
CLASS Notes (March 25, 2008) from Vincent

----

1. Comments on the statement that "context-awareness is only useful when it can automatically trigger an action."

Group #1: Disagree, because a wrongly-executed command causes frustration. Also, it is notable that only 100% accuracy will be enough, as even an error rate of only 1% will be very annoying (such as your computer failing).

Comments: Users are more forgiving towards humans than computers, because we have no sympathy for machines that crash, and high expectations for the machine to run without errors. Friendlier error messages will be better in that we will know the reason of the crash, having better understanding of why things fail.

Group #2:
Disagree, because context-awareness is not only about taking an action; sometimes presentation is more important.

2. Comments on the statement that "computers are bad at interpreting context data, because they lack common sense; therefore, context awareness must involve humans in the loop."

Group #1:
Common sense is very broad, involving human emotions/feelings, things which will demand a very good AI to be accurate.

Computers are good at a few things (e.g. absolute time, temperature, etc.), but not generally good at everything. For example, when someone walks into the room, we can automatically let the system open the door, turn on the lights, etc. But the computer doesn't know what the person intends to do, so it doesn't know whether to turn on the projector or not.

We would prefer human-in-the-loop for many things, because humans want to be in control, and also because it is too difficult for computers to determine human intention.

Group #2:
Generally agree; but the statement is too strong, using the phrase "must involve". Counterexamples are temperature control (where human perception is not better than computers), and ETC (where paying the toll is a distraction to humans).

So we can revise the phrase “must” to "often".

Trying to generalize conditions in which a computer is helpful:
Controlling the environment, create good environment for humans to live; or context which computers can inference from, but humans can't sense (i.e. when the sensing capability of computers is better, it is wise for humans to be taken out of the loop; i.e. involve humans only when humans are more accurate).

So the tradeoff between humans-in-the-loop and humans-out-of-the-loop are based on error tolerance for the problem, and the amount of user effort allowed.

Suggestion: split the "loop" into subloops (auto-execution) and superloops (human supervision).

Putting humans in the loop may not always be good, as humans can make mistakes; but humans can take responsibility and respond to the interaction immediately. Ultimately, technology is for humans, so it must always involve humans.

3. How to categorize/classify context-awareness.

Group #3:

Use “purpose” to categorize context-aware apps:
a) extra information
b) enhance emotional status; e.g. Lovers' cup
c) enhance physical status
d) no practical purpose (just for fun)

Rationale behind this kind of classification: practicality of the entity.

Group #4:
a) involves user or not?
No (doesn't involve users): e.g. automatic water sprinkler
Yes (involves users): (involving users doesn't mean that users always interact with the system, it may be that the system is doing something for/to the user)

b) supervised or not? - (configurability)
Yes (supervised): the system is automatic, but the user can customize the system
No (unsupervised): after the system is produced, we cannot configure the system any more.

c) influences user behavior or not?
Yes (tries to make users smarter): tries to educate people
No: just do things better for us

How to evaluate a proposed classification? Solution: is it meaningful for designers?
Otherwise the classification is just a "group" and not useful for designers.

4. Can we use AR (augmented reality) to solve the invisibility problem (where users become uncomfortable if they don’t know what is going one, i.e. the rationale for the application is invisible)?

No; since a HMD (head mounted-display) requires too much user attention, is not natural, and annoying.

Suggestion: put visibility in the peripheral of the physical environment.

Comment: some kind of AR/visibility can be useful for debugging (e.g. turns on HMD only when there is no expected action/response from the system).

Comment: wearable devices are not useful, since research shows that wearable devices will influence results.