Facing Uncertainty and Consequence in Context-Aware Systems: towards an Argumentation Approach

Seng W. Loke
School of Computer Science and Software Engineering
Monash University, Vic 3145, Australia
swloke@csse.monash.edu.au

ABSTRACT
This position paper proposes argumentation structures for automated reasoning in context-aware systems, for design of context-aware behaviour, for generating explanations of system actions to users, and for more expressive rules for user-programming of context-aware systems.

Keywords
context-aware pervasive computing, argumentation, reasoning, uncertainty, consequence, user-programming

INTRODUCTION

Due to the huge disparity between the physical world and the electronic world, one of the challenges in context-aware applications is to accurately abstract and model real world situations in the computer. Context-aware systems need to interpret and recognize situations with limited, uncertain, possibly inconsistent, and incomplete information in order to choose appropriate actions [2,3,7]. However, context-aware systems should take actions that not only suit the context but also try to avoid disastrous consequences. In this paper, we highlight the need to model uncertainty in what a context-aware system believes is happening in the real world (e.g., in what the system perceives is the context of its user), and the consequences of the intended actions of such a system. We contend that an understanding of the consequences of actions can provide additional safe-guarding, useful when uncertainty in perceiving context is inevitable. There are many formalisms for dealing with uncertainty and various taxonomies of uncertainty concepts [6]. We explore argumentation as a reasoning mechanism in context-aware systems, and more expressive rules for user-programming of context-aware systems. Also, argumentation structures can be used as a design artifact, for generating explanations (or justifications) for actions, and for debugging context-aware systems.

UNCERTAINTY OF CONTEXT AND CONSEQUENCES OF ACTION

A simple model of a context-aware system is one which senses the context of a user and based on the context of the user, decides whether to take a particular action. Such a system might be modeled via a set of rules mapping context to beliefs (i.e. what the system thinks is the context) and beliefs to actions. In the case where there is varying degrees of uncertainty in perceiving context, the system can be built to take action only when there is a high certainty in perceiving context. However, such behaviour might be unnecessarily cautious, especially when the consequences of an action taken might not be severe, i.e., for example, the effects are reversible, and/or there are reasonable compensatory actions – the potential benefits of the action outweighs any potential costs. Also, the actions might have no effect on the physical world. Table 1 below shows a simple decision analysis of a system where measures of uncertainty in perceiving context and consequences of actions to be taken are considered. For simplicity, we assume that the analysis is considered for a system whose behaviour is modeled by rules of the form:

IF Context THEN TAKE Action.

The action is only withheld when the system is highly uncertain of what is perceived of the context and the action has potentially severe consequences; otherwise, the system takes the action.

Uncertainty in Context gathered	**Consequence of Actions**	**Decision**
Low	Drastic	Action taken
High	Drastic	Action withheld
Low	Light	Action taken
High	Light	Action taken

Table 1. A simple decision matrix for actions.

Effectively, we are modeling the system with rules of the form:

IF Uncertainty(Context) < U AND Severity(Action) < S THEN DO Action

where Uncertainty() and Severity() are measures for uncertainty in the perceived context and severity of actions, and U and S are thresholds for the given Action. For a given action, the developer sets measures U and S as appropriate to the system's capabilities (in sensing context), the extent to which a context can be ascertained, and the severity of the action. Consider an example from [3]: "Spying a newsrack, Tom pulls his rented car to the side of the street and hops out to grab a paper. The car, recognizing the door has just closed and the engine is running, locks its doors." A problem of this scenario is perhaps that insufficient sensor inputs have been used in guessing the situation of the user. For example, seats could have an associated weighing machine to confirm that someone is seating in the car. However, if this is not feasible, another means to address the problem might be to explicitly model the consequences of the action "locking car doors", an issue which can be considered at requirements analysis or design time of the context-aware system. Understanding the severity of its intending action and the uncertainty about its user's situation, the car might then choose a less drastic action, or use other sensors (e.g., ask the user) before carrying the action out, or provide a means to the user to easily reverse (if possible) the action. One way to solve the ambiguity or uncertainty problem is by asking the user, and we admit that this will be sometimes needed. So, one extreme is to always ask the user and the other extreme is to never ask the user. Our approach of considering the extent of uncertainty of sensed context and consequences of action sits in between these two extremes and also helps the system decide when to ask the user. However, one cannot be completely exhaustive in anticipating all possible consequences - human system designers would need to judge the severity of actions.

RELEVANCE OF ARGUMENTATION FOR CONTEXT-AWARE APPLICATIONS

Argumentation theories provide a formal account of reasoning, typically in the case of arguing for or against an assertion [8]. Arguments are linked with explanations: an argument structure provides an explanation concerning the acceptance or rejection of a proposition. Additional information later obtained may augment the argument structure affecting the strength of the argument, or the explanation. Weaknesses in reasoning or difficulties can be identified via the argument structure. Arguments capture plausible reasoning: the degree of credibility of a proposition can be assessed via the strength of arguments for or against it. Hence, there is a notion of the grounds or justification for a conclusion made. Argument structures can be augmented as new information becomes available. As an example, Figure 1 describes an argumentation structure proposed by the philosopher Toulmin [9], where an argument comprises six components: data, modality, claim, backing, warrant, and rebuttal. The claim of the argument is the assertion being made based on the data. The warrant, supported by the backing, is the justification for making the claim based on the data. The rebuttal attacks the claim. So, a given claim might have several arguments for it and arguments against it.

Argumentation Structures as Design Artifacts in Context-Aware Systems

One use of argumentation structures such as the Toulmin structure above is as a way to represent how sensed inputs are related to beliefs about the user's situation (i.e. the perceived context) and how beliefs are justifications for actions. One way in which sensed inputs are map to perceived context and beliefs to actions is via a set of Condition-Action like rules. But an argumentation based approach provides a more comprehensive model since the sensed inputs are really merely hints (or arguments) to what the real context is. In addition, such a model will enable arguments for or against a belief to be represented and weighed, as well as arguments for or against an action to be considered. This approach is akin to argumentation-based design rationale, where argumentation is used to represent why an artifact is made a certain way. In this case, we use argumentation to represent context as justifications and explanations to users for recognized situations and for actions, and for designers in building systems. Figure 1 shows the example of the Toulmin argument for Tom being currently in a meeting in room F. For the same claim, with different sensors, different data and warrants might be

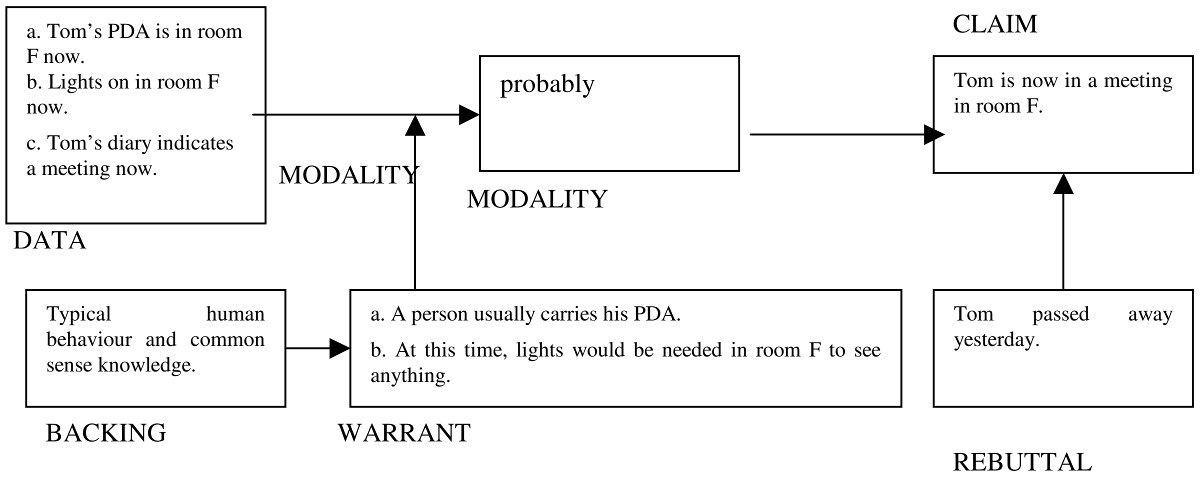
a. Tom's PDA is in room F now.
b. Lights on in room F now.
c. Tom's diary indicates a meeting now.
DATA
MODALITY
probably
MODALITY
CLAIM
Tom is now in a meeting in room F.
Typical human behaviour and common sense knowledge.
BACKING
a. A person usually carries his PDA.
b. At this time, lights would be needed in room F to see anything.
WARRANT
Tom passed away yesterday.
REBUTTAL

Figure 1. Toulmin argument for "Tom is in a meeting in room F now."

employed. There are numerous methods that can be used to infer the claim from the data, from fuzzy inference rules, deduction in first order logic, causal reasoning, neural networks to Bayesian techniques. The argumentation framework can provide a structure in which different methods might be integrated.

Moreover, the structure can be used as a basis for identifying weaknesses in a context-aware system design. Sources of weaknesses could either be due to the data (e.g., inadequate data or inadequate sensors, ambiguity in sensor readings, imprecision or errors as noted in [5]), the warrant for the claim given the data (the inference procedure used) including, for example, the correctness of inference rules, and the backing for the warrant (e.g., the commonsense background knowledge assumed).

Argumentation for Automated Reasoning about Context and Actions

While argumentation structures can be used in the design of context-aware systems, a formal representation of the arguments can be used to facilitate automated reasoning. Via declarative representations of the arguments, a system can generate explanations (perhaps only when asked) for its actions and for its inaction, and this facilitates the debugging of the context-aware system. For more complex systems, such a representation can be used to test the system via simulations – to determine what the system would do under varying circumstances. Now, we illustrate how reasoning about context and actions can be done using a formal system based on LA and LV [4].

We have first a database Δ comprising arguments that map sensory inputs to beliefs about the user's context. Each argument is of the form: $(B:G:W)$, where B is the belief, G is the grounds for believing B, and W is a measure of (un)certainty. B is a formula and G is a set of formulae (similar to propositional logic) and W is a symbol from a dictionary D. We let D = {+,-} here where "+" means the grounds support the belief B and "-" means the grounds opposes B. We treat Δ as containing mappings from sensory inputs to beliefs and from beliefs to other beliefs. This means that, given some sensory inputs **I** acquired by the system, we match **I** with every argument in Δ to produce the set of arguments δ whose grounds are contained in **I**, given by $\delta = \{(B:G:W) \mid \mathbf{I} \supseteq G \text{ and } (B:G:W) \in \Delta\}$. This means that δ contains the beliefs justified by the sensory inputs, that is, δ represents the perceived context of the user. The system might perceive the user to be in more than one possible context (represented by conflicting beliefs, for example). Here, for simplicity, suppose that Δ is designed such that all the arguments in δ are for a given belief B, i.e. every argument in δ either supports B or opposes B (this association, we denoted by δ_B). As in [4], we can compute the validity of B by applying the flattening function flat() on δ_B which combines all the arguments (for or against B) into a single number, where we define flat() by subtracting the number of opposing arguments from the number of supporting arguments, as follows:

$$\text{flat}(\delta_B) = |\{(B:G:W) \mid W = \text{`+'} \text{ and } (B:G:W) \in \delta_B\}| - |\{(B:G:W) \mid W = \text{`-'} \text{ and } (B:G:W) \in \delta_B\}|$$

This computation gives a quantitative measure of the certainty of perceived context with respect to the inputs **I**. In addition, we have

(1) a database Λ of rules which map beliefs (i.e. perceived user's context) to actions, where each rule is of the form ($B \Rightarrow A$: t) for a belief B, an action A and a threshold value t such that if $\text{flat}(\delta_B) > t$ then A will be selected as a potential action, which are essentially condition-action rules,

(2) a database Σ of rules which associates each action with formulae representing states of the world, where each rule is of the form ($A \mapsto C$:G) for an action A, state of the world C (i.e. C represents the consequences of the action A), and grounds G (i.e., the rule states that given assumptions G, the consequence of doing A is the result C), and

(3) a database υ of arguments representing the valuations of states of the world. Each argument in υ is of the form (C:G:V), where G is the grounds for saying that C has valuation V, where V might come from a dictionary (say D) above. The meaning of (C:G:+) is then that C is a favourable state of the world assuming G, and (C:G: -) means that C is not a favourable state of the world assuming G. (Note that other dictionaries might be used depending on the application semantics, and instead of having V explicitly in the database, we might have an algorithm to compute V based on the availability of compensation actions, the reversibility of the action, etc).

The idea is that for each rule of Λ where $\text{flat}(\delta_B) > t$ applies, a possible action is selected. Then, for each possible action A, we look at Σ to find the consequence C of A, using the rule where assumptions G are included in inputs **I**. Once we found the consequence C, we then find its valuation V using the knowledge from υ (with arguments whose grounds are contained in **I**). In summary, what we have is then a measure of the (un)certainty of the perceived context (as given by $\text{flat}(\delta_B)$) and a measure of the consequences (or severity) of taking a prescribed action A (as given by V). We can then use a rule such as the following as mentioned earlier to decide whether to take the action or not:

IF Uncertainty(Context) < U and Severity(Action) < S THEN DO Action

By adjusting the thresholds U and S in the rules, one can effectively tune the system from being highly conservative (always asking the user) to being highly autonomous but presumptuous. Additional rules can be added such as the following might be added which tells the system when to ask the user for approval regarding a given action:

IF Uncertainty(Context) > U' and Severity(Action) > S' THEN DO Ask-User

For users, qualitative tagging of situations (attributed to the certainty levels of context characterizing the situation) [1] and can be employed in rules, such as *certain* – evidence for the situation are adequate, *presumed* – evidence for the situation are strong but not adequate to be certain, *suggested* – evidence present but not strong, and *possible* – no evidence present but no evidence against the situation either.

IF certain(Context) and Severity(Action) < S'' THEN

DO Action

Or
IF presumed(Context) or Severity(Action) > S''' THEN
DO Ask-User
This is only one example of applying an uncertainty formalism to context-aware systems, which provides a richness of representation more than simple condition-action rules.

SUMMARY AND FUTURE WORK

Uncertainty in a context-aware system might not be avoidable given its limitations in sensing and reasoning capabilities – often humans cannot avoid such uncertainty too. What such a system perceives of the world might merely be a claim (a "good guess") which it can be made to support. We have argued for the use of an argumentation paradigm, which takes into account uncertainty in perceiving context and consequences of actions for

(1) user-programming (with rules more expressive than simple situation-action rules) context-aware systems,

(2) providing design artifacts for system context and actions,

(3) automated reasoning, and

(4) generating explanations for system actions and debugging.

The argumentation framework is general and can include different forms of reasoning. Different procedures (or warrant) to go from data to claim can be investigated. It can also be a means to combine different approaches in a hybrid manner. For example, given some data (e.g., sensor readings) one procedure produces a claim (e.g., an inferred context) with a measure of certainty and another procedure might produce a (possibly the same) claim with a different measure of certainty. Arbitration or consolidation between different such claims or different views about the same claim can then be carried out in an argumentation framework (in the spirit of the flat() function mentioned earlier).

There is much future work ahead. We have only sketched what is possible. Our ongoing work involves adding such reasoning and explanation capabilities into several context-aware applications. Various argumentation structures for designing context-aware applications can be investigated.

REFERENCES

1. Barnden, J.A. Uncertainty and Conflict Handling in the ATT-Meta Context-Based System for Metaphorical Reasoning. In: Proceedings of the 3rd International Conference on Modeling and Using Context. LNAI, Vol. 2116. Springer-Verlag (2001)
2. Dey, A.K., Mankoff, J., Abowd, G., and Carter, S. Distributed Mediation of Ambiguous Context in Aware Environments. In: Proceedings of the 15th Annual Symposium on User Interface Software and Technology (UIST 2002), Paris, France (2002) 121-130
3. Erickson, T. Some Problems with the Notion of Context-Aware Computing. In: Communications of the ACM 45(2) (2002) 102-104
4. Fox J., Parsons S. Arguing about Beliefs and Actions. In: Hunter, A. and Parsons, S. (eds.): Applications of Uncertainty Formalisms. LNAI, Springer (1998)
5. Henricksen, K., and Indulska, J. Modelling and Using Imperfect Context Information. In: Proceedings of the IEEE Annual Conference on Pervasive Computing and Communications Workshops, Orlando, Florida, USA (2004) 33-37
6. Parsons, S. Qualitative Methods for Reasoning Under Uncertainty. MIT Press (2001)
7. Shafer, S.A.N., Brumitt, B., and Cadiz, J.J. Interaction Issues in Context-Aware Intelligent Environments. In: Interactions (to appear).
8. Stranieri, A., Zeleznikow, J., and Yearwood, J. Argumentation Structures that Integrate Dialectical and Non-Dialectical Reasoning. In: The Knowledge Engineering Review 16(4) (2001) 331-348
9. Toulmin, S. The Uses of Argument. Cambridge University Press (1959)