On the Inherent Creativity of Self-Adaptive Systems

Simo Linkola, Niko Mäkitalo and **Tomi Männistö**
Department of Computer Science
University of Helsinki
Helsinki, Finland
{simo.linkola, niko.makitalo, tomi.mannisto}@helsinki.fi

Abstract

We argue that frameworks employed in architecting self-adaptive systems allow the system to exhibit creative behaviour, and that many of the existing self-adaptive systems operating in domains which are typically not associated with creativity are inherently creative. However, even the current state-of-the-art solutions do not fully exploit stronger forms of creative behaviour, which are required in complex environments, where the system constantly encounters fundamentally novel situations. To this end, software development necessitates a paradigm shift parallel to moving from procedural design methodology toward self-aware systems where the system adapts to its context at run time.

Introduction

Self-adaptive systems are software systems where a *base system* is monitored and adapted during runtime by another component, called a *manager*, so that the base system's operation is maintained towards its goals in changing situations (see e.g. Salehie and Tahvildari, 2009). The self-adaptive systems have been studied extensively in the software engineering and architectures (see e.g. Kephart and Chess, 2003; Garlan et al., 2004; Kounev et al., 2017).

We interconnect the existing work on self-adaptive systems and computational creativity (cf. Colton and Wiggins (2012)). Particularly, we consider a host of models where a system may explore new and/or assess old (and new) adaptations at run time. We separate different types of creative adaptations and argue that a self-adaptive system must be *self-aware* (Kounev et al., 2017) for stronger forms of creativity. For implementing creative self-adaptivity, we illustrate an example control flow which extends MAPE-K loop (Kephart and Chess, 2003). The early results using an example self-adaptive system support the usefulness of explicitly considering the self-adaptive system's creativity.

Creativity in Self-Adaptive Systems

Assessing a system's creativity depends on the adopted definition of creativity (Jordanous, 2012) and the assessment perspective (Jordanous, 2016). We adopt the standard definition: *creativity is the ability to produce novel and valuable ideas or artefacts* (Boden, 1992; Runco and Jaeger, 2012), in our case adaptations[1]. We extend the definition with the notion of intentionality (Ventura, 2017) as the creative process of a system should not be fully random (cf. creative autonomy (Jennings, 2010)).

Definition. *Creative self-adaptivity is the ability of the system to intentionally adapt in ways which are valuable and novel.*

To this end, the designer needs to provide the self-adaptive system means to appear creative by relaxing the soft constraints on the adaptive behaviour, increasing flexibility. Critically, the relaxed adaptive freedom should not result in erratic behaviour. The system should seek creative adaptations only when it reasons that it does not know of an acceptable solution to the current (or expected future) situation.

Prominently, to support creative behaviour in complex environments, we need to engineer on-going learning capabilities to the system, i.e. *the system needs to be self-aware* (Agarwal et al., 2009; Kounev et al., 2017).

In this paper, novelty, value and intentionality are inspected and argued through the system's internal reasoning processes. However, as the system's behaviour should reflect its higher-level design goals, our presumption is that strengthening system's own understanding of its creative and adaptive process will reflect in its designer granting more creativity to it.

Intentionality of the system's adaptation *process* is typically well justified in a self-adaptive system as the system adapts only when it perceives a clear reason to do so. However, if the system adapts only reactively, the situation may change before the adaptation completes, ultimately requiring different adaptation strategy (cf. Moreno et al. (2015)). For stronger intentionality, the system needs to be able to adapt proactively and verify candidate adaptations at run time.

Value of an adaptation corresponds to the evaluation

[1]By 'adaptation' we refer to all of the following: a configuration of the (base) system, behaviour the configuration causes in a situation, and the act of deploying the configuration in a particular situation. Ambiguities are pointed out where appropriate.

Proceedings of the 11th International Conference on Computational Creativity (ICCC'20)
ISBN: 978-989-54160-2-8

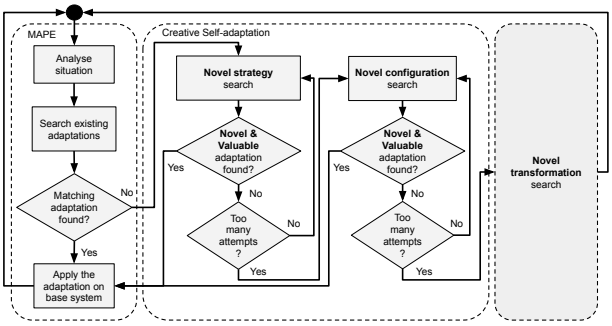

Figure 1: Example search flow for novel and valuable adaptations realised within the manager component.

of the system's behaviour in a particular situation, e.g. with respect to reliability and performance (Muccini et al., 2016). Naturally, value depends on the system's operational domain and it translates to, e.g. adhering to the hard configuration constraints of the base system (e.g. by using constraint evaluator as in Rainbow architecture (Garlan et al., 2004)), and/or adapting in a way which maximises the expected utility of the system's behaviour to some finite time horizon (e.g. Moreno et al., 2015).

Novelty of an adaptation is not commonly scrutinised in self-adaptive systems. The need for novel adaptations rises from novel situations the system encounters. The system may also explore new adaptations (or try existing ones in new situations) when it has free resources, e.g. using simulations which capture relevant aspects of the world w.r.t. its operational goals.

We separate three conceptually different senses of how an adaptation may be novel: *novel strategy, novel configuration* and *novel transformation*. Figure 1 shows an example of how a MAPE-K loop (Kephart and Chess, 2003) can be expanded to include separate processes for all novelty types in a sequence. In an actual system, however, they may be mixed within the same generic process or run in parallel.

Novel strategy means that an existing adaptation is deployed in a novel situation, which resembles combinational creativity (Boden, 1992). Novel strategies are pervasive in existing systems, as the systems are commonly engineered to analyse any encountered situation and select the most fitting adaptation for it (Salehie and Tahvildari, 2009).

Novel configuration is generated by a search-based method during the system's execution, which is analogous to exploratory creativity (Boden, 1992). The system may aim to find novel and valuable configurations for particular situations or generally well performing ones. Existing systems utilise such methods, e.g. by combining composite adaptations from elementary adaptation tactics (Moreno et al., 2015) or using AI search methods to find novel adaptations (Hadaytullah et al., 2012).

Novel transformation alters the whole system in new way, requiring same kind of mechanisms as transformational creativity (Boden, 1992; Linkola et al., 2017). In general, the transformation can be adjusting goals or any other part of the system, hence, flexible system validation and verification at run-time becomes paramount. For apt transformations, the system needs to have significant and timely understanding of its own operation and its relation to the environment, e.g. through self-modelling (Kounev et al., 2017, Chapter 9) and simulated behaviour.

Self-awareness (Kounev et al., 2017; Linkola et al., 2017) is a major enabler of creative behaviour, aiding the system to reason about value, intentionality and novelty, and act based on its reasoning. It is essential especially for novelty for inferring how and where two adaptations differ from each other, by their configuration and their behaviour (in particular situations), allowing to estimate how configuration changes affect the behaviour. Ultimately, new situations can be compared to previously encountered ones, and new configurations and transformations can be compared to previously deployed ones with an understanding of how reliable the comparison is. For example, when a robot is deployed in a new location, the system can compare its previous locations to the new one and rapidly orient itself to the new environment by selectively adapting parts of its configuration.

Particularly, transformational creativity becomes a requirement when shifting self-adaptive systems towards general AI paradigm as the system needs to be able to change its goals and be flexible in its reasoning based on its own experiences. To this end, we need rigorous software engineering practices to guide the development, e.g. to ensure safer (and ethical) behaviour (Winfield, 2014). Next, we distinguish some aspects of self-awareness which have direct influence on creative behaviour.

Self-Awareness Components

In computer systems, self-awareness can be considered as an umbrella term, enclosing a variety of different but partially overlapping aspects, many described in detail in Kounev et al. (2017). In computational creativity, these aspects have been considered with respect to metacreative systems (Linkola et al., 2017). We include three distinct awarenesses, and explicate how they contribute to the goals of creative self-adaption: *novelty*, *value*, and *intentionality*:

Goal-awareness ensures that the goals are coherent after their alteration to support intentional and valuable adaptations. The task is challenging since no component may have a direct knowledge of every goal. For this reason, the system can maintain a *goal model* which has an immediate influence on the evaluation function guiding the adaptation search. Thus, the goal changes reflected in the evaluation function bias the subsystems' behaviour towards adaptations conforming to the modified goals. Understanding the novelty of new goals allows the system to find the most similar previous goals

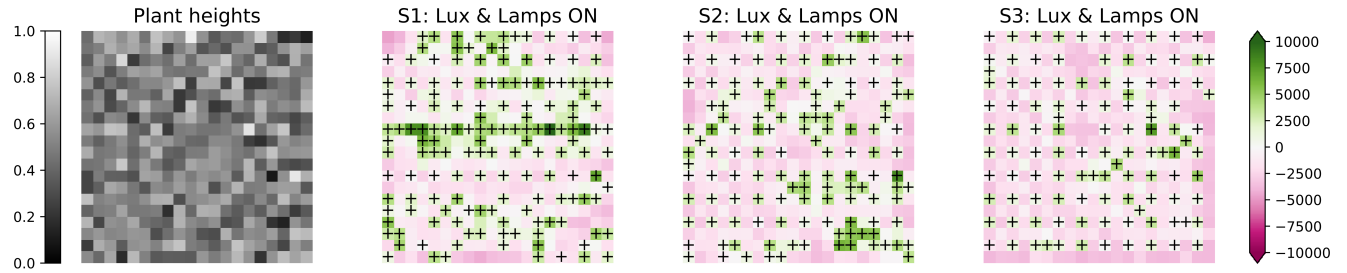

Figure 2: An example situation on the left and its adaptations on the right. Turned on lamps are marked with '+'. The colors indicate difference to the target light level per patch (in lux).

and use their deployed adaptations from its memory as a starting point for, e.g., novel configuration search (see Figure 1).

Context-awareness ensures that the system has the ability to understand the novelty of the system's current context to support intentional adaptations. For this purpose, context-awareness uses the input data provided by the base system, which then influences evaluation function. It can also motivate adaptations by enabling the base system to switch among external services with different behavioural or quality profiles. Context-awareness can also directly influence novel adaptation search, supporting context relevance and reducing search complexity.

Resource-awareness maintains the base system's current input and output mechanisms (e.g., sensors, actuators) as well as the data and control connection among the resources. Additionally, for making an adaptation intentional, a resource-centric sub-evaluation can be used to further filter out the adapted systems with excessive resource demands.

Example: A Light Control System

We use a community greenhouse light control system as an example Base System to demonstrate the usefulness of creative self-adaptation, focusing on how deliberate reasoning of the novelty (of the context, etc.) enhances the self-adaptive system's operation.

A community greenhouse is a shared space where people may rent patches to grow their own plants. Each patch is square shaped with its own lamp. For the sake of the example, we assume that plants in each individual patch are approximately of the same height, but the height varies from patch to patch. The goal of the system is to produce certain amount of light (lux) to the top of the plants in each patch. The system can adapt to different situations (ambient light level and plant height per patch) by turning lamps on and off.

The system has a set of base adaptations which are given by the system's designer, and it searches for *novel configurations* (lamp settings) in *novel situations* using the $(\mu + \lambda)$ evolutionary algorithm (Back et al., 1997). When the system deploys a novel configuration, it stores it to its memory with information about its deployment situation. Awarenesses are used to gauge which of the already deployed adaptations are used to induce the search in a new situation and how the adaptations are evaluated.

We consider the following systems, with their example adaptations shown in Figure 2:

S1: A goal-aware system perceives the ambient light level across the greenhouse and knows the target light level. The system assumes that the plant tops are always on the same level and that the height does not alternate from patch to patch.

S2: A goal- and context-aware system knows the target light level and uses sensors to measure the height of the plants in each patch.

S3: A goal- and context- and resource-aware system, in addition to S2, aims to minimise the number of lamps that are on.

We run each system 10 times through 2000 situations with varying plant heights and ambient light levels, reporting the averages for the first 1000 (fh) and the last 1000 (lh) situations in Table 1. The implemented awarenesses benefit the system by keeping it closer to its target light level (Lux Avg, marked as difference from the target) with decreased variance in the light level (Lux Std)).

In addition, S2 and S3 are able to efficiently utilise the previous contexts. Lux Avg of S2 approaches the target light level when comparing the last half to the first half, and both S2 and S3 are able to decrease Lux Std. Thus, the later adaptations of S2 and S3 are more apt because of the learned models, whereas S1 struggles to learn anything valuable from the ambient light level alone. Further, S3 is able to keep closer to the system's goal (Lux Avg, Lux Std) than S2 or S1, and does it by using less electricity (Lamps), saving both resources and money. Notably, the time used to adapt (Time, in seconds) does not increase with the added complexity of the implemented awarenesses (S2, S3). The detailed novelty assessment enables the system to filter out previous adaptations based on their perceived distance to the current situation, decreasing time used to process adaptation's in memory.

Proceedings of the 11th International Conference on Computational Creativity (ICCC'20)
ISBN: 978-989-54160-2-8

Table 1: Each system's statistics.

System	Lux Avg	Lux Std	Lamps	Time
S1 (fh)	+494	2701	191	0.486
S1 (lh)	+499	2704	192	0.506
S2 (fh)	+405	2708	192	0.468
S2 (lh)	+309	2610	192	0.475
S3 (fh)	+29	2616	181	0.491
S3 (lh)	-27	2530	182	0.492

Conclusions

We have argued that creativity is inherent in self-adaptive systems. Using an example system, we have shown that even simple mechanisms for novelty assessment can help, e.g., to improve value and decrease resource usage. However, especially for transformational novelty, we need to change our approach to software development, beginning from specifying the system requirements to allow more pronounced creative behaviour. This, in turn, places substantial demands on the system verification and validation at run time.

Acknowledgments

This work has been funded by Academy of Finland grants 313973 (CACS) and 328729 (CACDAR). We thank Hadaytullah Kundi, Ph.D. for his contributions towards this work.

References

Anant Agarwal, Jason Miller, Jonathan Eastep, David Wentziaff, and Harshad Kasture. 2009. *Self-aware computing*. Technical Report AFRL-RI-RS-TR-2009-161. MIT.

Thomas Back, David B. Fogel, and Zbigniew Michalewicz (Eds.). 1997. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, UK.

Margaret Boden. 1992. *The Creative Mind*. Abacus, London.

Simon Colton and Geraint A. Wiggins. 2012. Computational Creativity: The Final Frontier?. In *Proceedings of the 20th European Conference on Artificial Intelligence*. IOS Press, Amsterdam, The Netherlands, 21–26.

David Garlan, Shang-Wen Cheng, An-Cheng Huang, Bradley Schmerl, and Peter Steenkiste. 2004. Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure. *Computer* 37, 10 (Oct. 2004), 46–54.

Hadaytullah, S. Vathsavayi, O. Räihä, K. Koskimies, and A. Gregersen. 2012. Applying genetic self-architecting for distributed systems. In *2012 Fourth World Congress on Nature and Biologically Inspired Computing (NaBIC)*. IEEE, Mexico City, Mexico, 44–52.

Kyle E. Jennings. 2010. Developing Creativity: Artificial Barriers in Artificial Intelligence. *Minds and Machines* 20, 4 (2010), 489–501.

Anna Jordanous. 2012. A Standardised Procedure for Evaluating Creative Systems: Computational Creativity Evaluation Based on What it is to be Creative. *Cognitive Computation* 4, 3 (01 Sep 2012), 246–279.

Anna Jordanous. 2016. Four PPPPerspectives on computational creativity in theory and in practice. *Connection Science* 28, 2 (2016), 194–216.

Jeffrey O. Kephart and David M. Chess. 2003. The Vision of Autonomic Computing. *Computer* 36, 1 (Jan. 2003), 41–50.

Samuel Kounev, Jeffrey O. Kephart, Aleksandar Milenkoski, and Xiaoyun Zhu. 2017. *Self-Aware Computing Systems*. Springer.

Simo Linkola, Anna Kantosalo, Tomi Männistö, and Hannu Toivonen. 2017. Aspects of Self-awareness: An Anatomy of Metacreative Systems. In *Proceedings of the Eight International Conference on Computational Creativity*. Georgia Institute of Technology, Atlanta, Georgia, USA, 189–196.

Gabriel A. Moreno, Javier Cámara, David Garlan, and Bradley Schmerl. 2015. Proactive Self-adaptation Under Uncertainty: A Probabilistic Model Checking Approach. In *Proceedings of the 10th Joint Meeting on Foundations of Software Engineering*. ACM, New York, NY, USA, 1–12.

Henry Muccini, Mohammad Sharaf, and Danny Weyns. 2016. Self-adaptation for Cyber-physical Systems: A Systematic Literature Review. In *Proceedings of the 11th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*. ACM, New York, NY, USA, 75–81.

Mark A. Runco and Garrett J. Jaeger. 2012. The Standard Definition of Creativity. *Creativity Research Journal* 24, 1 (2012), 92–96.

Mazeiar Salehie and Ladan Tahvildari. 2009. Self-adaptive Software: Landscape and Research Challenges. *ACM Transactions on Autonomous and Adaptive Systems* 4, 2, Article 14 (May 2009), 42 pages.

Dan Ventura. 2017. How to Build a CC system. In *Proceedings of the Eigth International Conference on Computational Creativity*. Georgia Institute of Technology, Atlanta, Georgia, USA, 253–260.

Alan Frank T. Winfield. 2014. Robots with internal models: A route to self-aware and hence safer robots. In *The Computer After Me*. Imperial College Press, 237—-252.