Progressive automation to gain appropriate trust in management automation systems

Ralf Koenig[1], Elaine Wong[2] and Gregoire Danoy[3]

[1] LMU Munich, Munich Network Management Team, Germany
koenig@mnm-team.org
[2] EADS, Singapore
elaine.wong.kl@gmail.com
[3] University of Luxembourg, Luxembourg
gregoire.danoy@uni.lu

Abstract. This paper summarizes the work of the working group on "Progressive Automation" at Dagstuhl-Seminar 09201 "Self-Healing and Self-Adaptive Systems", May 5th – May 10th 2009.

In many management automation scenarios, the involved people (both developers and users of the MAS) tend to not understand all the effects of automation well. This is a problem, because both undertrust and overtrust have negative effects. Progressive automation will introduce automation in a way that the positive effects of automation are leveraged quickly, while the potential negative effects are reduced. As a consequence, progressive automation will cause people to build up the right amount of trust to the automation system.

Keywords. systems management, network management, aircraft maintenance management, automation, self-managing systems, autonomic systems, trust

For many repetitive management tasks, both simple and complex, human managers long for machine support. Users may also be requested from outside to use automation. Thus, management automation systems (MAS) can be found in many application domains, such as systems management automation in data centers, network management automation in network infrastructures (network management), aircraft maintenance management automation in with airline operators—just to name the domains, where the members of this working group have a background in.

However, in many of these scenarios the involved people (both developers and users of the MAS) tend to not understand all the effects of automation well. A typical confusion is about which parameters users still have to take into account and how they can influence the MAS. They also can rarely ask the MAS to explain its operations. As a result the level of trust may be inappropriate for existing MAS.

This is a problem, because both undertrust and overtrust have negative effects: on the one hand, undertrust reduces the benefits of positive effects of

Dagstuhl Seminar Proceedings 09201
Combinatorial Scientific Computing
http://drops.dagstuhl.de/opus/volltexte/2009/2098

automation as the automation is not adopted caused by a lack of trust. On the other hand, overtrust is likely to underestimate the potential negative effects of automation, as automation being deployed too quickly or in an immature state can be a serious threat. So far, appropriate trust has not been a design issue for MAS in the different domains.

The design of management automation systems itself needs to go beyond system properties and system implementation and also take care of interaction with human users and administrators. Interestingly, despite the diversity and differences of the necessary management decisions in the different domains, the stated problem and its common issues can be tackled in a cross-domain fashion. By aligning the various management automation systems to a simple common model, we can abstract from their domain background. Thus, we can can then make use of the similarities and use a common approach in all of them. In this generic model, *progressive automation* is one way to introduce automation in a way that the positive effects of automation are leveraged quickly, while the potential negative effects are reduced by gradually increasing automation, so that the user has a better knowledge and more experience when a high level of automation is achieved. At the time, where the MAS has to be introduced into its new environment and to its users, the level of automation is low enough so that users can actually comprehend the system's structure and behavior, benefits and risks.

Progressive automation is an approach to have people build up the appropriate amount of trust in the automation system, because the MAS will be introduced in a hardly automated, mainly manual mode with a limited set of functionality. In a learning-by-doing fashion the user will then get knowledgable about the system's structure and behavior. Structure-wise users will learn about components and data. Behavior-wise users will learn about performance, reactions to various situations, and eventually models, algorithms, compromises, influence factors, decisions and their reasons. Then, over time, the MAS's capabilities will be increased and the level of automation will raise.

The intended overall effect is that users and administrators of MAS's remain in control at all times. While essentially, they give up fine-grained control when the level of automation and the level of abstraction of management interactions are increased, they do not perceive the raising level of automation as giving away control, but rather as a welcome delegation of the details to a comprehensible automaton.

Ideally, potential negative effects of automation can thus be avoided early as misbehavior will already be observed in low automation modes. Even in such cases, users and administrators will not blame system designers and implementers for negative effects but understand why a MAS has failed and provide constructive feedback to the developers. Therefore, when entering a highly automated mode, all participating roles (designers, developers, administrators, users) have gained enough experience and the MAS has gained enough careful attention, so that faults will be highly unlikely.

We will take up the idea again in a follow-up academic paper.