Towards Improving Adaptability of Capability Driven Development Methodology in Complex Environment

Renata Petrevska Nechkoska[1,2(✉)], Geert Poels[1], and Gjorgji Manceski[2]

[1] Faculty of Economics and Business Administration, Ghent University, Ghent, Belgium
{renata.petrevskanechkoska, geert.poels}@ugent.be
[2] Faculty of Economics, University St. Kliment Ohridski, Bitola, Macedonia
gmanceski@t-home.mk

Abstract. We are triggered to incorporate adaptability in information system designs and methodologies corresponding to complex and unpredictable environment of today and tomorrow and to complex adaptive systems they are aimed for. Adaptability as non-functional requirement is being portrayed and investigated from broad multidisciplinary perspective that influences how dynamic business-IT alignment can be accomplished. Capability Driven Development methodology has supported delivering dynamic capabilities by providing context-aware self-adaptive platform in the CaaS project implementations, as our case study. Along with the already incorporated mechanisms, components that enable adaptability, there is open space for further evolutionary and deliberate change towards becoming truly appropriate methodology for dynamic reconfigurations of capabilities in organizations and business ecosystems that operate in complexity and uncertainty. The analysis and evaluation of adaptability of the CDD methodology through three dimensions (complexity of the external and internal environment, managerial profiling and artifact-integrated components) in this paper conclude with instigation of starting points towards achieving higher adaptability for complexity of the CDD methodology.

Keywords: Adaptability · Adaptiveness · Adaptation
Non-functional requirements · Capability Driven Development methodology
Complexity

1 Introduction

Adaptability is emerging as an important type of non-functional requirement (NFR) for just about any system, including information systems, embedded systems, e-business systems, and the like. It represents the system's ability to accommodate changes in its environment - in order to succeed or even to survive [1]. Especially in the service design phase, there is the additional requirement for high system adaptiveness along different technical requirements and different user expectations [2]. Complementary to the basic qualities (functionality, reliability, ease of use, economy and safety) there are

R. Matulevičius and R. Dijkman (Eds.): CAiSE 2018 Workshops, LNBIP 316, pp. 15–27, 2018.
https://doi.org/10.1007/978-3-319-92898-2_2

extra qualities, NFRs or soft-goals – flexibility, reparability, adaptability, understandability, documentation and enhanceability [3, 4].

The ability for system to change is essential to its continued survival and ability to provide requisite functions for its stakeholders [5] either through evolutionary [6] or goal-seeking, deliberate adaptation [7]. According the NFR (non-functional requirements) framework [3], there are functional requirements specifying 'what' should be achieved, and non-functional requirements - specifying 'how' outcomes will be achieved. Generally, the NFRs are "informally stated, often contradictory, difficult to enforce during development and evaluate for the customer prior to delivery" evaluated subjectively and qualitatively ('satisfying' or 'not satisfying') [8]. Adaptability as NFR is defined as the ease of system/component modification, modification of behavior in response to environment changes, adjustment to changing requirements [5].

Capability Driven Development (CDD) applies enterprise models representing enterprise capabilities to create executable software with built-in contextualization. It attempts to overcome the limitations of Model Driven Development (MDD) towards more suitable capture of business requirements, modeling execution contexts, offering functionality in different business contexts, capturing dynamic behavior of both functional and non-functional requirements – all of which enabling 'plasticity' in software applications that are business context aware [10]. It situates itself in complex and dynamically changing business environments, incorporating principles of agile and iterative IS development thus enabling continuous dynamic business-IT alignment in a structured and systematic way, using the concept of business capability. CDD aims for rapid response to changes in the business context and development of new capabilities which also requires run-time configuration and adjustment of the IS [11–13].

The external environment in which we operate is complex and unpredictable, portrayed in the Cynefin framework [14–18], and the Stacey matrix [19–21] (Fig. 1). This imposes specific considerations to deliver dynamic capabilities, in terms of managerial approach, internal environment as Complex Adaptive Systems (CAS) [22].

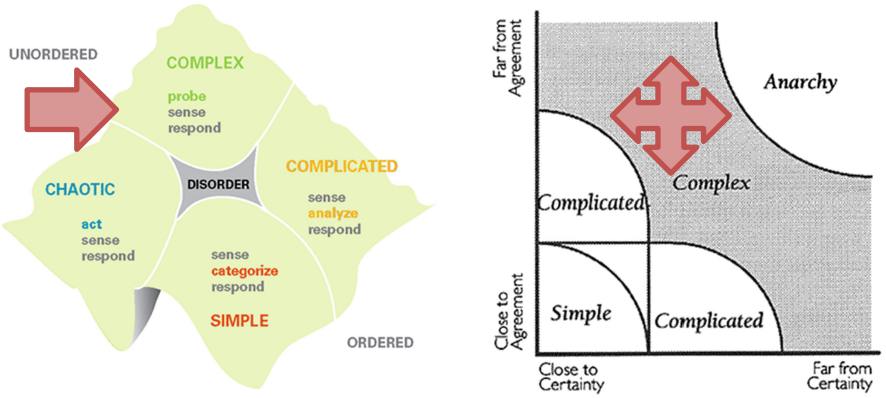

Fig. 1. The Cynefin Framework and recommended managerial approaches for complexity (left) [16, 60] and Stacey Matrix and managerial approaches for complexity (right) [20, 21]

This paper compiles a novel qualitative evaluation framework that investigates adaptability for complexity, using the case of the CDD as state of the art methodology designed for function in complex and unpredictable environment, through its incorporation in the CaaS project, as one of its most comprehensive, robust and exemplary implementations. Using this evaluation prism we detect and point out the existence of components of adaptability in CDD methodology (element and architectural) through 3 dimensions, and instigate future directions to improve CDD methodology and its effectiveness in supporting context-aware, self-adaptive platforms that model and deliver dynamic capabilities, such as CaaS.

The paper is structured as follows: In Sect. 2 we are discussing the main concepts of adaptation, adaptability and adaptiveness, meaning, names; as well as how they can be achieved and measured. In Sect. 3 we are decomposing the evaluation prism to aspects that ought to be incorporated in the adaptability components (on architectural and element level) and investigating in qualitative manner their existence, implicit incorporation or non-existence in the CDD methodology through 3 dimensions. Section 4 concludes the evaluation, assembles the recommendations for improvement, opening horizons for future multidisciplinary research.

2 Main Concepts

Adaptability and Adaptation. 'Adaptation means change in the system to accommodate change in its environment' [8, 9]. "Adaptation of a system (S) is caused by change from an old environment (E) to a new environment (E') and results in a new system (S') that ideally meets the needs of its new environment (E')" [9]. Adaptability involves three tasks: environment change detection, system change recognition and effectuating system change. The environment can be observed as inner and outer and changes can derive from it all – with regards to Complex Adaptive Systems [23–26]. Some changes entity needs to adapt to, but also changes being initiated in order to reach purpose(s), goal(s) – in the sense of evolutionary and revolutionary learning and adaptation as well as double loop learning [27–29]. The route of changes within the organization range from changes on operational level, or in resources, changes in goals, expectations, principles, KPIs [30].

Adaptability, from systems engineering perspective, as architectural property, is defined as: 'Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments' [5]. "A characteristic of a system amenable to change to fit altered circumstances, where 'circumstances' include both the context of a system's use and its stakeholders' desire" is definition of adaptability by [31]. Adaptability as 'degree to which adjustments in practices, processes or structures of systems are possible to projected or actual changes of its environment' [32, 33] has the elements of what it is (degree of adjustment), to which changes it responds (projected or actual), and how it can be achieved (through adjustments in practices, processes or structures). In taxonomy of service qualities (described as a behavior), adaptability is alongside availability, assurance, usability, interoperability, scalability, portability, extensibility [34, 35].

In [5] it is in changeability, and in COBIT [36], it is into supportability. The authors [37, 40] use adaptability and flexibility as concepts. In [41] ISO/IEC 25010:2011 for Systems and software Quality Requirements and Evaluation (SQuaRE), the definition of flexibility as 'adapting a product for additional user groups, tasks, cultures, enabling products to take account of circumstances, opportunities and individual preferences that had not been anticipated in advance' fits best.

Achieving and Measuring Adaptation. Our next point of interest is to analyze various approaches achieving adaptation. Today's CAS are 'socio-technical, characterized by the interplay between social and technical components, consisted of human actors but also software, hardware; representing the environment (the context) to which systems need to be aware of and functioning in. Context is the current state of the user or system to the objects in their surrounding environment. Human and automatic components are involved in the process of identification of and response to the context. Contextualization is used as a way to allow adaptation to changes both at design time and at run-time' [42]. Adaptation is done through monitoring and actuation. A system is adaptable if it can be altered by someone, while adaptive if it can sense the need and generate the alteration from within itself.

Rule-based adaptation, as analyzed in [43, 44] recognizing 'content analysis rules, content adaptation, corrective, enhancing, fuzzy, integration, monitor, production, matching rules'. They are directed towards various entities (concerning adaptable software) such as process adaptation, workflow and service-flow adaptation, content, GUI/AUI, software configuration, features adaptation. Adaptability transformations enable implementing simple and complex transformations through composition of basic refactoring; sensors and effectors; and design patterns [45–48].

Variability in the field of requirements engineering, variability analysis focuses on 'prioritizing one or few possible solutions to be implemented in the final product, with the strive to enable users to adjust and adapt the product as needed' [37, 49].

The Tropos development methodology in information system design is based on i* organizational modeling framework, through early requirements, late requirements, architectural design and other detailed dependencies [37]. [38–40] use mapping and measurement of adaptability, turbulence and adaptability indices, focused mainly on the external environment and business dimension. Founded on CAS approach, the analysis of IS architecture complexity paralleled with IS efficiency and flexibility (as opposing characteristics that can be mediated by evolutionary and revolutionary IS change), is the approach of [50]. Research in CAS specificity incorporates top-down 'official' and bottom-up 'emergent' co-evolutionary adaptation of information systems design with changing user requirements towards effective system design [51]. PAWS as framework for executing adaptive web-service processes [2, 52] aims for 'self-configuring, self-optimizing, self-healing, self-protecting computing systems'.

Through decomposition of the NFR of interest the POMSA framework (Process-Oriented Metrics for Software Architecture [9]) investigates adaptability of system's components, connections, patterns, constraints, styles that reflect changeability (decomposability, cohesiveness, understandability, simplicity), replaceability, reusability.

Important aspects of the adaptability of any system are controls ranging from classic, advanced, knowledge-based, game/queuing theory controls, feedback and feed-forward controls [51, 53, 54]. Authors [55–57], distinguish: semantic, syntactic, contextual and quality adaptation; [1, 8, 9] recognize: element and architecture adaptability incorporating effectors and adaptors and signature level (level of the entity), protocol level, service level and semantic level of adaptation.

3 Evaluating Adaptability of Capability Driven Development (CDD) for Complex Environment

The main challenges designers of CDD methodology have in front of themselves [10, 11, 13] in the CaaS implementations, are what CDD methodology should achieve: ***to model the desired capabilities – using dynamic capabilities that contain variability; to model the impact of context; towards context-aware self-adaptive platform.***

The primary purpose [58] of the CDD methodology is directed towards increasing the value of business services by providing efficient development methodology and capability management lifecycle to continuously design and deliver capabilities that are adjusted for the context of use. We will be examining the adaptability components both through element and architectural prism, in an attempt to perceive how CDD methodology can achieve semantic, syntactic, contextual and quality adaptation on conceptual level, as being implemented and enhanced by the CaaS project.

The three main dimensions for achieving adaptation in complexity that represent frame of analysis are: *Complexity of the environment (External and Internal), Managerial (Strategic, Tactical, Operational) profiling, Artifact-integrated components.*

These three dimensions incorporate a set of interrelated and complex aspects that need to be present on architectural and elementary level of a CDD-like methodologies to achieve higher level of adaptability as necessary NFR for addressing complexity. The qualitative assessment of the important aspects that compose the dimensions is threefold: 'Satisfying (+/+)', 'Semi-satisfying (+/–)', 'Non-satisfying (–/–)'. The evaluation results with starting points for improvement of certain aspects of the methodology towards greater adaptability for complexity (Table 1).

Dimension 1. The external environment is consisted of interrelated agents networked together (in known and unknown manner to the observer, manager, facilitator) producing emergent effect where cause and effect, only coherent in retrospect. Its complexity is perceived in the incomplete knowledge for all the external and internal relations among the entities and here *Probe-Sense-Respond strategy* fits best (Cynefin framework [14–18] (Fig. 1, left)). In the Stacey matrix [19–21] (Fig. 1, right), which considers certainty of outcome and agreement on outcome for various management approaches (relevant here through the decision logic incorporated in CDD), the zone of complexity enlists un-programmable decision making, 'outcomes' instead of 'outputs and solutions'. Organizations as socio-technical Complex Adaptive Systems (CAS) are the internal complex environment. *CAS characteristics* of nonlinearity, self-organization, emergence, co-evolution initiate a question: how do we facilitate a complex adaptive system towards purpose(s) and emergent effects? CAS need to be

Table 1. Main dimensions and their interrelated aspects for analyzing and evaluating adaptability as non-functional requirement, case of CDD methodology

Dimension 1: Complexity of the environment (external & internal)	Dimension 2: Managerial (Strategic, Tactical, Operational) Profiling	Dimension 3: Artifact-integrated components
Probe-Sense-Respond strategy (+/–) CAS characteristics (+/–) Broad business ecology (+/+) Multifaceted context capture (+/+) SIDA & PDCA loops (+/+) Top-down/bottom-up/lateral learning (+/–)	Clarification and proper addressing of strategy, tactics, operations (+/–) Purposeful/Purposive system (+/–) Outcomes/Outputs (+/–) Qualitative/Quantitative information (+/–)	Adaptability transformations (+/+) Variability support (+/+) Modularity (+/+) Positive and negative feedback (+/+) Patterns (±)

addressed with (1) simple rules, (2) moderately dense connections, (3) human rules on how to detect information, how to interpret information, how to act in response [22].

In the example of [13] 'the search for error' approach in method component for context modeling exists in definition of exception types, investigation, development, linkage with the context sets; run-time adaptation options. This supports un-programmable decision making and identification-development-selection approach in CDD. 'Probe' is missing of the Probe-Sense-Respond and is assessed as (+/–).

In the example of [61], CDD's runtime adaptation options range from fully automatic, semi-automatic, to manual – where the system discovers change needs but the change in itself is too complex to implement and human intervention is needed to handle the situation [61]. *CAS specificities* towards emergent effect, as well as loose relations and dependencies, alternative influence on other capabilities, business goals [68] are not fully incorporated in the design or exhibited in the implementations (±).

CDD in the example [69] uses the Onion framework portraying all considerations taken into account in the architectural design and clarifying that the context is not the only source of variability incorporating *broad business ecology* (+/+) through a scope of entities that describe and detect the context considering invariant characteristics such as domain, entity, problem, dynamic processes. In component 1 of the architecture of the CDD environment for context modeling, it is clearly visible that the inputs, the data providers can be internal and external, while contextual elements are captured in multifaceted manner (device, user, task, data source, document representation, time, geo-location) – ensuring *comprehensive multifaceted context capture* (+/+).

The adaptability loop *Sense-Interpret-Decide-Act (SIDA)* or OODA, is effectuated through the CDD actions of capturing context (Sense/Observe) – use patterns & predict (Interpret/Orient) – decision logic (Decide) – runtime adjust, pattern update, deploy (Act) (Fig. 2 (left). However, the loop detects changes in external context, the system continues to reason according pre-defined decision logic on how to react. If we expand the need for the system to detect the changes in expected outcomes (KPIs, goals) and reconfigure accordingly, it may have limited response within a 'given' set of alternative reactions, requiring new system instance. The *top-down approach* should be combined

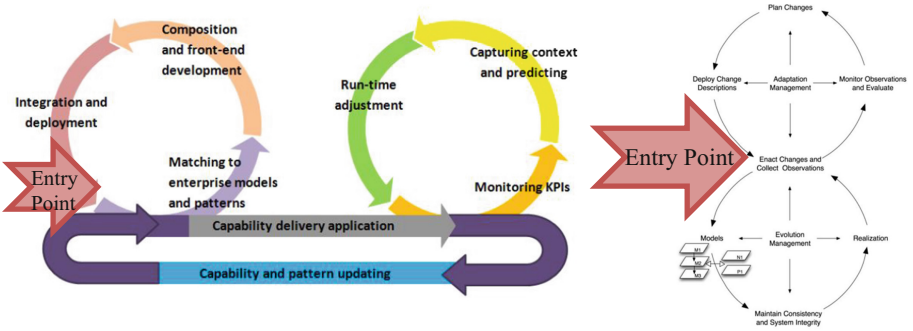

Fig. 2. Capability Driven Development Methodology (left) [10], and Processes involved in runtime adaptation (right) [59]

with *bottom-up and lateral learning* (Fig. 4). *Plan-Do-Check-Act* loop is the one necessary to initiate deliberate changes or model planned responses (evolution and adaptation management). For complex environment, it may have been better situated on secondary level, immediately after the execution, in the process of learning (Fig. 4). Its existence is sufficient mechanism for deliberate change (Fig. 2 (right)) assessed as (+/+). Adaptation and evolution management are in perfect constellation for context-aware CDD. Applicability in various contexts is directly handled by method extension for evolutionary development of capabilities delivered as software service bundles where the system allows the client switch from one configuration to another to adapt to changes.

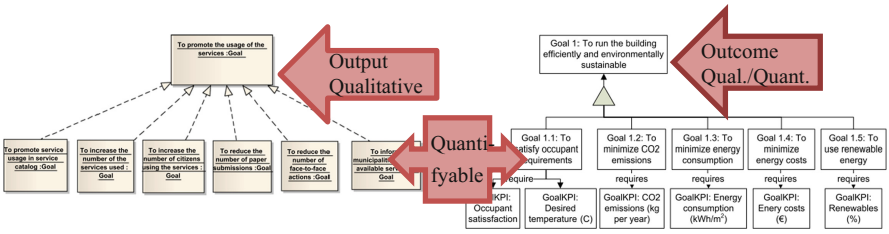

Fig. 3. To promote usage of the municipality services (left) [10], and Generic goal model for building operator (right) [59]

Dimension 2. Clarification and proper *addressing of the managerial functions – strategy, tactics, operations* is especially important for complexity. CDD binds with strategic management and operationalizes directly. For tactical and strategic management, it needs better focus (±). '*Purposive system* is multi-goal- seeking system whose goals result with the system's purpose. This type of system can pursue different goals but it does not select the goal to be pursued. It does choose the means by which to pursue its goals. *Purposeful system* is one which can change its goals, it selects ends as well as means, and thus displays will.' [64] CDD methodology supports development

of purposive systems – dynamically reconfiguring towards goals, primarily changing own behavior, by choosing means to pursue goals. It doesn't, aim to offer purposeful search for goals and strategy (±). CDD expects stable strategy [59], given goals [66] and KPIs providing dynamic capabilities. To avoid the mismatch between systems and their models [27, 64] we need not to rely not just on simple/complicated (but deterministic) mechanistic relations among entities and among outputs (±). CDD follows top-down the business model, enterprise architecture, context and capabilities - which are consisted of *qualitative and quantitative values*. Goals and KPIs are values and ranges, with 'hard-coded' decision logic and measurable properties [67], but there is a lot of space for adding qualitative inputs to relate to reality. Goals and KPIs are *mixture of outputs and outcomes*, in *qualitative and quantitative terms* (evaluated with (±)). A goal model in example [10] declares 'to promote the usage of the services' (Fig. 3, left) as qualitative output (put all efforts into promoting usage of the services) decomposed as: promote service usage in service catalog, increase the number of services used, number of citizens, reduce number of paper submissions. Outcome would be 'achieve increased usage of services by X% by customer Y' portraying mixture of outcomes/outputs with mostly quantitative KPIs. In reality, these issues cannot and should not be quantifiable, but combine with qualitative information towards adaptive management on 'how' to make choices.

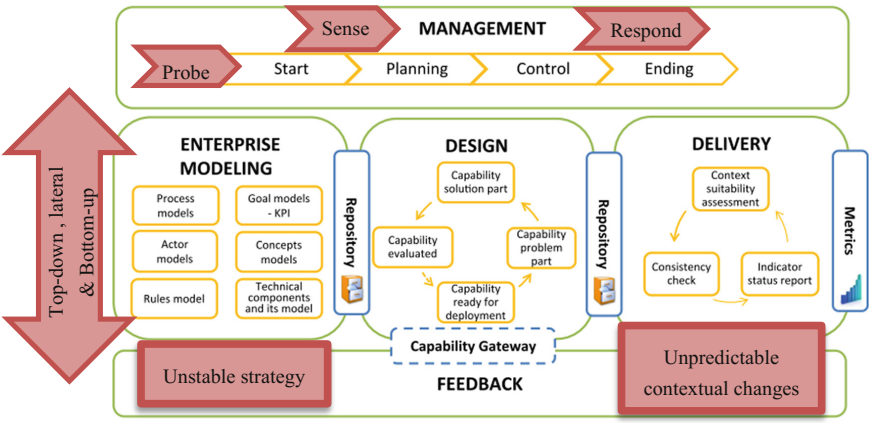

Fig. 4. CDD Methodology [59] and adaptability needs for complex environment

Dimension 3. CDD has numerous important components integrated in its core providing adaptability. *Adaptability transformations,* evaluated with (+/+) help adaptability at most granular level, like refactorings, creational transformations required by sensors and effectors complementing *variability* which requires existing variables receiving various parameters in introducing new blocks to be composed dynamically at run-time striving to enable users adjust and adapt the product as needed [37]. Adaptation as adjustment on baseline level (case study [13]) incorporates scheduled and event-based adjustment coordinated through integrated procedure, use/re-use the constant multiple instances [13]. *Positive and negative feedback* in capability delivery

adaptation can be triggered by search based adaptation allowing deliberate adaptation as structural adjustment [65], with context modelling in alternative method pathways and in the capability pattern lifecycle. Here predictive analysis runs adjustment algorithm that adapts capability delivery in response to changes in context to meet capability delivery goals (evaluated with (+/+)). CDD method extension with strategies: Global-as-local, assuming overall optimization changing behavior of local system which requires information about global one; Local-as-Global, assuming local systems adapting behavior using only their local context information makes bridge human intervention and re-alignment of strategies across the ecosystem. *Patterns* in CDD are reusable solutions for reaching business under different situational context [13]. In the example [62] they filter contexts and perspectives relating with complex environment where the probe-sense-respond approach is recommended. Patterns are recommended initially according the capability structure, currently applied patterns and contextual conditions, but CDD can also provide run-time recommendations of other patterns to perform better in given context and situation. However, patterns in CDD are mostly perceived for efficiency, while in complexity, patterns also help orient, gain knowledge about the context, how to make proper moves in the solution space [63] (±).

4 Conclusion

CDD methodology incorporates many necessary components and traits on element and architectural level to support development of context-aware self-adaptive platforms delivering dynamic capabilities (Fig. 4). Dimensions of adaptability as NFR for complex environment reflect complexity of the environment (external, internal), managerial profiling and artifact components influencing semantic, syntactic, contextual and quality adaptation. CDD has SIDA & PDCA loops necessary for evolutionary and goal-seeking adaptation to crawl through the problem/solution space; multifaceted context capture, open information flows from broad business ecology to detect changes, effectors to address them; diverse range of adaptability transformations to provide extensive variability; modularity. Envisioned improvement points suggested here are in direction of CDD methodology configuration to face unstable strategy in dynamic reconfiguration of capabilities, design to learn bottom-up, laterally and top-down. The Interpret-Decide stages (of the SIDA loop) can be enhanced – detection and configuration of patterns should transcend the notion of using them for efficiency but also effective situation awareness about the context and the solution space. CDD needs to combine qualitative and quantitative information in unprogrammable decision making, through clarification of outcomes (customer-back defined and accomplishment of purpose) or outputs (company-forward definition and judgement of accomplishment of goals), especially on strategic and tactical level where the decision logic, purposefulness or purposiveness of the managerial function play role. And last, but not least, adaptive and adaptable denote different abilities – and different systems' behavior. In this paper we investigated adaptability of CDD methodology, while the true adaptiveness of socio-technical artifacts, is predominantly unaccomplished mission.

References

1. Chung, L., Subramanian, N.: Adaptable system/software architectures. J. Syst. Archit. **50**(7), 365–366 (2004)
2. Cappiello, C., Comuzzi, M., Mussi, E., Pernici, B.: Context management for adaptive information systems. Electron. Notes Theor. Comput. Sci. **146**(1), 69–84 (2006). SPEC. ISS.
3. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Springer, Boston (2000). https://doi.org/10.1007/978-1-4615-5269-7
4. Borgida, A.T., Chaudhri, V.K., Giorgini, P., Yu, Eric S. (eds.): Conceptual Modeling: Foundations and Applications. LNCS, vol. 5600. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02463-4
5. Adams, K.M.: Adaptability, flexibility, modifiability and scalability, and robustness. Nonfunctional Requirements in Systems Analysis and Design. TSRRQ, vol. 28, pp. 169–182. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18344-2_9
6. National Geographic Society, "National Geographic Encyclopedia - Adaptation," Encyclopedia of Evolution (2018). https://www.nationalgeographic.org/encyclopedia/adaptation/
7. New England Complexity Science Institute, Concepts: Adaptive, NECSI (2018)
8. Subramanian, N., Chung, L.: Metrics for software adaptability. Softw. Qual. Manag. SQM **2001**, 95–108 (2001)
9. Chung, L., Subramanian, N.: Process-oriented metrics for software architecture adaptability. In: Proceedings of IEEE International Conference Requirements Engineering, pp. 310–311 (2001)
10. Bērziša, S., Bravos, G., Gonzalez, T.C., Czubayko, U., España, S., Grabis, J., Henkel, M., Jokste, L., Kampars, J., Koç, H., Kuhr, J.-C., Llorca, C., Loucopoulos, P., Pascual, R.J., Pastor, O., Sandkuhl, K., Simic, H., Stirna, J., Valverde, F.G., Zdravkovic, J.: Capability driven development: an approach to designing digital enterprises. Bus. Inf. Syst. Eng. **57**(1), 15–25 (2015)
11. Zdravkovic, J., Stirna, J., Grabis, J.: A comparative analysis of using the capability notion for congruent business and information systems engineering. Complex Syst. Inf. Model. Q. **10**, 1–20 (2017)
12. Bērziša, S., España, S., Grabis, J., Henkel, M., Jokste, L., Kampars, J., Koç, H., Sandkuhl, K., Stirna, J., Valverde, F., Zdravkovic, J.: Capability as a Service in digital enterprises Task 5. 1 Result Report : State-of-the-Art in relevant methodology areas, no. 611351 (2014)
13. Bērziša, S., España, S., Grabis, J., Henkel, M., Jokste, L.: Deliverable 5.3: The Final Version of Capability Driven Development Methodology-Capability as a Service in digital enterprises (2016)
14. Snowden, D.J.: Complex acts of knowing: paradox and descriptive self- awareness. J. Knowl. Manag. **6**(2), 100–111 (2002)
15. Snowden, D.J.: Multi-ontology sense making: a new simplicity in decision making. Inform. Prim. Care **13**(1), 45–53 (2005)
16. Kurtz, C.F., Snowden, D.J.: The new dynamics of strategy: sense-making in a complex-complicated world. IBM Syst. J. **42**(3), 483 (2003)
17. Snowden, D.J., Boone, M.E.: A leader's framework for decision making. Harvard Bus. Rev. **85**(11), 68–76 (2007)
18. Dettmer, B.H.W.: Systems thinking and the Cynefin Framework: a strategic approach to managing complex system. Goal Syst. Int., 1–13 (2007)
19. Stacey, R.: Strategic Management and Organisational Dynamics. Pearson Education, London (1996)

20. Stacey, R.: Complexity and management papers. Gr. Anal. Complex. Manag. Pap. **35**, 1–16 (2001)
21. Stacey, R.D.: Strategic Management and Organisational Dynamics: The challenge of complexity to ways of thinking about organisations. Pearson, London (2011)
22. Waldrop, M.M.: Complexity: the emerging science at the edge of order and chaos. J. Chem. Inf. Model. **53**(9), 1689–1699 (2013)
23. Gell-Mann, M.: Complexity and complex adaptive systems. In: The Evolution of Human Languages. (Santa Fe Institute Studies in the Sciences of Complexity. Proceedings vol. 10), pp. 3–18 (1992)
24. Holland, J.H.: Complex Adaptive Systems. In: Daedalus, vol. 121, no. 1 (2010)
25. Lichtenstein, B., Uhl-Bien, M., Marion, R., Seers, A., Orton, J.D., Schreiber, C.: Complexity leadership theory. ECO Emerg. Complex. Organ. **8**, 2–12 (2006)
26. Mitchell, M., Newman, M.: Complex systems theory and evolution. In: Encyclopedia of Evolution. Oxford University Press, Oxford (2002
27. Ackoff, R.L.: Towards a system of systems concepts. Manage. Sci. **17**(11), 661–671 (1971)
28. Argyris, C.: Double loop learning in organizations. Harv. Bus. Rev. **55**, 115–125 (1977)
29. Anon, Smith, M.: Chris Argyris: Theories of Action, Double-Loop Learning and Organizational Learning, no. Elkjaer 2000, pp. 1–16 (2000). http://www.Infed.Org/Thinkers/Argyris.Htm
30. Petrevska Nechkoska, R.: Tactical management Sense-and-Respond framework enhancement using ICT. Ghent University Belgium, Ghent (2017)
31. Engel, A., Browning, T.R., Reich, Y.: Designing products for adaptability: insights from four industrial cases. Decis. Sci. **48**(5), 875–917 (2017)
32. Andrzejak, A., Reinefeld, A., Schintke, F., Schütt, T.: On adaptability in grid systems BT. In: Getov, V., Laforenza, D., Reinefeld, A. (eds.) Future Generation Grids, pp. 29–46. Springer, Boston (2006). https://doi.org/10.1007/978-0-387-29445-2_2
33. Fricke, E., Schulz, A.P.: Design for changeability (DfC): principles to enable changes in systems throughout their lifecycle. Syst. Eng. **8**(4), 342–359 (2005)
34. The Open Group, Application Platform Service Qualities (2018)
35. Radhakrishnan, R.: Non-Functional Requirements (NFR) Framework: A Subset of the Enterprise Architecture Framework (2009)
36. Oluwaseyi, O.: Developing business capabilities with COBIT 5. In: ISACA (2017)
37. Castro, J., Kolp, M., Mylopoulos, J.: Towards requirements-driven information systems engineering: the Tropos project. Inf. Syst. **27**(6), 365–389 (2002)
38. Andresen, K., Gronau, N.: An approach to increase adaptability in ERP systems. In: Managing Modern Organizations with IT, pp. 15–18 (2005)
39. Andresen, K., Gronau, N.: Managing change–determining the adaptability of information systems. In: European and Mediterranean Conference on Information Systems 2006, pp. 1–9 (2006)
40. Gronau, N., Rohloff, M.: Managing change: Business/IT alignment and adaptability of information systems. In: ECIS, no. 2007, pp. 1741–1753 (2007)
41. International Standardization Organization, ISO/IEC 25010:2011(en): Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models (2017)
42. Santos, E., Pimentel, J., Dermeval, D., Castro, J., Pastor, O.: Using NFR and context to deal with adaptability in business process models. In: Proceedings of 2011 2nd International Workshops on Requirements RE@RunTime, no. May 2014, pp. 43–50 (2011)
43. Jokste, L., Grabis, J.: Rule based adaptation : literature review. In: Proceedings of 11th International Science Practice Conference, vol. 2, pp. 42–46 (2017)
44. Kalareh, M.A.: Evolving Software Systems for Self-Adaptation (2012)

45. Opdyke, W.F.: Refactoring Object-oriented Frameworks. University of Illinois at Urbana-Champaign, Champaign, IL, USA (1992)
46. J. Kerievsky, "Refactoring to Patterns (Google eBook)," p. 400, 2004
47. Mens, T., Tourwé, T.: A survey of software refactoring. IEEE Trans. Softw. Eng. **30**, 126–139 (2004)
48. Tokuda, L., Batory, D.: Evolving object-oriented designs with refactorings. Autom. Softw. Eng. **8**(1), 89–120 (2001)
49. Mackrell, D., Mcdonald, C.: Evaluation into Action Design Research (2014)
50. Schilling, R.D., Beese, J., Haki, K.M., Aier, S., Winter, R.: Revisiting the impact of information systems architecture complexity: a complex adaptive systems perspective. In: Thirty Eighth International Conference on Information System, no. December (2017)
51. Benbya, H., McKelvey, B.: Toward a complexity theory of information systems development. Inf. Technol. People **19**, 12–34 (2006)
52. Ardagna, D., Comuzzi, M., Mussi, E., Pernici, B., Plebani, P.: PAWS: a framework for executing adaptive web-service processes. IEEE Softw. **24**(6), 39–46 (2007)
53. Onix, M.F.A., Fielt, E., Gable, G.G.: Complex adaptive systems theory in information systems research - a systematic literature review. In: Twenty-Fourth European Conference on Information System, pp. 1–18 (2016)
54. Shevtsov, S., Berekmeri, M., Weyns, D., Maggio, M.: Control-theoretical software adaptation: a systematic literature review. IEEE Trans. Softw. Eng. **43**, 1–28 (2017). https://ieeexplore.ieee.org/document/7929422/
55. Cechich, A., Piattini, M.: Managing COTS components using a six sigma-based process. In: LN Product Focused Software Process Improvement 5th International Conference PROFES Proceedings 2004, Japan, 5–8 April 2004 (2004)
56. Cechich, A., Piattini, M.: Quantifying COTS component functional adaptation. In: Bosch, J., Krueger, C. (eds.) ICSR 2004. LNCS, vol. 3107, pp. 195–204. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27799-6_16
57. Brogi, A., Cámara, J., Canal, C., Cubo, J., Pimentel, E.: Dynamic contextual adaptation. Electron. Notes Theor. Comput. Sci. **175**(2), 81–95 (2007)
58. Haeckel, S.H.: Adaptive Enterprise: Creating and Leading Sense-And-Respond Organizations, vol. 33, no. 2. Harvard Business School Press, Boston (1999)
59. Stirna, J., Grabis, J., Henkel, M., Zdravkovic, J.: Capability driven development – an approach to support evolving organizations. In: Sandkuhl, K., Seigerroth, U., Stirna, J. (eds.) PoEM 2012. LNBIP, vol. 134, pp. 117–131. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34549-4_9
60. Snowden, D.: Cynefin: A Sense of Time and Place, pp. 1–35 (1999)
61. Henkel, M., Stratigaki, C., Stirna, J., Loucopoulos, P., Zorgios, Y., Migiakis, A.: Extending capabilities with context awareness. In: Krogstie, J., Mouratidis, H., Su, J. (eds.) CAiSE 2016. LNBIP, vol. 249, pp. 40–51. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39564-7_4
62. Kampars, J., Stirna, J.: A repository for pattern governance supporting capability driven development. In: Business Informatics Research (2017)
63. Andrews, M., Pritchett, L., Woolcock, M.: Building State Capability - Evidence, Analysis, Action. Oxford University Press, Oxford (2017)
64. Ackoff, R.L., Gharajedaghi, J.: On the mismatch between systems and their models. Syst. Res. **13**(1), 13–23 (1996)
65. Walker, B., Holling, C.S., Carpenter, S.R., Kinzig, A.: Resilience, adaptability and transformability in social – ecological systems. Ecol. Soc. **9** (2004). https://www.ecologyandsociety.org/vol9/iss2/art5/manuscript.html

66. Alter, S., Sherer, S.A.: A general but readily adaptable model of information system risk. Commun. ACM **14**, 1–28 (2004)
67. Liu, J., Xue, C., Dong, L.: The Adaptability Evaluation of Enterprise Information Systems BT (2011)
68. Danesh, M.H., Yu, E.: Modeling enterprise capabilities with i*: reasoning on alternatives. In: Iliadis, L., Papazoglou, M., Pohl, K. (eds.) CAiSE 2014. LNBIP, vol. 178, pp. 112–123. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07869-4_10
69. Rosemann, M., Recker, J., Flender, C.: Contextualisation of business processes. Int. J. Bus. Process Integr. Manag. **3**(1), 47 (2008)