Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 00 (2014) 000–000

Procedia
Computer Science

1st International Workshop "From Dependable to Resilient, from Resilient to Antifragile Ambients and Systems" (ANTIFRAGILE 2014)

Antifragility = Elasticity + Resilience + Machine Learning

Models and Algorithms for Open System Fidelity

Vincenzo De Florio*

PATS research group, University of Antwerp & iMinds Research Institute, Middelheimlaan 1, 2020 Antwerpen, Belgium

Abstract

We introduce a model of the fidelity of open systems—fidelity being interpreted here as the compliance between corresponding figures of interest in two separate but communicating domains. A special case of fidelity is given by real-timeliness and synchrony, in which the figure of interest is the physical and the system's notion of time. Our model covers two orthogonal aspects of fidelity, the first one focusing on a system's steady state and the second one capturing that system's dynamic and behavioural characteristics. We discuss how the two aspects correspond respectively to elasticity and resilience and we highlight each aspect's qualities and limitations. Finally we sketch the elements of a new model coupling both of the first model's aspects and complementing them with machine learning. Finally, a conjecture is put forward that the new model may represent a first step towards compositional criteria for antifragile systems.

Keywords: Resilience, computational antifragility, antifragile engineering, elasticity

1. Introduction

As well-known, open systems are those that continuously communicate and "interact with other systems outside of themselves"[1]. Modern electronic devices[2] and cyber-physical systems[3] are typical examples of open systems that more and more are being deployed in different shapes and "things" around us. Advanced communication capabilities pave the way towards collective organisation of open systems able to enact complex collective strategies[4] and self-organise into societies[5], communities[6,7], networks[8], and organisations[9].

One of the most salient aspects of open systems—as well as a key factor in the emergence of their quality—is given by the compliance between physical figures of interest and their internal representations. We call this property as fidelity. A high fidelity makes it possible to build "internal" models of "external" conditions, which in turn can be used to improve important design goals—including performance and resilience. Conversely, low fidelity results in unsatisfactory models of the "world" and the "self"—an argument already put forward in Plato's Cave.

As an example, real-time systems are open systems that mainly focus on a single figure—physical time. Such figure is "reified" as cybertime—an internal representation of physical time. Intuitively, the more accurately the

* Corresponding author. Tel.: +32-3-2653905; fax: +32-3-2653777.
URL: https://www.uantwerp.be/en/staff/vincenzo-deflorio (Vincenzo De Florio)

internal representation reflects the property of a corresponding physical dimension, the higher will be the quality exhibited by such class of open systems.

In what follows we consider the more general case of n-open systems, namely systems that interact with environments represented through n context figures. This means that, through some sensory system and some sampling and conversion algorithms, each of these n context figures is reified in the form of an internal variable reflecting the state of the corresponding figure. These "reflective variables"[10] are the computational equivalent of the biological concept of *qualia* (pl. *quale*)[11] and represent an open system's primary interface to their domains of intervention (typically, the physical world.)

This work introduces two models for the fidelity of n-open systems. Each of those models provides a different view to an n-open system's nature and characteristics.

The first model is presented in Sect. 2 and mainly focuses on *elasticity* support to fidelity. Quality is reached through simple schemes with as limited as possible an overhead and as low as possible an impact on functional design goals. Resource scheduling, redundancy, and diversity are mostly applied through worst-case analyses and at design-time, possibly with simple switching among Pareto-optimal strategies during the run-time[2]. As mentioned above, the key strategy in this case is elasticity: unfavourable changes and faults are meant to be masked out and counterbalanced by provisions that do not require intensive system reconfigurations. This model considers the system and its intended deployment environments as known and stable entities (cf. synchronous system model[12]) and identifies a snapshot of the system in its intended (viz., "normal") operational conditions.

Conversely, our second model—introduced in Sect. 3—is behavioural and focuses on *resilience* support to fidelity. Systems and their environments are regarded as *dynamic systems* whose features are naturally drifting in time (as it is the case, e.g., in the timed-asynchronous system model[13]). Corresponding variations in the operational conditions within and without the system boundaries may be tolerated through different strategies, and this model focuses on the quality of the behaviours that a system may employ, during the run-time, in order to guarantee its fidelity despite those variations.

A discussion is then elaborated in Sect. 4. Positive and negative aspects of both models are highlighted. Then it is shown how the two models may co-exist by distinguishing between normal and critical conditions. A general scheme for context-conscious switching between elasticity and resilience strategies is proposed. Said scheme also incorporates a machine learning step such that the system may acquire some form of "wisdom" as a by-product of past history. A conjecture is put forward that the general scheme may represent a first preliminary step towards the engineering of *antifragile* systems[14], namely systems not merely able to tolerate adverse conditions, but rather able to strengthen in the process their ability to do so.

Section 5 finally concludes with a view to our future work.

2. Algebraic Model

Here we introduce our first model of fidelity. First the main objects of our treatise are presented in Sect. 2.1. Section 2.2 follows and introduces our Algebraic model based on those objects.

2.1. Formal entities

As mentioned in Sect. 1, open systems are those computer systems that interact with a domain they are immersed in. A prerequisite to the quality of this interaction is the perception[15] of a domain- and application-specific number of figures, say $0 < n \in \mathbb{N}$. The quality of the perception services is the cornerstone to fidelity[16], the latter taking the shape of, e.g., optimal performance, strong guarantees of real-timeliness and safety, high quality of experience, etc.

Perception in n-open systems is modelled in what follows as a set of functions $q_i, 0 < i \leq n$, defined between pairs of Algebraic structures, $\mathcal{U}_i$ and $C_i, 0 < i \leq n$, respectively representing sets of physical properties of interest (called in what follows "raw facts") and sets of their corresponding computer-based operational representations—their "reflective variables", or "quale". "Reflective maps" is the term we shall use to refer to the above functions.

Let us refer as $\mathcal{U}$ and C respectively as to any of the $\mathcal{U}_i$ and C_i, and as q to any of the reflective maps. Thus, if $u \in \mathcal{U}$ is, e.g., the amount of light emitted by a light-bulb, then $q(u)$ may for instance be a floating point number stored in some memory cells and quantifying the light currently emitted by the bulb as perceived by some sensor and as represented by some sampling and conversion algorithm. A reflective map takes the following general form:

$$q : \mathcal{U} \to C \tag{1}$$

and obeys the following Condition: $\forall u_1, u_2 \in \mathcal{U} : q(u_1 + u_2) = q(u_1) + q(u_2) + \Delta.$ (2)

Here overloaded operator "+" represents two different operations:

- In $\mathcal{U}$, as in the expression on the left of the equal sign, operator "+" is the property resulting from the composition of two congruent physical properties. As an example, this may be the amount of light produced by turning on two light-bulbs in a room, say light-bulb l_1 and light-bulb l_2. (Note that the amount of light actually perceived by some entity in that room will depend on the relative positions of the light-bulbs and the perceiver as well as on the presence of obstructing objects in the room, and other factors.)
- In C, as in the expression on the right of the equal sign, operator "+" is the algorithm that produces a valid operational representation of some property by adding any two other valid representations of the same property. In the above example, the operator computes the qualia corresponding to the sum of the quale representing the light emitted by l_1 with that of l_2.

Let Δ be a variable representing any $\Delta_i, 0 < i \leq n$, in turn defined as the "preservation distance" of reflective map q_i, meaning that q_i would preserve operation "+" were it not for the extra Δ_i. Quantity Δ thus represents an error value that depends on the nature of the involved properties; their operational representations; and "the environment", the latter being modelled as a set of context figures representing the hardware and software platforms; the operational conditions; the user behaviours; and other factors. Environmental conditions shall be cumulatively represented in what follows as vector $\vec{e}$.

2.2. Model

Function (1) and Condition (2.1) may be used to characterise concisely the fidelity of an n-open system, namely how coherent, consistent, and robust is the reflection in C_i of the physical properties of $\mathcal{U}_i, 0 < i \leq n$. Our exemplary focus in the rest of this section will be that of real-time systems (namely 1-open systems whose domains of interests are cybertime[1] and physical time) though this will not affect the generality of our treatise. In the rest of this section C will refer to cybertime and $\mathcal{U}$ to physical time. The corresponding reflective map shall be simply referred to as q while Δ shall be q's preservation distance.

The formal cornerstone of our model is given by the concept of isomorphism—a bijective map between two Algebraic structures characterised by a property of operation preservation. As well-known, a function such as reflective map q is an isomorphism if it is bijective and if the preservation distance Δ is equal to zero. In this case the two domains, C and $\mathcal{U}$, are in perfect correspondence: any action (or composition thereof) occurring in either of the two structures can be associated with an equivalent action (resp. composition) in the other one. In the domain of time, this translates in perfect equivalence between the physical and artificial concept of time—between cybertime and physical time that is. Different interpretations depend on the domain of reference. As an example, in the domain of safety, the above correspondence may mean that the consequence of C-actions in terms of events taking place in $\mathcal{U}$ be always measurable and controllable—and vice-versa.

Obviously the above flawless correspondence only characterises a hypothetically perfect computer system able to sustain its operation in perfect synchrony with the physical entities it interacts with—whatever the environmental conditions may be. The practical purpose of considering such a system is that, like Boulding's transcendental systems[17] or Leibniz's Monads and their perfect power-of-representation[18], it is a *reference point*. By identifying specific differences with respect to said reference point we can categorise and partition existing families of systems and behaviours as per the following definitions.

[1] We shall refer in what follows to any artificial concept of time, as manifested for instance by the amount of clock ticks elapsed between any two computer-related events, as to "cybertime".

Definition 1 (Hard real-time system). *A hard real-time system is the best real-life approximation of a perfect real-time system. Though different from zero, its preservation distance (the Δ function, representing in this case the system's "tardiness") has a bound range (limited by an upper threshold) equal to a "small" interval (drifts and threshold are, e.g., one order of magnitude smaller than the reference time unit for the exercised service). A hard real-time system is typically guarded, meaning that the system self-checks its preservation distance.*

We shall call "t-hard real-time system" a system that matches the conditions in Definition 1 with threshold t.

Definition 2 (Soft real-time system). *A system is said to be "soft real-time" if its preservation distance Δ is statistically bound. As in hard real-time systems, a threshold characterises the Δ error function, but that threshold is an average value, namely there is no hard guarantee, as it was the case for hard real-time systems, that the error will* never *be overcome. Both threshold and its standard deviation are "small". As hard real-time systems, also soft real-time systems are typically guarded—viz., they self-check their tardiness (preservation distance.)*

In what follows we shall call "(t, σ)-soft real-time system" a system that matches the conditions in Definition 2 with average threshold t and standard deviation σ.

Definition 3 (Best-effort real-time system). *A system is said to be "best-effort" if care and experience have been put to use, up to a certain degree[2], in order to design and craft that system; said care and experience, to the best of the current knowledge and practise, should allow the Δ values experienced by the users to be considered as "acceptable", meaning that deviations from the expected behaviours are such that the largest possible user audience shall not be discouraged from making use of the system. Internet-based teleconferencing systems are examples of systems in this category. Unlike hard and soft real-time systems, best effort systems do not monitor the drifting of their Δ[3].*

It is important to highlight once more how function Δ is also a function of $\vec{e}$—the environmental conditions. As mentioned already, the above conditions include those pertaining to the characteristics and the current state of the deployment platform. As a consequence of this dependency, special care is required to verify that the system's deployment and run-time hypotheses *will stay* valid over time. Sect. 3 specifically covers this aspect. Assumption failure tolerance[19] may be used to detect and treat deployment and run-time assumption mismatches.

Definition 4 (Non-real-time system). *A non real-time system is one that is employed, deployed, and executed, with no concern and no awareness of the drifting of function Δ. With respect to time, the system is context-agnostic and is meant to be used "as is"—without any operational or quality guarantee.*

Definitions 1–4 can be used to partition systems into disjoint blocks (or equivalence classes). Said classes may be regarded as "contracts" that the systems need to fulfil in order to comply to their (real-timeliness) specifications.

Definition 5 (System identity). *We define as system real-time identity (in general, its system identity) the equivalence class a (real-time) system belongs to.*

We now discuss a second and complementary aspect—system behaviour and its effect on the correspondence between C and $\mathcal{U}$.

3. Behavioural Model

As already hinted in Sect. 1, our Algebraic model and its Definitions 1–4 do not cover an important aspect of open systems[1], namely the fact that, in real-life, the extent and the rate of the environmental changes may (as a matter of fact, *shall*) produce a sensible effect on $\vec{e}$ (and thus, on Δ) even when the system has been designed with the utmost

[2] A trade-off between design quality, usability, time-to-market, costs and other factors typically affects and limits the employed care.

[3] In some cases monitoring data are gathered from the users. As an example, the users of the Skype teleconferencing system are typically asked to provide an assessment of the the quality of their experience after using the service. This provides the Skype administrators with statistical data regarding the Δ's experienced by their users.

care. In this case the system may experience an "identity failure", namely a drift[4] that produces a loss of the system identity (cf. Def. 5).

In order to capture a system's ability to detect, mask, tolerate, or anticipate identity failures, that system needs to enact a number of resilient behaviours[21,15]. In what follows we first briefly introduce the concepts of resilience and behaviours (respectively in Sect. 3.1 and Sect. 3.2) and then discuss in Sect. 3.3 how resilient behaviours constitute a second "parameter" with which one may characterise salient aspects of the fidelity of open systems.

3.1. Resilience

Resilience is a system's ability to retain certain characteristics of interest throughout changes affecting itself and its environments. By referring to Sect. 2.2 and in particular to Def. 5, resilience may be defined as *robust system identity persistence*, namely a system's "ability to pursue completion (that is, one's optimal behaviour) by continuously re-adjusting oneself"[15]. Resilience closely corresponds to the Aristotelian concept of entelechy[22,23], namely "exercising activity in order to guarantee one's identity", or to "comply to one's 'definition'."

As suggested in[21], resilience calls for (at least) the following three abilities:

- Perception, namely the ability to become timely aware of some portion of the raw facts in the environment (both within and without the system boundaries).
- Awareness, which "defines how the reflected [raw facts] are accrued, put in relation with past perception, and used to create dynamic models of the self and of the world"[24,15].
- Planning, namely the ability to make use of the Awareness models to compose a response to the changes being experienced.

A general scheme for robust system identity persistence is then given by the following three phases:

1. Monitor the drifting of the Δ functions.
2. Build models to understand how the drifting is impacting on one's system identity.
3. Plan and enact corrective actions such that the system identity is not jeopardised.

Phases 2. and 3. refer to the concept of behaviour—which is the subject of next subsection.

3.2. Behaviour

Behaviour is defined in[25] as "any change of an entity with respect to its surroundings[5]". In the context of this paper behaviour is to be meant as *any change an entity enacts in order to sustain its system identity*. In other words, behaviour is the response a system enacts in order to be resilient. In the cited paper the authors discuss how the above mentioned response may range from simple and predefined reflexes up to complex context-aware strategies. The following classes are identified:

1. Passive behaviour: the system is inert, namely unable to produce any "output energy".
2. Active, non-purposeful behaviour. Systems in this class, albeit "active", do not have a "specific final condition toward which they strive".
3. Purposeful, non-teleological (i.e., feedback-free) behaviour. A typical example of systems exercising this type of behaviour is given by servo-mechanisms.
4. Teleological, non-extrapolative behaviours are those typical of *reactive* systems. A feedback channel provides those systems with "signals from the goal". Behaviour is then adjusted in order to get "closer" to the goal as it was perceived through the channel. Reactive systems function under the implicit hypothesis that the adjusted behaviours bring indeed the system closer to the goals.

[4] The problem of system identity drift going undetected is one that may produce serious consequences—especially in the case of safety-critical computer systems. Quoting Bill Strauss, "A plane is designed to the right specs, but nobody goes back and checks if it is still robust"[20].

[5] Here and in what follows, when not explicitly mentioned otherwise, quotes are from[25].

5. Predictive behaviours are typical of *proactive* systems, namely systems that base their action upon a hypothesised future state computed through some model. In[25] predictive behaviours are further classified according to their "order", namely the amount of context variables their models take into account. Thus a system tracking the speed of another system to anticipate its future position exhibits first-order predictive behaviours, while one that considers, e.g., speed and flightpath, is second-order predictive. Systems constructing their models through the correlation of two or more "raw fact" dimensions, possibly of different nature, are called higher-order predictive systems.

The above model of individual behaviour may be naturally extended by considering collective behaviours—namely the conjoint behaviours of multiple individual systems. We distinguish three major classes of collective behaviour:

1. Neutral social behaviour. This is the behaviour resultant from the collective action of individual, purposeful, non teleological behaviours. Each participant operates through simple reflexes, e.g., "in case of danger get closer to the flock". Lacking a "signal from the goal", the rationale of this class of collective behaviours lies in the benefits deriving by the sheer number of replicas available. Examples include defencive behaviour of a group of individuals from a predator and group predation.
2. Individualistic social behaviour. This is the social behaviour of systems trying to benefit opportunistically in a regime of competition with other systems. Here participants make use of more complex behaviours that take into account the social context, namely the behaviours exercises by the other participants. It is worth noting how even simple "systems" such as bacteria may exercise this class of behaviour[26].
3. Cooperative or coopetitive social behaviours. These are social behaviours of systems able to establish mutualistic relationships (mutually satisfactory behaviours) and to consider proactively the future returns deriving from a loss in the present. Examples of behaviours in this class are, e.g., the symbiotic relationships described in[6,27].

As a final remark we deem worth noting how resilience and change tolerance are not absolute properties: in fact they emerge from the match with the particular conditions being exerted by the current environment. This means that it is not possible to come up with an "all perfect" solution able to withstand whatever such condition. Nature's answer to this dilemma is given by redundancy and diversity. Redundancy and diversity are in fact key defence frontlines against turbulent and chaotic events affecting catastrophically an (either digital or natural) ecosystem. Multiple and diverse "designs" are confronted with events that determine their fit. Collective behaviours increase the chance that not all the designs will be negatively affected. In this sense we could say that "resilience abhors a vacuum", for empty spaces—namely unemployed designs and missed diversity—may potentially correspond to the very solutions that would be able to respond optimally to a catastrophic event.

A treatise of collective behaviours is outside the scope of this paper. Interested readers may refer to, e.g.,[4,26,28,29].

3.3. Fragility as a measure of assumptions dependence

The type of behaviour exercised by a system constitutes—we deem—a second important characteristic of that system with reference to its ability to improve dynamically its system-environment fit. This "second coordinate" of a system's fidelity to systemic, operational, and environmental assumptions is meant to bring to the foreground how dependant an open system actually is on its **system model**—namely, on its prescribed assumptions and working conditions[30,12].

Classes of resilient behaviours allow us to assess qualitatively a system's "fragility" (conversely, robustness) to the variability of its environmental and systemic conditions. As an example, let us consider the case of traditional electronic systems such as, e.g., the flight control system that was in use in the maiden flight of the Ariane 5 rocket. A common trait in such systems is that enacted behaviours are mostly very simple (typically purposeful but non-teleological). While this enhances efficiency and results in a lean and cost-effective design, one may observe that it also produces a strong dependence on prescribed environmental conditions. It was indeed a mismatch between the prescribed and the experienced conditions that triggered the chain of events that resulted in the Ariane 5 failure[19].

4. Beyond both Elasticity and Resilience

We have introduced two complementary models to reason about the fidelity of open systems. The two models are orthogonal, in the sense that they represent two independent "snapshots" of the system under consideration:

1. The Algebraic model regards the system as a predefined, immutable entity. Conditions may drift but the system exhibits no complex "motion"—no sophisticated active behaviours are foreseen in order to reduce the drift.
2. The behavioural model captures instead the dynamicity of the system. Also in this case the system measures the system-environment fit, but the system may actively use this measure in order to optimise its quality.

Intuitively, the first model is backed by redundant resources dimensioned through worst-case analyses; events potentially able to jeopardise quality are *masked* out. The minimal non-functional activity translates in low overhead and simple design. Embedded systems typically focus on this approach.

Conversely, the second model calls for complex abilities—among others awareness; reactive and proactive planning; quorum sensing[26]; collective strategy planning[4], etc. Events jeopardising quality are *tolerated* rather than masked; moreover, complex analyses and strategies are mandated by the overhead typically associated with non-functional behaviours. This notwithstanding, said behaviours may be the only effective line of defence against the highly dynamic environments characterising *open* embedded systems (such as cyber-physical things[3]) and, *a fortiori*, future collective cyber-physical societies[5] and fractal social organisations[9].

Though orthogonal in their assumptions, the overheads associated with the design approaches corresponding to the above two models are not side-effect free (suffice it to consider the effect of complex behaviours on the worst-case analyses called for, e.g., by hard real-time systems). Our tentative answer to this problem is given by a new general scheme revising the one presented in Sect. 3.1. In the new scheme the systems perform as follows:

1 Monitor the drifting of their Δ functions and possibly other context figures providing insight on the current environmental conditions.

2 Build simple, low-overhead models of the turbulence and chaotic nature of their environments.

3 While the current conditions and trend are deemed as "unsafe", repeat:

4.1 Build and maintain more complex reactive and proactive models to understand how the drifting is impacting on one's system identity.

4.2 Plan and enact corrective behaviours choosing between the following two options:

4.2.1 Self-reconfiguration: the system reshapes itself by choosing new system structures and new designs best matching the new environmental conditions. Examples of self-reconfiguration strategies may be found, e.g., in[19].

4.2.2 Establish social relationships with neighbouring systems. This may include for instance simple actions such as "join collective system" or "leave collective system", opportunistic strategies such as "improve one's Δ's to the detriment of those of neighbouring systems", or complex mutualistic relationships involving association, symbiosis, or mutual assistance. An example of said mutualistic relationships is described in[15].

4.3 Measure the effectiveness of the attempted solutions, rank them with respect to past solutions, derive and persist conclusions, and update the reactive and proactive models accordingly.

As can be clearly seen from its structure, the above scheme distinguishes two conditions: one in which system identity is not at stake, and correspondingly complexity and overhead are kept to a minimum, and one when new conditions are emerging that may result in identity failures—in which case the system switches to more complex behaviours. A self-managed, dynamic trade-off between these two approach, we conjecture, may provide designers with a solution reconciliating the benefits and costs of both options. We refer to future systems able to exercise said dynamic trade-offs as to "auto-resilient"—a concept first sketched in[15].

As a final remark, the machine learning step **4.3** in the above scheme implies that the more a system is subjected to threats and challenging conditions, the more insight will be acquired on how to respond to new and possibly more threatening situations. We conjecture that insight in this process may provide the designers with guidelines for engineering *antifragile* cyber-physical systems[14].

5. Conclusions

We presented two orthogonal models for the synchrony and real-timeliness of open computer systems such as modern electronic systems[2], cyber-physical systems, and collective organisations thereof. We discussed how each of the two models best-match certain operational conditions—the former, stability; the latter, dynamicity and turbulence. Finally, we proposed a scheme able to self-optimise system processing depending on the experienced environmental conditions. As the scheme also includes a machine learning step potentially able to enhance the ability of the system to adjust to adverse environmental conditions we put forward the conjecture that antifragile systems may correspond to systems able to learn while enacting elastic and resilient strategies. Future work will be devoted to simulating compliant systems with the support of self-adaptation frameworks such as ACCADA[31,32] and Transformer[33,34].

References

1. Heylighen, F.. Basic concepts of the systems approach. In: Heylighen, F., Joslyn, C., Turchin, V., editors. *Principia Cybernetica Web*. Principia Cybernetica, Brussels; 1998,
2. Munaga, S., Catthoor, F.. Systematic design principles for cost-effective hard constraint management in dynamic nonlinear systems. *International Journal of Adaptive, Resilient and Autonomic Systems* 2011;**2**(1).
3. Lee, E.A.. Cyber physical systems: Design challenges. Tech. Rep. UCB/EECS-2008-8; EECS Department, University of California, Berkeley; 2008.
4. Astley, W., Fombrun, C.J.. Collective strategy: Social ecology of organizational environments. *Academy of Mgmt. Rev.* 1983;**8**:576–587.
5. Zhuge, H.. Cyber physical society. In: *Semantics Knowledge and Grid (SKG), 2010 Sixth Int.l Conference on*. 2010, p. 1–8.
6. Sun, H., De Florio, V., Gui, N., Blondia, C.. The missing ones: Key ingredients towards effective ambient assisted living systems. *Journal of Ambient Intelligence and Smart Environments* 2010;**2**(2).
7. De Florio, V., Blondia, C.. Service-oriented communities: Visions and contributions towards social organizations. In: *On the Move to Meaningful Internet Systems: OTM 2010 Workshops*; vol. 6428 of *LNCS*. Springer; 2010, p. 319–328.
8. Latour, B.. On actor-network theory. a few clarifications plus more than a few complications. *Soziale Welt* 1996;**47**:369–381.
9. De Florio, V., Bakhouya, M., Coronato, A., Di Marzo Serugendo, G.. Models and concepts for socio-technical complex systems: Towards fractal social organizations. *Systems Research and Behavioral Science* 2013;**30**(6).
10. De Florio, V., Blondia, C.. Reflective and refractive variables: A model for effective and maintainable adaptive-and-dependable software. In: *Proc. of the 33rd EUROMICRO Conf. on Software Engineering and Advanced Applications (SEAA 2007)*. Lübeck, Germany; 2007.
11. Kanai, R., Tsuchiya, N.. Qualia. *Current Biology* 2012;**22**(10):R392–R396.
12. De Florio, V.. *Application-layer Fault-Tolerance Protocols*. IGI Global, Hershey, PA; 2009. ISBN ISBN 1-60566-182-1.
13. Cristian, F., Fetzer, C.. The timed asynchronous distributed system model. *IEEE Trans. on Parallel and Distr. Systems* 1999;**10**(6):642–657.
14. Taleb, N.N.. *Antifragile: Things That Gain from Disorder*. Random House Publishing Group; 2012. ISBN 9781400067824.
15. De Florio, V.. Preliminary contributions towards auto-resilience. In: *Proc. of SERENE 2013, LNCS 8166*. Springer; 2013, p. 141–155.
16. De Florio, V.. On the role of perception and apperception in ubiquitous and pervasive environments. In: *Proceedings of the 3rd Workshop on Service Discovery and Composition in Ubiquitous and Pervasive Environments (SUPE'12)*. 2012.
17. Boulding, K.. General systems theory—the skeleton of science. *Management Science* 1956;**2**(3).
18. Leibniz, G., Strickland, L.. *The shorter Leibniz texts: a collection of new translations*. Continuum impacts. Continuum; 2006.
19. De Florio, V.. Software assumptions failure tolerance: Role, strategies, and visions. In: Casimiro, A., de Lemos, R., Gacek, C., editors. *Architecting Dependable Systems VII*; vol. 6420 of *Lecture Notes in Computer Science*. Springer; 2010, p. 249–272.
20. Charette, R.. Electronic devices, airplanes and interference: Significant danger or not? 2011. `http://spectrum.ieee.org/riskfactor/aerospace/aviation/electronic-devices-airplanes-and-interference-significant-danger-or-not`.
21. De Florio, V.. On the constituent attributes of software and organisational resilience. *Interdisciplinary Science Reviews* 2013;**38**(2).
22. Sachs, J.. *Aristotle's Physics: A Guided Study*. Masterworks of Discovery. Rutgers University Press; 1995. ISBN 0-8135-2192-0.
23. Aristotle, , Lawson-Tancred, H.. *De Anima (On the Soul)*. Penguin classics. Penguin Books; 1986.
24. Runes, D.D., editor. *Dictionary of Philosophy*. Philosophical Library; 1962.
25. Rosenblueth, A., Wiener, N., Bigelow, J.. Behavior, purpose and teleology. *Philosophy of Science* 1943;**10**(1):18–24.
26. Schultz, D., Wolynes, P.G., Ben Jacob, E., Onuchic, J.N.. Deciding fate in adverse times: Sporulation and competence in bacillus subtilis. *Proc Natl Acad Sci* 2009;**106**:21027–21034.
27. Sun, H., De Florio, V., Gui, N., Blondia, C.. Participant: A new concept for optimally assisting the elder people. In: *Computer-Based Medical Systems, 2007. CBMS '07. Twentieth IEEE International Symposium on*. 2007, p. 295 –300.
28. Sousa, P., Silva, N., Heikkila, T., Kallingbaum, M., Valcknears, P.. Aspects of co-operation in distributed manufacturing systems. *Studies in Informatics and Control Journal* 2000;**9**(2):89–110.
29. Guseva, K.. *Formation and Cooperative Behavior of Protein Complexes on the Cell Membrane*. Ph.D. thesis; Institute of Complex Systems and Mathematical Biology of the University of Aberdeen; 2012.
30. De Florio, V., Deconinck, G.. On some key requirements of mobile application software. In: *Proc. of the 9th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS)*. Lund, Sweden: IEEE Comp. Soc. Press; 2002, .
31. Gui, N., De Florio, V., Sun, H., Blondia, C.. Toward architecture-based context-aware deployment and adaptation. *Journal of Systems and Software* 2011;**84**(2):185–197.
32. Gui, N., De Florio, V., Sun, H., Blondia, C.. ACCADA: A framework for continuous context-aware deployment and adaptation. In: *Proc. of the 11th Int.l Symp. on Stabilization, Safety, and Security of Distr. Sys., (SSS 2009)*; vol. 5873 of *LNCS*. Springer; 2009, p. 325–340.
33. Gui, N., De Florio, V.. Towards meta-adaptation support with reusable and composable adaptation components. In: *Proceedings of the sixth IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2012)*. IEEE; 2012.
34. Gui, N., De Florio, V., Holvoet, T.. Transformer: an adaptation framework with contextual adaptation behavior composition support. *Software: Practice & Experience* 2012.