Interwoven Systems: Self-improving Systems Integration

Kirstie Bellman
Topcy House Consulting
bellmanhome@yahoo.com

Sven Tomforde
University of Augsburg
sven.tomforde@informatik.uni-augsburg.de

Rolf P. Würtz
Ruhr Universität Bochum
rolf.wuertz@ini.ruhr-uni-bochum.de

***Abstract*—Current trends in information and communication technology show that systems are increasingly influencing each other – which is seldom completely anticipated at design-time. As a result, mastering system integration with traditional methods becomes infeasible due to the resulting complexity. In this paper we argue that self-improving system integration is the most promising solution to counter the resulting challenges. Thereby, we highlight the different aspects of such a process with special attention to the optimisation question and discuss how approaches from the domain of self-organising systems – in particular Organic and Autonomic Computing – will be beneficial when researching possible solutions.**

I. Motivation

Information and communication technology (ICT) pervades every aspect of our daily lives. This inclusion changes our communities and all of our human interactions. It also presents a significant set of challenges in correctly designing and integrating our resulting technical systems. For instance, the embedding of ICT functionality in more and more devices (such as household appliances or a power grid's thermostats) leads to novel interconnections and a changing structure of the overall system. Not only technical systems are increasingly coupled, a variety of previously isolated natural and human systems have consolidated into a kind of overall system of systems — an interwoven system structure.

In this context, the term "interwoven" [12] refers to the several aspects of coupling and mutual influences between component systems: heterogeneous elements interact, indirect influences of behaviour can be observed, and the context, cooperation and composition of systems are uncertain during runtime. For instance, legacy systems have to cooperate with novel solutions or different versions of the same system have to coexist and cooperate. This leads to novel challenges for system engineers and administrators.

The ongoing integration of interwoven systems depends on the valid and timely knowledge of each of the participating systems on dynamically changing goals or priorities, on the state of its available resources and on the details of its operational context. It is infeasible to have such timely and intimate dynamically-changing knowledge of the participating systems come from an external master controller for the overall interwoven system. Hence, the most promising approach to master such systems relies on increasing their self-organisation capabilities. Within the last decade, initiatives like Autonomic Computing (AC) [6] and Organic Computing(OC) [11] have started to investigate and build systems based on the fundamentals of self-organisation: Concepts from natural organisms have been transferred to technical systems in order to achieve "life-like" characteristics such as adaptivity, robustness, and flexibility. Architectural patterns, customised machine learning techniques and collaboration mechanisms have been developed. However, these approaches and mechanisms have been developed with the focus on individual, isolated systems; that is to address such questions as how can one system be equipped with the desired properties and which techniques are necessary?

Although research in self-organising systems – such as the *Organic Computing* (OC) and *Autonomous Computing* (AC) initiatives – has seen an exciting decade of development with considerable success in building individual systems, OC/AC is faced with the difficult challenge of integrating multiple self-organising systems, and integrating self-organising systems with traditionally engineered ones as well as naturally occurring human organisations. In addition, although there has been important development in system of systems methodologies (e.g., service-oriented architectures, clouds technology etc.), many of these developments lack scalable methods for rapidly proving that new configurations of components/subsystems are correctly used or their changes verified or that these frameworks have pulled together the best possible context-sensitive configuration of resources for a user or another system.

The change of structure from individual to interwoven system is fundamental and affects the entire production cycle of technical systems. The dynamic and relatively open structure of interwoven systems means that one cannot 'freeze' and certify the system requirements, functionality and configuration one time; this basically makes standard system integration and testing infeasible. The increasingly complex challenges of developing the right type of modelling, analysis, and infrastructure for designing and maintaining ICT infrastructures has continued to motivate the autonomic and self-organising community. In this paper, we intend to lay out some of the challenges and the needs for novel approaches to system of system integration and testing by applying OC principles; specifically we want approaches that allow for a continual process of self-integration among components and systems that is self-improving and evolving over time towards an "optimised" and stable solution. The term "optimised" is in

quotes because in fact, one of the clear challenges in an interwoven system is to go beyond our current concepts of optimisation, and even "satisficing" of multiple objectives and to develop methods for ensuring overall improvement in not only responses and behaviour, but perhaps in preparation and readiness for accomplishing diverse goals among a continually changing set of participating systems.

The purpose of this paper is to discuss ideas and approaches beyond the perspective of OC. Instead of building isolated systems with "life-like" and self-x properties, in interwoven systems we must counter the (possibly) negative effects of aggregating classical and OC systems, which includes exchanging, updating or removing components, or abstracting influences on other systems in the environment during the design processes of the interwoven system or its participating systems. Besides individual decisions about current behaviour (to achieve adaptivity and robustness), the design itself and the integration processes have to become modifiable and revisable at runtime. Also the challenges of self-adaptive systems remain relevant here [5], [8].

To start the discussion of how to apply OC approaches to the level of interwoven systems, we start with one of the defining characteristics of OC individual systems – their ability to self-modify and self-optimise. At the level of the individual system, self-optimisation methods (whether by the interaction among 'agents' or various algorithmic approaches) allow the system to self-improve with regards to a set of goals. These goals or objectives can be given to a specific subsystem or self-generated based on guiding principles designed into the overall system, and there are feedback mechanisms from the system and from the environment to determine whether the goals have been properly formulated and met. These concrete sets of self-optimising and feedback processes make an excellent starting point for laying out the challenges of applying OC to interwoven systems, as one must now develop frameworks and methods for integrating among what could be overly local optimising processes. As we argue below it also brings to the fore new strategies for deciding how to schedule attention and optimisation among a number of self organising systems goals, as well as 'fairness' and trust among the cooperating interwoven systems.

Subsystem integration is one basic requirement for Organic Computing systems [10]. For interwoven systems, the communication and cross-adaptation must become part of the control process. This requires metrics on how well the communication with other subsystems is going. For example, if message queues are increasing in length without the possibility of fulfilling the respective requests, this should lead to some behaviour change, which needs to be propagated through the system for the search of alternative operation flows. Certainly, this requires *redundancy*. Systems that are at full load cannot be interwoven, because they are only concerned with themselves. The "insights," as to what some subsystems can do to to avoid damage to others must become part of their design and subject to internal reasoning. This is different from a purely 'selfish gene' evolutionary approach, where taking over as many resources as possible could be construed as the most desirable goal for agents or subsystems. Rather, as discussed more below, this starts to bring out the motivations for cooperation and trust and perhaps even altruism seen in biological systems. A well-behaved system probably requires enforcement of diversity constraints (like antitrust laws) or evolutionary dynamics for which monocultures are not stable attractors. Thus, in technical systems of our design Adam Smith's "invisible hands" that would guarantee fair global behaviour cannot be taken for granted, but need to be demonstrable if not provable system properties.

II. Description of the challenges

Taken the dramatic changes in ICT system management and utilisation seriously demands for a fundamental change of the engineering and administration processes. Design in the classical sense is a hierarchical top-down process starting with a pure functional specification and ending with a detailed fabrication "recipe". After going into production, systems are only slightly adapted and updated. Increasing the degree of self-organisation means not only to allow for self-x properties like self-configuration, self-optimisation or self-healing. Instead, it demands concepts for changing and managing the structure and composition of large-scale integrated systems at runtime and without the help of a user. Systems need abilities to manage their place within a large-scale coupled system, to modify their design in certain stages, and to self-reflect or reflect about connected systems and their behaviour. This paper discusses challenges related to these issues.

Optimisation in complex systems, especially ones composed of System of Systems [9] or interwoven systems do not have one global function and hence one multi-criteria objective function. Nevertheless, global constraints on behaviour must be enforceable. Some of the key characteristics of optimisation in complex systems are:

1) Optimisation is **context-dependent**. What is a good solution or a good step towards a solution depends on the context. It can include dropping goals, giving up on an objective, or not persisting at some endeavour. Hence, persistence to a fixed criterion is not correct.
2) As noted above, the optimisation criteria are not fixed; the systems or components active at any given point also may vary. Hence, optimisation is continual and **continually changing**.
3) There is **no unique solution**; there are in fact many 'paths to Rome' based on what combination of objectives are targeted and how resources are adjusted.
4) The system(s) must be continually **"viable"** (in a good enough state to survive, satisficing, in a good 'safehold' if an emergency state etc.). One cannot take interwoven systems totally offline. This also relates to the problem that taking no decision at all has severe consequences.
5) **Goals can be altered** at runtime – making the underlying fitness landscape continuously changing. Such a change can be triggered by users but also by the system itself. Consider for instance traffic control: the normal

goal is to decrease averaged waiting times. In case of oversaturated situations, the system might switch to the goal of relieving as much traffic as fast as possible. These are contradictory strategies. We need concepts to allow for such a change and for techniques that can adapt quickly and reliably.

6) Optimisation at this **system level** can include trade-offs among goals, and hence includes issues close to scheduling problems and social issues such as 'trust' (systems are willing to be deferred or operate at sub-optimal levels as long as they 'get their turn'). At this level, there can be new strategies and new policies (this aspect is discussed in more depth in the remainder of this paper).

As one can see, **evaluation**, understanding, and defining what is good or satisficing or optimal in interwoven systems is complex. Instead of finding the optimal solution, we need good enough and fast enough results. But: How do we know if the current solution is good enough? And that a given result is soon enough to be effective?

Due to complexity and scalability reasons, bio-inspired **strategies** as proposed by OC are promising. For example, animal systems are multi-goaled [1]. That is, the animal's available sensors and musculature must be configured and recombined to form a wide spectrum of behaviours – to accomplish many goals. The emphasis will be on having the mechanisms that allow rapid switching to a new behaviour, on modifying that movement to fit the incoming external demands, to recombine rapidly the use of the identical movement from one action sequence and purpose to another, to blend or merge movement patterns, rather than the consistent replication of one movement or one movement 'sequence'. Through evolution, there have been numerous mechanisms developed within the animal kingdom that allow very different strategies for handling simultaneous goals, including "merging", which gracefully combines the behavioural characteristics of several goals at the same time. These sophisticated adaptive decision processes are seen even in 'lower' animals (e.g., crayfish and lizards [1], [2].)

As another example of a common strategy with which biological systems handle competing demands is that used to handle critical timing demands: generating 'holding actions' or 'buying time' while at the same time investing the time and effort to plan a more adequate, specific response. Many reflexes operate in exactly this fashion; they are rigid units of behaviour that are triggered and released ballistically by events that by (evolutionary) experience need immediate and very fast responses. They often buy the system time to plan and put into motion more sophisticated behaviours. The 10 msec tail flip in crayfish, which is among the fastest known animal reflexes, is soon overtaken by a slower response (50 msec) for directed swimming that is activated in parallel [2], [7] As shown by this example, a system may execute a rapid fixed response, foregoing assessment of the environment, reflection, planning, reasoning, and coordination in order to meet urgent time constraints. However, the system is also preparing a more reasoned and deliberate response that takes into account its previous exploration and knowledge of its operational environment. See [4] for other biological examples of use to OC systems, especially in the use of Central Pattern Generators that allow very quick changes between different response patterns by modulating at low-cost parameters in the operational system and that help the operational system quickly find a 'good enough' solution given conflicting trade-off factors.

The biological use of "common components" within several behavioural responses or patterns used to buy time and other time-buying strategies that bring the system into a better position for the next set of goals help us recognise that 'good enough' must not only be thought of in terms of competing goals, but over several time scales. Other concepts are closely related: emergency behaviours or modes (allowing only critical actions to dominate and actually shutting systems down); sacrificing current optimisation in order to position for a better outcome later (e.g., example of football player positioning self); permitting "inefficiencies" for a later good (e.g., in animals, exploratory behaviour, playing, learning, sleep.) – just to name the most important ones.

However, as interesting and potentially important as these biological strategies are for the management of competing goals, it is not clear how to incorporate them into strategies for the case of interwoven systems. One of the challenges we face in interwoven systems is that there is no one system – it is less like a body and more like a community. This means that our strategies for merging the goals and behaviours (or any of the other strategies noted above) of different participating systems may in fact look quite different from the ones used inside each participating system. Obviously, optimisation at a system of system level requires new strategies for integrating different levels of optimisation processes as well as local and global optimisations. In this context, we argue that the ability of systems to have self-reflection processes — that is to explicitly reason about their own goals, resources, performance, and environment — is critical to the continual self-improvement and self-integration approaches for interwoven systems. Optimisation within complex systems need different kinds of model. Hence, we probably need self-reflection techniques to derive runtime models of the system and their capabilities, about the neighbouring systems, and about the environment.

Recently [3] proposed a strategy for using reflective architectures to help organise and integrate the optimisation methods applied throughout a complex system at different levels. Additionally, self-modelling has been emphasised since the beginning, as a critical aspect of OC systems. However, there are many challenges in developing the appropriate self-monitoring, self-reflection, and self-modelling capabilities for interwoven systems. Since the interwoven system has no central controller or final authority, the system as we have noted is more like a community with all the needs for negotiation, trust, and social policies that the term "community" implies. Hence, we not only have all the difficult research questions on how to use self-models and reflection within single systems

so as to improve a given control mechanism (i.e., being responsible for the self-adaptation and self-managing parts) but also now the challenges of understanding what role those processes and their results play in the overall coordinated control of the interwoven systems. This brings up issues such as distinguishing between inner and outer models: What do I want to let neighbours know about me? As much as needed? As much as possible? How can we distinguish these models automatically and according to different situations? Further, how do we quantify certainty within and about such models (both one's own and those provided by neighbours.)

Optimisation among interwoven systems will include many 'social' level issues including trust, such that one system will allow another system to optimise for now or fulfil its goals first as long as it has the assurance that it will be treated similarly later. Perhaps some new versions of queuing theory would be of interest here. Certainly this trust is not for a single transaction, but rather for a longer term set of interactions and will include **policies** as a major subject. For instance, high-level policies could include such rules as: "selfish gene policy" (never give up any of your system's individual objectives, no compromises), "never starve anyone" (never allow any of the individual systems to go below a threshold in actions accomplished – related to continual viability); allow all systems to be the "winner takes all", one at a time; have a policy in regards to percentage of non-optimal actions (e.g., not all actions done well) to maybe even a percentage of non-sufficient (e.g., not all actions done or ones allowed to fail). Several further approaches are possible – making runtime optimisation a complex problem.

Such an optimisation process is – by design – not a fully mathematical complex any more. Instead, we will rely on heuristic approaches such as evolutionary or swarm-based algorithms. These are mostly based on populations of individuals exploring and exploiting the search space. Such concepts could be quite useful to the evolution in the interwoven systems.

Lastly, interwoven systems are collections of entities that are brought together either permanently or temporarily and do not belong to one authority. Yet they have to interact with each other, either distantly (e.g., have some knowledge of activities impacting one's own actions or decisions) or intimately (e.g., exchange vital information or even code for needed agents or capabilities). One major question here is how can we automatically establish technical trust among such entities, how can this be adapted dynamically and how is it affected by disturbances such as malicious behaviour?

III. Interwoven systems greatly change our design processes

Interwoven systems have many implications for traditional design processes, and now even more if we seriously apply OC design and operational principles to them. Some of the questions are: What are the implications of the OC design notion on today's perception of design flows and tools? What fraction of the bottom-up design can/has to be addressed during pre-fabrication design time? What amount of upfront overhead in form of redundant system resources should an OC system provision in order to allow self-organised functional or performance enhancement at run-time? Alternatively, assuming fixed resources, can OC systems degrade lower priority functions for mission critical ones? How do we deal with recording and evaluating the dynamic priorities that are negotiated among the distributed entities?

We also need incremental design processes, where we can add further OC capabilities at runtime. We have to make the optimisation aspect a major capability of systems and a major subject of design. Again there is a Yo-Yo-kind of design approach that must allow the system to add/remove/adapt or replace optimisation capabilities, to have as much freedom as possible to self-determine its approach to fulfilling its requirements within its operational environments, and for gaining the knowledge it needs about the environment and the other participating systems in order to take into consideration when optimising itself.

Coming up with design principles that support successful interwoven systems may require boundary conditions on the very objective functions that guide the behaviour. These could have the form that "red" parameter ranges at other subsystems must lower the system's own success. Objectives that overemphasise minimising resource use or hardware cost usually have fixed points just on the edge of breakdown. Which is what the optimised systems do under infinitesimal unexpected problems. These issues not only challenge design decisions, but change how specifications need to occur for interwoven system functions, as well as their verification and validation.

IV. Bio-inspired (OC) optimisation in complex systems

As already discussed in the previous section, we search for solutions in the domain of bio-inspired techniques. This section summaries some of the impacts of bio-inspired – or better: OC-based – optimisation in interwoven systems.

1) Properties that an OC/AC-based system integration could enable:
   a) The characteristics of self-organising systems (i.e., self-improving, self-monitoring and self-reflective) to be applied to system integration.
   b) Methods for self-integration – making the integration continual and evolving, self-improving over time, and guided by models (coupled to self-monitoring, reasoning and analysis).
   c) Collaboration schemes that can provide these models or enough timely information to other systems in order to better interface and coordinate efforts.
2) Approaches for self-modelling at runtime.
3) Quantification of the system integration's quality and performance.
4) System integration with uncertainty.
5) Implications on traditional design processes:
   a) Implications of the OC design notion on today's perception of design flows and tools.

b) Trade-off between bottom-up design and prefabrication.
c) Overhead in form of redundant system resources to be provisioned to allow self-organised functional or performance enhancement at runtime.

We argue that OC has some of the characteristics to approach these problems above:

1) OC emphasises using bio-inspired strategies.
2) OC sets us up for investigating the optimisations/solutions available to interactions among systems/agents rather than classical multi-criteria optimisation.
3) Because of overlap between optimisation problems and biological approaches to scheduling and resolving multiple goals noted above there are a number of OC strategies for scheduling activities that might be of interest and use here.
4) OC provides experience in the use of a diverse set of distributed learning algorithms and approaches that can act as basis for system's continually learning about their environment and the other participating systems
5) OC changes the general idea from "do optimisation" to an adapting system that "continually optimises" so that any change in context, goals, integration among components or agents automatically causes adjustments. Thereby, the resulting fitness landscape and the current context play a major role.
6) Of use here will be self-modelling approaches, such as the reflective architecture and "continual contemplation" [3] as a strategy for pulling together individual local, self-optimisations.

Recently, improvements in the fields of modelling (self-modelling, modelling at runtime), technical trust (reputation, trustworthiness), system design, or the analysis of socio-technical impacts of system behaviour paved the way for an orchestrated approach to deal with these challenges. Based on these concepts, a solution to the motivating problem lies in the ability of self-design: We have to accept the fact that our technical systems are increasingly interwoven and therefore find ways to allow systems to integrate into the overall context.

This includes modelling themselves within the dynamically changing uncertain environment, to be capable of reflection, and to give other related systems the possibility to reason about the possible effects of interaction based on (parts of) such a model. By tuning their models at runtime and taking possible effects into account, the systems themselves can rearrange their overall structure and thereby self-improve the performance.

This is a very brief overview of the challenges facing the OC/AS community as they tackle the integration of system of systems using new self-optimising and self-improving methods for a style of continual self-improving systems integration.

REFERENCES

[1] K.L. Bellman. The Conflict Behavior of the Lizard, Sceloporus Occidentalis: And Its Implication for the Organization of Motor Behavior. Ph.D. Dissertation. University of California, San Diego, 1979, 450 pps.

[2] K.L. Bellman and F.B. Krasne. Adaptive complexity of interactions between feeding and escape in crayfish. Science, vol. 221, no. 4612, pp. 779–781, 1983.

[3] K.L. Bellman and C.A. Landauer. Reflection processes help integrate simultaneous self-optimization processes. In: Proceedings of SAOS (ARCS 14), Lübeck, Germany, 2014.

[4] K.L. Bellman and P.R. Nelson. Developing Mechanisms for Determining 'Good Enough' in SORT Systems. In: Proc. Workshop on Self-Organizing Real Time Systems, Newport Beach, California, March 28-30, 2011.

[5] B.H.C. Cheng et al. Software Engineering for Self-Adaptive Systems: A Research Roadmap. In: Self-Adaptive Systems, Springer LNCS 7475, pages 1–32, 2013.

[6] J.O. Kephart and D.M. Chess. The Vision of Autonomic Computing. In: IEEE Computer, vol 36(1), IEEE, pages 41–50, 2003.

[7] A.P. Kramer et al.. Different command neurons select different outputs from a shared premotor interneuron of crayfish tail circuitry. In: Science, vol. 214, pp. 810812, 1981.

[8] R. de Lemos et al. Software Engineering for Self-Adaptive Systems: A Second Research Roadmap. Proc. of Software Engineering for Self-Adaptive Systems, Springer LNCS 5525, pages 1–26, 2009.

[9] M.W. Maier. Architecting principles for systems-of-systems. In: Systems Engineering, 1(4), pages 267–284, 1998.

[10] C. von der Malsburg. The organic future of information technology. In R.P. Würtz (ed.), *Organic Computing*, chapter 2, pages 7–24. Springer, 2008.

[11] C. Müller-Schloer. Organic Computing: On the Feasibility of Controlled Emergence. In: Proc. CODES+ISSS'04, ACM, pages 2–5, New York 2004.

[12] S. Tomforde, J. Haehner, H. Seebach, W. Reif, B. Sick, A. Wacker, and I. Scholtes. Engineering and Mastering Interwoven Systems. In: ARCS 2014 Workshop Proceedings, VDE Verlag, pages 1–8. 2014.