context-dependent, as the work of Holtgraves cited earlier suggests. Mantovani (1996) concludes that "patterns of activity" are regulated by cultural models, and he suggests a role for context strikingly similar to one of the roles we study in our work: context evaluation gives rise to actions intended to cope with "foreseeable" events.

Context should affect an artificial agent's behavior in similar ways. Above, we discussed some of the ways an AUV's behavior is affected by context. Here, we discuss this in more detail. We have identified the following aspects of an agent's behavior that context should affect.

1. *Making sense of the situation.* Before an agent can decide how to behave, it must first understand what its current context is. Situation assessment begins with identifying the context the agent is in, then uses knowledge about that context to flesh out the agent's understanding of its situation.

Knowledge about the context should help the agent answer such questions as given below.

- *What features of the context are predicted, but not yet seen in the current situation?* For example, when entering an unfamiliar movie theater, you can confidently predict, based on your knowledge about the context, that there is an auditorium inside, there are seats and a screen in the auditorium, etc. When an AUV realizes it is in a harbor, it should be able to predict that there is likely to be traffic overhead. This is important for guiding the agent's actions (e.g. be careful when surfacing).
- *What features of the current situation are unusual, i.e. do not agree with the agent's expectations about the context?* For example, if there is no surface traffic in the harbor,

the AUV should recognize that this is unusual, and it may allow the agent to make additional hypotheses (e.g. there may be a storm or it may *not* be in a harbor, etc.) that have a bearing on what behavior is appropriate.

- *Do some concepts have different meanings in this context, and if so, what are those meanings?* For example, in a harbor, the concept of being too deep means something different than it does in the open ocean. In the former, it may mean the AUV is too close to the bottom, while in the latter, it may mean that the vehicle is near its crash depth.
- *How should the agent interpret sensor data in this context?* For example, detection of an object in the water column by sonar would usually cause the AUV to create a hypothesis about a potential obstacle. In the context of "in a mined harbor", however, the AUV can create the more specific hypothesis that the object may be a mine.

With respect to the last point, context can also help interpret ambiguous data. For example, an AUV may often detect *two* sonar returns from its down-looking sonar that could be the bottom. Knowing that the context is "in an estuary", the AUV should automatically take the topmost return as the actual depth of the bottom and the lower return as the harder surface under the silt. In the open ocean, the lower return is more likely to be the bottom, with the upper one resulting from the "deep scattering layer" composed of a high concentration of animals (Levinton, 1982).

2. *Automatically modulating behavior to fit the context.* Humans do this quite well. When one enters a library, one automatically lowers one's voice; when a movie is over, one automatically acquires the goal of leaving the theater. An artificial agent should also be able to do this. Once a context is recognized, the agent should automatically behave appropriately without further reasoning effort. For example, upon entering a harbor, an AUV should automatically tighten its depth envelope; when lost, it should automatically have the goal of surfacing for a GPS fix or to radio home.

3. *Handling unanticipated events.* The world is a cruel place for autonomous agents. Uncertainty and incomplete knowledge virtually guarantee that unanticipated events will arise and, if not handled, spell disaster for the agent. Often when events occur, the agent will not have the luxury of unlimited time for reasoning about them.

Contextual knowledge can help an agent handle unanticipated events. Once the context is recognized, knowledge about event-handling should be ready and available immediately when an event occurs. The knowledge would help the agent to do the following.

- *Detect the event:* for example, an AUV is unlikely to have a "current detector"; instead, it must infer that it is in a current by other means, such as comparing its position with the expected position. Contextual knowledge can provide both a context-specific hypotheses based on observed data and information useful in confirming the hypotheses.
- *Evaluate the event:* what an event means, and hence how important it is, depends on the context. For example, "low power" is quite important in many contexts for an AUV, yet not very important in others, such as the context of just having been recovered by the support vessel. The context also determines when an event is important enough to do something.

- *Respond to the event*: what action to take in response to an event depends $\langle \text{context} \rangle$. In the open ocean, for example, if an AUV detects an incipient loss of power, it may make sense to surface and radio for help; however, in a harbor, it may be better to land and release a buoy, so that the AUV will not get run over. Since the response to this event, like many others, must be selected swiftly, the agent's knowledge about the context it is in is vital.

4. *Deciding what to focus attention on*. Any agent with limited cognitive resources faces this problem, which is context-dependent. For example, a perfectly reasonable thing for an ambulance driver to do on the way home from work is focus on the goal of picking up his or her cleaning; the driver should not do this, however, if the context is going to pick up a patient. Similarly, an AUV that is interrupted from its sampling task to rescue a diver should not focus attention on sampling-related goals until the rescue is complete.

5. *Selecting actions to achieve goals*. Once an agent decides what goal to work on, it must choose a means to achieve it. Action selection is highly context-dependent. For a person who is hungry, the appropriate action to achieve the goal of satisfying hunger might be "cook food" if at home, "ask about food" if visiting, and "buy food" if at the movies. For an AUV with the goal of determining its position, the appropriate action might be to use GPS if it can surface, long-baseline (LBL) navigation if it is within an LBL network, and dead reckoning in other contexts.

Ideally, an agent's knowledge of its context should allow the agent effortlessly to know what actions are appropriate in the context. Inappropriate actions should not be brought to mind at all. Returning to the example that opened the paper, one should not think of riding a bicycle on a train because one's contextual knowledge about train riding does not even suggest it.

6. *Selecting problem-solving strategies*. Different contexts require different problem-solving strategies. For example, the way a medical student takes a history and a physical examination of a patient is different from the way an experienced physician does. The student is taught to follow a pre-set history and physical examination of a patient, whereas a more experienced practitioner is more confident and can break from the routine pattern to follow up important symptoms. This is an example of context-specific strategy selection, where the context here has to do with properties of the person (level of experience) rather than environmental features.

5. Context-mediated behavior

Context-mediated behavior (CMB) is a mechanism for ensuring that an agent behaves appropriately for its context. It explicitly represents an agent's contextual knowledge and ensures that the right knowledge is available at the time it is needed. CMB is implemented by ECHO, one of Orca's modules.

The overall CMB process is shown in Figure 3. When the system is initialized, and whenever there is a significant change in the situation, ECHO searches Orca's long-term memory for contextual schemas that are similar to the new situation. This process is

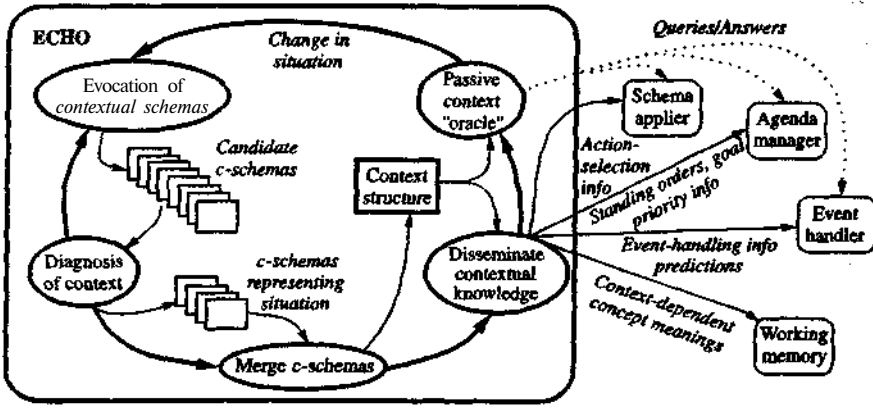


FIGURE 3. The context-mediated behavior process.

called "evocation" (cf. Miller, Pople & Myers, 1982), since the current situation reminds (cf. Kolodner, 1984) the reasoner of, or "evokes", the c-schemas. Once a pool of candidate c-schemas has been found, ECHO determines which of the c-schemas best characterize the current situation by diagnosing the situation as being an instance of one or more contexts. The result is used to create a *context structure* representing the current context. If more than one c-schema fits the situation—which is likely—then their information is merged. Contextual knowledge is then disseminated to the agent's other reasoning modules.

In the rest of this section, we discuss the pieces of CMB in more detail. For readability, we write as if the implementation is complete, which it is not. At the time of writing, the CMB process has been designed as described in this paper, and implementation is currently under way. We have fleshed out CMB in some areas [e.g. context-dependent meaning of fuzzy knowledge (Turner, 1997)] more than others (e.g. detecting context change). We anticipate completion of an initial implementation at or near the time of publication.

5.1. REPRESENTING CONTEXTUAL KNOWLEDGE

All contextual knowledge in CMB is stored in *contextual schemas*. Each c-schema is a frame-like knowledge structure. A frame is a slot-filler representation of knowledge, in which the slots (or "roles") are features of the thing being represented and the filler is a description of the value of the feature. Each c-schema represents a particular context, i.e. a particular class of problem-solving situations.

Figure 4 shows a c-schema from Orca representing "in a harbor". C-schemas are implemented in Orca as frames based on CLOS (common lisp object system) objects. The notation "name" means a frame named "name". The lists shown as fillers are themselves frame-like structures called "frame patterns". The head of a list is a frame, and the rest is a set of slot/filler pairs providing information about the frame. For example, the

```

Actors:
Self: on AUV
("ACTOR-DESC (VARIABLE 7SELF) (BINDING JSELF) (CF 1.0)
  (DESCRIPTION ("AUV» (NAME ACIKPENALTY 1.0))

Objects:
Selling: α place
("OBJECT-DESC (VARIABLE 7PLACE)
  (BINDING ILOCALE) (NAME OBO)
  (DESCRIPTION ("PLACE»(CF 1.OXPENALTY 1.0»

Mission: a missioa
(«OBJECT-DESC (VARIABLE7MISSION)(BINDINGJMISSION)
  (DESCRIPTION ("MISSION)) (CF0.5) (NAME OBI))

WC: the water column
("OBJECT-DESC (VARIABLE ?WC)
  (DESCRIPTION ("WATER-COLUMN))(NAME OB3»

Surface: the surface
("OBJECT-DESC (VARIABLE 7SURFACE)
  (DESCRIPTION ("SURFACE) (NAME OB2»

Description:
The name for this context is "in harbor".
("FEATURE-DESC
  (DESCRIPTION (NAME SCONTEXT "in harbor»)
  (CF 1.0) (NAME FEO)) .
The water column has fuzzy value shallow.
("FEATURE-DESC (DESCRIPTION (DEPTH 7WC SHALLOW)
  (CF 0.8) (NAME FBI»

There is (fuzzy value) some surface traffic.
("FEATURE-DESC (DESCRIPTION
  (AND (TRAFFIC-VOLUME 7SURFACE 7VALUE)
    (>=7VALUESOME)»
  (CF 0.7) (NAME FEZ))

Definitions:
;; for linguistic value "shallow" of ling, variable "depth":
CWZZY-DEFINITION-DESC
(LINGUISTIC-VARIABLE (SLOT "PHYSICAL-OBJECT DEPTH»
(LINGUISTIC-VALUE SHALLOW)
(MEMBERSHIP-FUNCTION ((0 1) (10 0)»
(CF 0.8) (COMBINATION-TYPE REPLACE) (NAME FUO»

Standing orders:
Tighten depth envelope.
("STANDING-ORDER
  (CONDITION T)
  (DESCRIPTION
    (SET-LLA-PARAMETER DEPTH-ENVELOPE (5 10»
    (CF 0.8) (WHEN DURING) (NAME STO))

Events:
Low power: critical importance, though unlikely: if occurs,
  will cause the ntumion to foil and self to fail;
  response should be to abort the mission.
("EVENT-DESC
  (DESCRIPTION (POWER-LEVEL 7SELFLOW»
  (DIAGNOSTIC-INFORMATION NIL)
  (LIKELIHOOD UNLIKELY) (IMPORTANCE CRITICAL»
  (EFFECTS
    ("EVENT-DESC (DESCRIPTION
      (STATUS 7MISSION FAILED» (CF 0.9»
    ("EVENT-DESC (DESCRIPTION
      (STATUS 7SELF FAILED» (CF 0.9»))

(RESPONSE
("RESPONSE-DESC
  (DESCRIPTION (ACHIEVE ("A-ABORT»)
  (CF 1.0») (NAME EVO»

Goals:
Any goal to be at the surface: importance is low.
("GOAL-DESC
  (DESCRIPTION
    ("ACHIEVEMENT-GOAL (STATE (AT 7SELF (2X ?Y 0»»
    (IMPORTANCE LOW) (NAME GOO»

Actions:
To abort, land and release buoy
("ACTION-DESC
  (DESCRIPTION ("A-ABORT»
  (ACTION ("P-ABORT-TO-BOTTOM»
  (CF0.9)<NAMEAC6>

```

FIGURE 4. A portion of the c-schema c-harbor.

first list in the figure essentially means: "An actor description frame: the variable used in this c-schema to refer to the object represented is 'self', it is an AUV, it is the same as the agent's own representation of itself ('Self), the certainty associated with this object being present is 1.0 and the penalty for this feature being missing is 1.0".

Contextual schemas have several parts. The context description is contained in three slots, **actors**, **objects** and **description**. As can be seen in the figure, this information can include an estimate of how much each feature is expected in the context and what penalty is paid for its absence during context assessment. This information can also be used to make predictions about the current situation based on it being an instance of the context. Context-specific meanings of concepts are contained in the c-schema's **definitions** slot. Information about what to do automatically when entering or exiting the context, called the "standing orders", is contained in a slot of the same name. Event-handling information is contained in the **events** slot. This information consists of descriptions of unanticipated events that may happen, information about

how to recognize them, and knowledge useful for evaluating their importance and selecting an appropriate response. Attention-focusing information is contained in the **goals** slot. This consists of descriptions of goals likely to be present whose importance is different than in other contexts. This information is used to rate the relative priority of goals in the context. Information about appropriate actions for the context is contained in the **actions** slot. These are suggestions of actions to be taken in order to achieve goals that arise in the context. We discuss this information in more detail below.

Strategic knowledge, both domain-independent and domain-dependent, is also provided in c-schemas. An agent could have, for example, a c-schema representing the "meta-context" of being in an unfamiliar context. The c-schema representing this might specify a strategy via goal priorities (e.g. favor goals to gather information), the kinds of event-handling information it provides, standing orders (e.g. be reactive) and actions suggested. The strategy, although implicit, might be thought of as "be cautious". In the future, we will examine the question of whether explicit strategies are useful, perhaps represented by contextual schemas similar to MEDICS strategic schemas (Turner, 1994). We will also consider whether there should be other domain-independent and/or "meta" c-schemas, for example, to represent hypothetical or learning contexts.

For the present, c-schemas in Orca will be acquired from domain experts. We believe that organizing knowledge by the context in which it is needed will prove helpful in eliciting that knowledge from experts. Ultimately, c-schemas should be learned from the agent's own experiences. When a context arises in which the behavior prescribed by the corresponding c-schemas is inappropriate, then the knowledge in those c-schemas will need to be changed or a new c-schema created. When a context arises that cannot be adequately represented by merging existing c-schemas, this should also trigger the agent to create a new c-schema.

We should point out that although we believe the kind of information we represent in c-schemas has wide applicability, the CMB approach is not tied to any particular representation of contextual knowledge. Orca uses a frame-based representation, but CMB applied to other reasoners could store other things in c-schemas. For example, CMB could generate context-sensitive behavior in a neural network by storing context-dependent sets of weights in c-schemas. When the context is recognized, then the appropriate weights for the network would be instantly available.

5.2. RECOGNIZING THE CONTEXT

Context assessment is diagnosis. Features of the situation, information about known kinds of situations and the linkages between the two provide the information necessary to diagnose the current situation as an instance of one or more known contexts. Familiar diagnostic tasks, such as medical diagnosis, can be seen as examples of context assessment. The features of the situation are the patient's signs and symptoms, medical and family history, etc. The contexts the diagnostician knows about are those involving diseases and other pathological states. A diagnosis is an identification of the current case as being an instance of one or more disease states.^t

^t We can go further with this identification of context-mediated behavior and medical diagnosis. Prognosis and treatment information can be seen as specialized types of contextual knowledge associated with the known disease states.

Diagnosis is an abductive task (Miller *et al.*, 1982). We make use of this fact and the work done in AI on diagnostic reasoning. Context assessment in CMB is based on the kind of abductive reasoning done in INTERNIST-I (Miller *et al.*, 1982). The basic process is as follows (see Figure 3). First, Orca's long-term memory is probed using features of the present situation to "evoke" c-schemas that are similar. Each c-schema returned will have an "evoking strength", a number (in this case in (0,1)) that represents the memory's estimate of how much the c-schema should be "brought to mind" by the current situation. The context manager, ECHO, then compares information in the c-schemas such as how strongly they predict certain features to the actual presence or absence of those features, to determine which c-schema(s) are the best candidates to represent the current context. It then merges these to form the *context structure*, which represents the context.

5.2.7. Evoking contextual schemas

In CMB, contextual schemas are organized in a conceptual, content-addressable memory patterned after the CYRUS memory program (Kolodner, 1984). This is the kind of memory often used in case-based and schema-based reasoning systems, and it has several advantages, among which are content-based, efficient retrieval; self-organization; and support for generalization as a by-product of storage of new information.

Each c-schema is a memory structure containing not only contextual information but also information about how it relates to other c-schemas. This information is primarily contained in the c-schemas's *indices*, which link a c-schema to others that are (usually) specializations of it. The memory is thereby organized into a set of multiple discrimination networks, with the nodes being c-schemas and records of particular instances of contexts (cases) and the links being indices. Figure 5 shows a representative piece of such a memory.

Each index has four parts: a feature description, a description of a value for that feature, a pointer to another c-schema and an evoking strength indicating the strength of the link. An index's feature is one believed to be predictive (Kolodner, 1984). Predictive

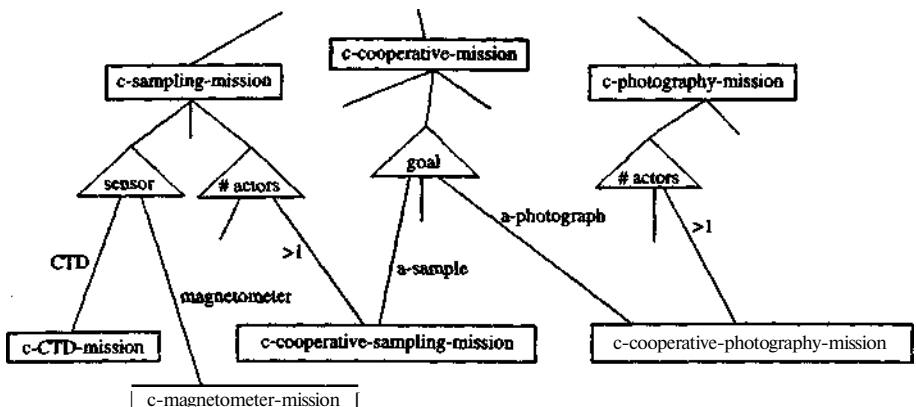


FIGURE 5. Example of a long-term memory organization.

features for a context are those that are expected to be useful in discriminating between various specializations of the c-schema. The value portion is based on the value of the feature in the indexed c-schema.

Contextual schemas are evoked based on the current situation. The retrieval process starts at the top-level c-schema(s) in long-term memory. LTM considers each of the c-schema's predictive features to determine if the current situation contains features that could fit it. If so, then possible values based on that situational feature are generated in a process of index elaboration (Kolodner, 1984), and each predictive feature/value pair is compared to the indices to see if they match. If so, then the c-schema listed in the index is a candidate for representing the situation. If any candidate is found whose evoking strength is over a certain threshold, then the parent c-schema is discarded in favor of the candidate. Each candidate is traversed in the same manner by comparing the working memory to its indices. During the search process, the evoking strength of each candidate is computed. LTM finally gives ECHO a list of c-schemas and their evoking strengths.

5.2.2. *Diagnosing the context*

Once ECHO has a set of evoked c-schemas, it begins the process of deciding which of them, if any, sufficiently match the current situation to be used in context assessment. ECHO first partitions the c-schemas into *logical competitor sets* (Feltovich, Johnson, Moller & Swanson, 1984). Each member of a logical competitor set (LCS) is in competition with the others to explain the presence of a set of features in the current situation. For example, suppose that the forward velocity of an AUV has unexpectedly dropped to zero. This feature might give rise to an LCS consisting of c-schemas representing the contexts "low power", "thruster failure", "aground", "collision" and "caught in net", each of which could explain the feature. The task facing ECHO is to decide which of these c-schemas best represents the context.

This task is often called *differential diagnosis*, since potential diagnoses can be compared to one another for the purposes of information gathering. As in INTERNIST-!, each c-schema in an LCS receives a score based partly on its evoking strength and partly on its predictions for the situation. The job of differential diagnosis is to gather information to confirm the best diagnosis and/or to deny the others. Miller *et al.* (1982) describe several strategies for going about this. We are in the process of examining these strategies for their inclusion in CMB.

Once an LCS is solved, then the features its c-schema accounts for are removed from consideration and a new set of LCSs is formed. The process continues until all important features are accounted for. At this point, ECHO is left with a set of c-schemas, each of which characterizes the situation along a different axis.

5.2.3. *Creating the context structure*

From the set of c-schemas characterizing the situation, the next step is to merge them to create a context structure incorporating their information and representing a coherent picture of the context. This is a complex task, and we have yet to arrive at a general solution.

^t Feltovich *et al.* (1984) suggested that in human experts, such LCSs may be stored in memory to facilitate future diagnostic sessions; this is an interesting possibility to explore in future work.

The major problem is what to do when information from one c-schema convicts with that from another. There are several possible approaches to this problem. *

- *Do nothing*: assume that in the case of conflicting information, reasoning, rather than a priori contextual knowledge, is needed.
- *Leave it ambiguous*: record the set of conflicting values and allow the agent to choose among them when a single value is needed.
- *Take the best*: select the value that has the highest certainty factor.
- *Merge the values*.

It is likely that different kinds of information will call for different strategies. For example, if two different depths are suggested, merging the information by coming up with an intermediate depth may be worse than choosing one or the other. If one was chosen to keep the AUV near the surface to get it out of the way of other submersible traffic, and the other was chosen for the same reason to keep it near the bottom, then selecting an average value would be precisely the wrong thing to do. As our work progresses, we will investigate when the above strategies make sense as well as how to specify which to use for which kind of information.

We have looked in detail at merging one kind of information, the context-dependent meaning of fuzzy subsets (see Turner, 1997). We use a simple form of fuzzy set theory (Zadeh, 1965) in our work to support inferences made by Orca's modules.

In the fuzzy knowledge representation we use, the meaning of linguistic values (e.g. "shallow") for linguistic variables (e.g. "depth") represented by a membership function mapping the variable's numeric range onto the interval [0,1] (see e.g. Zadeh, 1994). It represents how much each element of the range belongs to the set. A c-schema can contain links from a linguistic variable/value combination to a membership function giving that value's meaning. In addition, it specifies how that information should be merged with other membership functions for the variable/value pair.

An example is shown in Figure 6. The information on the left means that in this context, the value "shallow" of the linguistic variable "depth" has the membership function shown on the right, that Orca is 0.8 certain of this in this context (on the scale [- 1, 1]), and that this value should replace any other membership functions for this value. Actual replacement would depend not only on this directive, but also on the certainties of other potential meanings. Other possibilities for combination directives include: replace this with any other information available; take the fuzzy union of all conflicting membership functions; and take the fuzzy intersection.

```
(("fuzzy-definition-deac
  (linguistic-variable (:slot "physical-object depth)) J
  (linguistic-value shallow)
  (membership-function ((01) (10 0)))
  (cf 0.8)
  (combination-type replace))
```

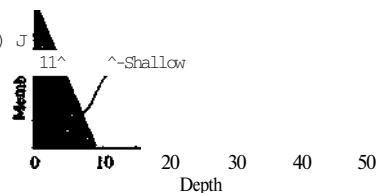


FIGURE 6. A description of a linguistic variable/value pair, from c-harbor.

The context structure itself looks like a c-schema, except it also contains book-keeping information about which c-schemas went into its creation. Keeping the structures the same facilitates memory update. If the context meets the criteria for being remembered, the context structure can be directly turned into a c-schema, then indexed in memory relative to its component c-schemas based on its differences from them.

5.2.4. *Handling novel contexts*

The process outlined above also allows CMB to handle novel contexts. Even when no existing c-schema completely characterizes a new situation, it is likely that it will be similar to some known contexts. In this case, the c-schemas representing the similar contexts will be retrieved and merged to form a representation of a new context corresponding to the situation. The resulting context structure, annotated with the results of the agent's behavior in the context, could then be stored in long-term memory to help the agent handle instances of the context encountered in the future.

5.3. USING CONTEXTUAL KNOWLEDGE

This section discusses how CMB is used to make an agent's behavior fit its context. We will discuss how this happens in Orca and assume that other agents will have equivalent functionality.

5.3.1. *Making sense of the world*

The information provided by c-schemas to support situation assessment includes a description of the features expected; predictions about events; and context-dependent meaning of concepts. When the context is recognized, this information becomes automatically available to the agent to help it make sense of its situation.

Contextual schemas help interpret sensory data by providing top-down predictions about the situation, including predictions of unseen features. For example, when working with other AUVs, an agent's contextual knowledge should allow it to quickly identify moving, AUV-sized objects. Predictions can also affect an agent's plans or allow it to pre-set responses for events that are expected to occur. In addition, this information can allow the agent to determine what has happened in the past, if the predicted event has already occurred. This can help the agent determine where it is in a context extending over time (e.g. "power failing"). In Orca, ECHO sends the context structure's predictions to Event Handler, which posts them in working memory as needed.

A c-schema also provides information about concepts that have a different meaning in the context it represents. As discussed above, c-schemas store context-specific membership functions for linguistic values when those values have a different meaning than usual in the context (see Figure 4). For example, if an AUV is loitering with no mission, "nominal depth" in a harbor may be different than in the open ocean—in the latter, staying near the surface is advisable, whereas in a harbor, to remain too near the surface is to risk collisions. This is handled by the c-schemas representing the two different contexts each providing a different membership function for "nominal depth".

Context-specific meanings are also posted in working memory. As described elsewhere (Turner, 1997), this becomes part of a table of such meanings. For fuzzy values, the table entries are membership functions, and they are indexed by a description of where the

meaning will be needed. For example, an index might be a linguistic variable/value pair, such as "depth of AUVs"/"shallow". Information about the context-dependent meaning of concepts can then be used by all parts of the agent, for example to support how predictions are made, events are handled, attention is focused and actions are selected.

5.3.2. *Modulating behavior*

The standing orders in the context structure are used to automatically modulate the agent's behavior. These are represented in c-schemas by information about goals to activate/deactivate and parameters to set when entering or exiting the context. For example, c-harbor should include standing orders to tighten the depth envelope (see Figure 4) to avoid collisions with surface traffic and the bottom. Our approach allows the c-schema to specify the timing and duration of each standing order. For example, the standing order in c-harbor that sets the depth envelope specifies that the setting should be in effect only during the context. Parameters can be the agent's own, for example, a setting for the event-importance threshold that controls reactivity. Parameters can also be those of the agent's underlying software and hardware, for example, velocity envelopes and sensor sampling rates.

In Orca, when ECHO detects a context change, it determines if any of the old context structure's standing orders should be rescinded and if any new orders should be triggered when exiting the context. These are combined with the orders from the new context structure that become effective on entering that context. Standing orders related to goals are sent to the Agenda Manager. Internal parameters are set directly, and settings for parameters of lower-level software and hardware are sent to the appropriate place.

5.3.3. *Handling unanticipated events*

Contextual schemes contain three kinds of knowledge for event handling. First, there is event detection information, which can include both the description of the event as well as information to support indirectly diagnosing the event from other data. If an event directly matching a description arises, it can easily be detected. However, this will not usually be the case, and the agent will often have to infer the event's presence based on the information provided. Since Orca's Event Handler uses fuzzy rule-based systems for handling events, event detection knowledge is stored as fuzzy rules.

Second, there is event assessment information, which is needed to determine what the event means in the context, i.e. how important it is. This can be expressed simply by an importance estimate, or additional information can be given that allows the event's importance to be modulated based on the particulars of the current situation. This allows a single c-schema to adjust the behavior to different situations that are all instances of the context it represents. A rule-like representation is used for this in both MEDIC and Orca.

And third, there is information about how to respond to unanticipated events. In our approach, this takes the form of information linking events to goals to activate when the event occurs. For example, the c-schema representing being in a harbor might suggest activating the goal "land on bottom and release a buoy" in response to the event "leak". This is an example of c-schemas pre-setting the agent's reflexes in a context-specific way. Should an event occur, the agent automatically knows what to do. The c-schema also suggests how important the goal should usually be. Attention-focusing information from

the current context will also affect whether attention is focused on this response or not. In addition, a c-schema may contain a standing order governing the overall reactivity of the agent.

In Orca, event-handling information is used primarily by Event Handler, though any responses to events will be posted to the Agenda Manager. Event assessment information is in the form of fuzzy rules used by one of EH's rule-based systems.

5.3.4. *Focusing attention*

The context structure provides information from c-schemas about the relative priorities of goals that might become active in the context when their importance is likely to be different than in other contexts. As with event-assessment information, the information can be as simple as an estimate of the importance of the goal, or the importance of a goal can depend on other features, including other goals that are also active. Standing orders also affect attention-focusing, since they can suggest deactivating a goal in the context.

In Orca, attention-focusing information is provided in the form of rules that Agenda Manager can use to determine the priority of goals. This was done in a simple way in MEDIC and is being implemented in Orca by giving AM these rules for use in its internal fuzzy rule-based system. Standing orders to deactivate a goal will also be handled by AM.

5.3.5. *Selecting actions*

The context structure suggests actions appropriate for goals in the context. When the agent wants to achieve a goal, it uses these suggestions as starting points for searching for what is to be done. In a schema-based reasoner such as Orca or MEDIC, these are suggestions of p-schemas to interpret. The suggestion gives an entry point into long-term memory, from which the agent can fine-tune the action by looking for specializations fitting the particulars of the situation. This can short-circuit much of the work of p-schema retrieval. In other kinds of reasoners, these suggestions would be for plans, rules or whatever form the procedural knowledge takes.

5.4. CHANGING THE CONTEXT

An important problem for CMB is a detecting when the context has changed. Our approach to this problem is to allow c-schemas to contain information about events that signal a context change. For example, the c-schema "in a harbor" may predict a context change if the AUV passes the harbor's mouth. When these events are noticed, then the context manager can begin the process of determining what the new context is. We assume that such triggering events can be given to the agent for many contexts it knows about. Learning these events on its own, however, may be quite difficult.

However, it is likely that it will not always be possible to detect or unambiguously identify such events. Consequently, the context manager should periodically re-examine the world, probing memory for new c-schemas that are evoked and judging how well existing c-schemas fit the situation.

We have yet to implement this portion of CMB in Orca. This will be an important topic for future research.

6. Related work

**<*

Early work in AI related to context included research in planning and top-down predictions in computer vision. The former typically encoded contextual information as preconditions of operators or macro operators (e.g. Fikes & Nilsson, 1971). This has the unfortunate side-effects of redundantly encoding the knowledge and having no explicit identification of the context. Most expert systems treat context similarly, by encoding contextual information in their rules' antecedents! As we have discussed, an agent benefits from having explicit representations of contextual knowledge, since this allows reasoning about the context itself. An early approach to this problem was the CENTAUR project (Aikins, 1980), though only one kind of context (the consultation) was represented, and then for very limited purposes. One can also think of MDX (Chandrasekaran, Gomez, Mittal & Smith, 1979) as having diagnostic contexts, since its "concepts" served to partition its rules along context-specific lines.

In the past few years, interest in and work on context has been increasing in artificial intelligence and related fields, as evidenced by recent workshops at the major AI conferences, and a recent conference, CONTEXT-97 (Brezillon & Cavalcanti, 1997), focused solely on context. One very active area is formal representation of context. This work mostly follows from McCarthy's (1987) work on formalization of context in logic. Another area of strong interest is natural language processing/computational linguistics. This area, for example, accounts for 12 of the 33 papers in the CONTEXT-97 conference. Work has also been done on cognitive models of context. This includes Kokinov's (1994) DUAL cognitive architecture, Kolodner's (1984) model of human episodic memory (the basis for CMB's memory and one of the origins of c-schemas), and Oztiirk and Aamodt's (1997) case-based diagnostic program. Feltovich *et al.* (1984), considering the development of human diagnostic expertise, proposed the concept of "disease models" organized hierarchically; this is similar in some ways to our hierarchical arrangement of contextual schemas in long-term memory.

There has also been some work focused on context-sensitive behavior in real-world applications. One of the earliest efforts was the author's work on MEDIC (Turner, 1989a, 1994), of which Orca is a direct descendent. MEDIC was a medical diagnostic reasoner developed at the Georgia Institute of Technology. The hypothesis developed in MEDIC was that diagnostic consultations could be approached by *adaptive reasoning*, a kind of context-sensitive, reactive planning.

In MEDIC, contextual schemas were monolithic structures representing entire problem-solving contexts, or consultations. As the consultation progressed, whatever c-schema was in use changed based on the features of the evolving situation. A particular case might progress from being represented as a "medical consultation", then as a "cardiopulmonary consultation" and finally as a "cardiopulmonary consultation in which the patient is an alcoholic" as the program's understanding of the situation evolved during information gathering. C-schemas contained information about goals, events and actions that guided the program's diagnostic behavior.

^t The "context" mechanism of the MYCIN family of expert systems (Shortliffe, 1976) was rudimentary and was not the kind of context we are concerned with here.

CMB was not fully designed nor implemented in MEDIC. There was no context manager *per se*, context assessment was rudimentary and there was no notion of context-dependent concept meanings. In addition, medical diagnosis is not a particularly rich domain for developing a model of context-sensitive reasoning, since it has a limited range of contexts and of behavior. To fully develop adaptive reasoning in general and context-mediated behavior in particular, a richer domain was needed, such as AUV control.

Other work on using context in real-world applications includes the work of Brezillon, Gentile, Saker and Secron (1997) on context's impact on problem-solving for subway incident handling; Guha and Lenat's (1990) work on common sense reasoning and Pinto, Stephens and Bonnell's (1995) work on geographic reasoning in Cyc; Ozturk and Aamodt's (1997) work on medical diagnosis; E. Turner's (1989) work on conversational control; and Vamos' (1995) work on modeling domain expert knowledge, which resulted in patterns similar to very simple contextual schemas.

Our work differs from these authors' in several ways. First, the range of behavior affected by context in CMB is broader, since all aspects of the agent's behavior is context-dependent. Second, CMB allows the agent to automatically behave appropriately once a context is recognized. In most of the other approaches, reasoning about the context of decision-making must occur throughout the time spent in the context. Third, CMB explicitly represents contexts as objects in their own right, which facilitates reasoning about the context, acquiring knowledge, and handling novel contexts by merging existing ones. Except for the Guha and Lenat (1990) work on microtheories, most of the other approaches do not represent contexts explicitly.

7. Conclusion and future work

Context-sensitive behavior is critical for any intelligent agent, natural or artificial. In this paper, we have described context-mediated behavior, an approach to context-sensitive behavior for artificial, intelligent and autonomous agents.

Context-mediated behavior has several important features. First, it makes use of explicitly represented contextual knowledge, which allows the agent to reason about the context it is in. It can examine alternative contexts it knows about to determine which best characterizes its current situation. It can then use what it knows about the context to make predictions, interpret sensory information, determine what concepts mean in the context and decide how to behave.

Second, CMB provides *automatic* context-sensitive behavior. Knowledge about how to behave is automatically brought to mind when the context is recognized. Reasoning effort is expended when the context changes in order to save time when the contextual knowledge is needed. This can be very important for real-world agents, since often they must act and react rapidly.

Third, CMB allows the use of multiple contextual schemas to represent the current context. This reduces the number of contexts that must be represented and allows the agent to handle many novel contexts by blending information about contexts the situation resembles.

Fourth, CMB is encapsulated within a module, ECHO, that is not necessarily tied to one reasoner. ECHO is being implemented to allow the agent's other reasoning

modules to register their information needs with it. When the context changes, [^] will use the registry to determine what to send to whom. This frees ECHO from needing to be intimately aware of the reasoner it is part of. This, in turn, means that ECHO can potentially be used with reasoners other than Orca. For example, we can imagine its use with a neural network controller or a standard rule-based system; in these cases, the c-schemas would contain weights and rules, respectively, that are appropriate for the context.

There is much left to be done in the future on CMB. In the near term, the implementation of a complete version of ECHO in Orca will be finished and evaluated in simulation tests and, ultimately, in in-water tests aboard AUVs. We will examine the issue of how ECHO can acquire contextual knowledge. We believe using c-schemas will facilitate knowledge acquisition from humans, and we will also begin to look at how ECHO can learn its own c-schemas from experience. We will also explore the need for "meta" and domain-independent c-schemas, as described earlier, to represent strategies and contexts such as "in an unfamiliar context" or "in a learning context".

We will also begin to address moving CMB beyond Orca. One area is using ECHO with other kinds of reasoners. A second has to do with our research (e.g. R. Turner, E. Turner & Blidberg, 1996) in cooperative distributed problem-solving (CDPS). We envision multiple, cooperating agents, each controlled by a context-sensitive reasoner using CMB. This gives rise to interesting issues, such as how to represent contexts involving cooperative problem solving and how agents can communicate about, and agree upon, their shared context.

The author would like to thank the United States National Science Foundation for its support of this work via grant number BES-9696044. The author would also like to thank Elise Turner and the other members of the University of Maine Cooperative Distributed Problem Solving (CDPS) research group for their comments during the work and the preparation of this paper and Patrick Brezillon for his very helpful comments on an earlier draft of this paper.

References

- AIKINS, J. S. (1980). *Prototypes and production rules: a knowledge representation for computer consultations*. Ph.D. Thesis, Stanford University.
- BLIDBERG, D. R., TURNER, R. M. & CHAPPELL, S. G. (1991). Autonomous underwater vehicles: current activities and research opportunities. *Robotics and Autonomous Systems*, 7, 139-150.
- BREZILLON, P. & CAVALCANTI, M., Eds (1997). *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*, Rio de Janeiro, Brazil. (For information on obtaining copies, contact Marcos Cavalcanti, COPPE/UFRJ (F109), CP 68507, 21945-970, Rio de Janeiro, Brazil.)
- BREZILLON, P., GENTILE, C., SAKER, I. & SECRON, M. (1997). SART: a system for supporting operators with contextual knowledge. In *Proceedings of the 1997 International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*.
- CHANDRASEKARAN, B., GOMEZ, F., MITTAL, S. & SMITH, J. (1979). An approach to medical diagnosis based on conceptual structures. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, Stanford, CA.
- DRAHMER, E. & PIWEK, P. (1997). Exploiting context for filling presuppositional gaps. In *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*. Rio de Janeiro, Brazil: COPPE/UFRJ.

- FELTOVICH, P. J., JOHNSON, P. E., MOLLER, J. A. & SWANSON, D. B. (1984). LCS: the role and development of medical knowledge and diagnostic expertise. In W. J. CLANCEY, and E. H. SHORTLIFFE, Eds. *Readings in Medical Artificial Intelligence*, pp. 275-319. Reading, MA: Addison-Wesley.
- FERSTL, E. C. (1994). Context effects in syntactic ambiguity resolution: the location of presuppositional phrase attachment. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society*. Hillsdale, NJ, USA: Lawrence Erlbaum Associates.
- FIKES, R. E. & NILSSON, N. J. (1971). STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2, 189-208.
- GLASS, A. N. & HOLYOAK, K. J. (1986). *Cognition*, 2nd edn. New York: Random House.
- GUHA, R. & LENAT, D. B. (1990). Cyc: a midterm report. *AI Magazine*, 11, 32-59.
- HOLTGRAVES, T. (1994). Communication in context: effects of speaker status on the comprehension of indirect requests. *Journal of Experimental Psychology*, 20, 1205-1218.
- KoKINOV, B. N. (1994). The context-sensitive cognitive architecture DUAL. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, Atlanta, GA.
- KoLODNER, J. L. (1984). *Retrieval and Organizational Strategies in Conceptual Memory*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- LEVINTON, J. S. (1982). *Marine Ecology*. Englewood Cliffs, NJ: Prentice-Hall.
- MCCARTHY, J. (1987). Generality in artificial intelligence. *Communications of the ACM*, 30, 1030-1035.
- MANTOVANI, G. (1996). Social context in HCI: a new framework for mental models, cooperation and communication. *Cognitive Science*, 20, 237-269.
- MELLERS, B. A. & COOKE, A. D. J. (1996). The role of task and context in preference measurements. *Psychological Science*, 7, 76-82.
- MILLER, R. A., POPE Jr, H. E. & MYERS, J. D. (1982). INTERNIST-1, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307, 468-476.
- OZTURK, P. & AAMODT, A. (1997). Towards a model of context for case-based diagnostic problem solving. In *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*. Rio de Janeiro, Brazil: COPPE/UFRJ.
- PEVTZOW, R. & GOLDSTONE, R. L. (1994). Categorization and the parsing of objects. In *Proceedings of the 16th Annual Conference of the Cognitive Science Society*, Hillsdale, NJ, USA: Lawrence Erlbaum Associates.
- PINTO, N. B., STEPHENS, L. M. & BONNELL, R. D. (1995). Organizing domain theories for geographic reasoning. In *Working Notes of the IJCAI-95 Workshop on Modelling Context in Knowledge Representation and Reasoning*. Montreal, Canada.
- SHORTLIFFE, E. H. (1976). *Computer-based Medical Consultations: MYCIN*. New York: Elsevier.
- TURNER, E. H. & CULLINGFORD, R. E. (1989). Using conversation MOPs in natural language interfaces. *Discourse Processes*, 12, 63-91.
- TURNER, R., TURNER, E. & BLIDBERG, D. (1996). Organization and reorganization of autonomous oceanographic sampling networks. In *Proceedings of the IEEE Symposium on Autonomous Underwater Vehicle Technology*, Monterey, CA.
- TURNER, R. M. (1989c). Using schemas for diagnosis. *Computer Methods and Programs in Biomedicine*, 30, 199-208.
- TURNER, R. M. (1989*). When reactive planning is not enough: Using contextual schemas to react appropriately to environmental change. In *Proceedings of the 11th Annual Conference of the Cognitive Science Society*, pp. 940-947. Detroit, MI.
- TURNER, R. M. (1994). *Adaptive Reasoning for Real-World Problems: A schema-Based Approach*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- TURNER, R. M. (1995). Intelligent control of autonomous underwater vehicles: the Orca project. In *Proceedings of the IEEE International Conference of Systems, Man, and Cybernetics*. Vancouver, Canada.
- TURNER, R. M. (1997). Determining the context-dependent meaning of the fuzzy subsets. In *Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*. Rio de Janeiro, Brazil: COPPE/UFRJ.

- TVERSKY, A. & KAHNEMAN, D. (1974). Judgments under uncertainty: heuristics and biases. *Science*, 185, 1124-1131.
- VAMOS, T. (1995). A strategy of knowledge representation for uncertain problems: modeling domain expert knowledge with patterns. *IEEE Transactions on Systems, Man, and Cybernetics*, 25, 1365-1370.
- ZADEH, L. A. (1965). Fuzzy sets. *Information and Control*, 8, 338-353.
- ZADEH, L. A. (1994). Fuzzy logic, neural networks and soft computing. *Communications of the ACM*, 37, 77-84.