I'm the Operator of my Pocket Computator: Dangers of Context Automation

Pertti Huuskonen

Nokia Research Center
Visiokatu 1, 33720 Tampere, Finland
pertti.huuskonen@nokia.com

These are good times for context awareness. Millions of people roam the planet with mobile devices that are capable of context sensing. Phones, media players, cameras and navigators contain embedded sensors for location, acceleration, radios, sound, or light, among other properties. They encounter numerous systems with embedded computing and networking, and, increasingly, ambient intelligence. The people deal with thousands of web services that aggregate their contextual data, such as presence, location, activities, and media consumption. Together, these various systems create an immense global context awareness platform that promises better services, better user experience, and better information filtering, among other benefits.

To be useful, context awareness has to work in an automatic mode, working as expected without user intervention. Only in rare situations should the users need to actively control the systems, for instance to teach them new context functions or deal with errors. With the global context platform there is very good potential for such automation. But is it good enough? What are the dangers?

Automation in general may work beautifully given the right conditions. When the outside world stays within foreseen limits, the designers of automated systems can anticipate the needed functions. As a result, the users can focus on other things. However, context awareness is a difficult area for automation. Context data is often noisy, ambiguous, just plain messy. What's worse, contextual situations can be very complex since they often deal with the real world, which can be very complex indeed for computers. The resulting mistakes can lead to severe problems.

When something goes wrong, the users of context aware systems (like any automated system) suddenly need to engage in elaborate fault diagnosis. What was the cause of the malfunction? Was it me or the system? Was something damaged? Was the configuration wrong? Did someone hack into the system? Are we in danger? Can we trust the system for continued operation?

Consider the case of mobile phone in a church. You downloaded an automatic macro function that turns your phone to silent mode when you enter a meeting. The function has worked beautifully in office settings, so you intuitively believe it also works in other places where people come together. However, when the phone suddenly rings in the middle of a sacred ceremony, you pay for the error in terms of social embarrassment. This is clearly not a life-critical event, but would still be perceived negatively. You will wonder why it happened -- did the phone not notice that there were hundreds of people present? Or did the loud church organ fool it? What are the other places where it could make you lose your face?

We have seen this all before. Similar issues have been found throughout the history of automated technologies in such diverse fields as process control, robotics, aircraft control, network management, and others. A great deal of research, analysis and design has sought to make automation as reliable as possible, and make fault diagnosis and recovery and smooth as possible. However, such systems place great demands to their operators. These are experts with years of experience in monitoring, controlling, planning and diagnosing the systems they supervise. Most safety critical systems (for instance, aircraft) need human pilots for taking over when automatics fail.

There is a danger that users of context aware systems may have to assume the role of the process operator, supervising the context automation and taking action when it fails. Given the speed that today's Internet and mobile devices are turning into context aware systems, this danger is now imminent. Though most people would never imagine of becoming operators, they might implicitly fall into such a situation when they engage context automation in their devices. Unless the automation is 100% reliable (which it will not be), they will have to gain some kind of experience on when it works and when not, either by education, by automated help, by peer support, or -- most likely -- by trial and error.

This danger can be alleviated, to a degree. Better UIs can make operation more transparent and problems easier to deal with. Careful system design can make systems more aware of their limitations. The choice of less risky application domains (e.g. gaming) can lessen the risks involved. However, as long as context aware systems need to deal with the complexity of our world, they are bound to fail in complex ways.

This talk explores the potentials and risks of context awareness, draws analogies from industrial automation, and suggests some avenues for further work.