Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 159 (2019) 878–883

Procedia Computer Science

www.elsevier.com/locate/procedia

23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Introducing intents to the OODA-loop

Johan Silvander[a*], Lars Angelin[b]

[a]*Software Engineering Research Lab Sweden, Blekinge Institute of Technology, Karlskrona, Sweden*
[b]*Ericsson AB, Sweden*

Abstract

Together with Ericsson AB, we are using the design science framework when investigating how to create an intent-driven system for their business support system and its business studio. The aim is to present our initial results on how an extended OODA-loop can be used to realize a robust, but still flexible, software architecture for an intent-driven system. We explain how an extended OODA-loop is constructed and provide suggestions of how different part of it can be implemented. The initial results are promising but further research is needed to use the extended OODA-loop as reusable components in intent-driven systems. Our next step is to extend the generic methods with knowledge representation and reasoning capabilities.

Keywords: Intent-driven system; Extended OODA-loop; Business Support System

1. Introduction

An enterprise can be seen as a hierarchical line-of-command structure in which business rules at the top steer, align, and control the activities and behaviors further down in the structure. Business rules also steer all behavior in a business support system (BSS), in pursuit of the goal of creating and maintaining a successful business.

The difference between a business rule and a business intent is a small shift of perspective. A business rule is static and states what to do in a given situation, while a business intent states the desired or optimal outcome of a given situation. An intent can range in complexity from an atom intent to an algorithm of intents that combines a set of sub-intents. An atom intent on one level, is likely to fan out into several intents on a lower level. During the interaction about an intent the actors have to prove their understanding of the intent in order to gain a common understanding and knowledge about the intent. This interaction can evolve over several steps and might re-shape the original intent. This requires an architecture which can support context awareness in a compositional way. We define this type of systems as intent-driven systems.

E-mail address: johan.silvander@bth.se

1877-0509
10.1016/j.procs.2019.09.247

Intents must be formulated for both human and machine consumption to facilitate automation in a BSS. The level of automation, of decision and action selection, depends on the involved components capabilities and rights of taking decisions and performing actions.

The architecture must support an intent-driven system to optimize itself by continuously compare, and evaluate both business outcomes and its own capabilities. The results may require continuous adaptions of a business intent's definition and execution. This requires components with a high, but adaptable, level of automation. To our knowledge no such BSS exists in the industry today. Despite industrial relevance, little research has been conducted in the area of intent-driven systems and how software companies can be supported in building and managing these systems.

The aim with this paper is to introduce how an extended observe, orient, decide, and act loop (OODA-loop) can be used to realize a robust, but still flexible, software architecture for an intent-driven system.

The remainder of this article is organised as follows. The background and related works are presented in Section 1.1 and the research design and the research methods are presented in Section 2. The results are presented in Section 3 and an analysis thereof are presented in Section 4. The results are further discussed in Section 5 and conclusions and future work are presented in Section 6.

1.1. Background and Related Work

In our previous work [21] BSS and its business studio was our focus. We introduced the ideas of intent-driven systems, and introduced Pask's conversation theory [15] as a model to describe intent-driven systems. We defined a *context frame* as the total domain information for a specific domain an actor has obtained.

We have chosen to describe the architecture of an intent-driven system as a compositional systems of context frames. A context frame is described as an extended version of an OODA-loop. The OODA-loop was initially developed by John Boyd in the 1970s, as an in-combat decision tool of the U.S. Air Force [17]. Simplified versions of the OODA-loop are found in today's software systems.

In the literature we find guidance about the construction of intent-driven systems. Ditzel et al. [7] describes modelling principles for designing distributed adaptive observation systems and DiMario et al. introduce the concept of System of Systems collaborative formation mechanism [6]. How a Knowledge-Based Engineering system can be configured to provide automated design support for a compositional system is presented by Tanik et al. [22]. How to model capabilities and characteristics of a component is discussed by Derler et al. [5]. The challenges regarding information fusion are elaborated by Blasch et al. [2] while Seth et al. describes how computing for human experience and physical-cyber-social computing can be performed [19].

Vital contributions for an intent-driven system regarding the OODA-loop exists in the literature. McCauley-Bell and Freeman presents a methodology which proposes the analysis of evidence accrual by categorizing responses in the OODA-loop as a result of knowledge systems and belief systems [13]. A Bayesian network framework for situation assessment which can be seen as part of the observe and orientate components of the OODA-loop is presented by Bladon et al. [1]. Fusano et al. presents the OO-part of the OODA-loop from a group perspective, with the attention to the interaction between each of the components. [10]. Sholes presents how to identify the metrics and taxonomies in the field of autonomy along the four dimensions of the OODA-loop [20]. Consoli et al. [4], present how the relationship between the Belief-Desire-Intention framework and the OODA-loop can be enabled.

The intent-driven system shall be used to realize a BSS and its business studio. The study is performed during the time Ericsson AB was investigating and elaborating the requirements needed to support the planning and monitoring of business intents in their next-generation BSS and its business studio.

2. Methodology

Together with Ericsson AB, we are using the design science framework [11] as our research framework. During our design science study we have used the following research methods: systematic literature review, case study, quasi experiment, and action research.

3. Result

One of the results from our design science study is an extended version of the OODA-loop. We regard the extended OODA-loop as a context frame. The extended OODA-loop is shown in Figure 1.

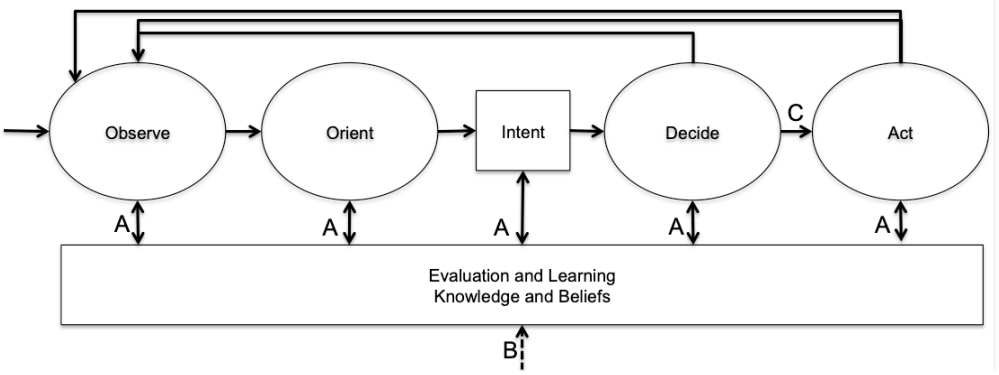

Fig. 1. The extended OODA-loop.

The original OODA-loop consists of four phases. First, observe or detect changes, events, stimuli or other things that happen in the context of interest, including internal states. Observations can be single or unfolding events, and they can be simple or complex in structure. The orient phase consists of aggregating, and analyzing the observations that form the basis for the decisions. The decide phase is about what one wants to achieve and the act phase is how to achieve it. The decisions can be limited by the inventory of available possible actions. The action can be of atomic or composite nature and might be implemented as a business process in a BSS.

In Figure 1, the visible extensions are the intent box, the box containing evaluation and learning together with knowledge and beliefs, and the arrows marked with A or B. We have extended the phase with an intent phase. Each of the phases have its own logic. This logic needs to be provided with knowledge and beliefs. The logic itself can be changed, based on evaluation and learning.

Intents are not explicitly mentioned in the original OODA-loop. We make the intents explicit in order to, in a given situation, clearly separate what to do from desired outcome. This adds flexibility to how to adapt to a certain situation, give certain capabilities.

In Figure 1, the box containing evaluation and learning, and knowledge and beliefs, are implicit in the original OODA-loop. We make this explicit for two main reasons, i.e. a logic place for the use of artificial intelligence (AI), and the possibility to bootstrap a solution.

The box containing evaluation and learning, and knowledge and beliefs, can have various types of learning capabilities, ranging from deterministic to AI. This makes it possible to evaluate the effectiveness and efficiency, and learn how to improve the behaviour of the OODA-loop itself.

The use of AI must support the human actors in their knowledge creation. Thus, it is critical that the people working in AI-enabled processes are able to understand the reasoning behind AI-generated results. All of the steps in our extended OODA-loop can be understood and executed by both humans and machines, which makes it possible to work together in the most efficient way possible based on the particular circumstances of the organization.

The arrows marked with A in Figure 1 serve three different purposes. The first purpose is to provide the logic in a phase with the needed knowledge, and beliefs. The knowledge and beliefs can be provided as part of the configuration of the logic or at execution time. The second purpose is to feedback information about the execution of the logic, to be evaluated and used for learning. The last purpose is to change the logic itself, based on what is learnt.

In a BSS it must be possible to utilize the knowledge and beliefs which exist in an organization. This knowledge and beliefs will be used during the execution of the OODA-loops. The arrow marked with a B in Figure 1 have two different purposes. First, it shall expose the knowledge and beliefs in an enterprise as an information graph. Secondly, it shall make it possible to retrieve the desired knowledge and beliefs without any knowledge about implementation of the serving interfaces.

It must be possible for an enterprise to bootstrap the different context frames used in its BSS. The bootstrapping concerns the knowledge and beliefs, as well as the logic used in the different phases, and in the evaluation and learning. This introduces two different logical layers named define layer and execute layer. We regard the define layers, and execution layers as OODA-loops, seen from different viewpoints. In practice, the different layers are all executed as different context frames.

The extended OODA-loop introduce an important aspect on the arrow marked with C in Figure 1. We have extended the OODA-loop by allowing different actors be responsible for the decision and the action. Normally the OODA-loop can be seen as an execution layer. When the actor responsible for the decision makes a stated intent about what actions are needed, the OODA-loop is used in the define layer. The actor responsible for the action base its promises on its own intents. When an agreement about the actions are concluded the OODA-loop will be executed based on a composition of intents. This extension creates the possibilities for a recursive nature of the OODA-loop, which makes it possible to build a compositional systems of intents.

The ability to use compositions makes it possible to use the same OODA-loop engine to observe, decide and select the proper actions for all event types, regardless of complexity. It can also be used recursively to build layered structures with arbitrary depth, i.e. it can support multilayered business processes and interactions.

Below we describe how the different results from our studies supports the idea of an extended OODA-loop as a compositional building block of an intent-driven system.

4. Analysis

Since the intent-decision-action chain is defined by business people, these artefacts are often stated in a non-executable form. We have investigated a solution to support visual and logical validation of decisions and to generate these decisions as executable rules. The creation of the executable decision was done via a machine learning (ML) pipe-line. We extended the decision tree algorithm ID3 [16], and visualized the result in the form of a decision tree. The tree format was used to generate code for what-if scenarios, and the resulting executable rules for the decisions. The tree contains the intent as a root node and the actions as leaf nodes.

We investigated how an intent of an actor can be matched against available capabilities. We used fuzzy sets [23] to realize a component selection logic. The component selection logic was used to select the components which best matched an actors intent.

The assessment of intents might be done by external actors. Such assessment requires the possibility to learn which decisions to take in a given context. In order to deliver a value according to the assessments, business process optimization might be used. A reinforcement learning algorithm used in deep neural networks called Deep Q-learning [14] was used to optimize the flow of decisions taken in business processes.

In order to investigate the correspondence between the define layer and the execute layer of an OODA-loop, we have investigated how to to perform validation, and visualization, of compliance to stated decisions. The actual actions were compared to the stated decisions by a ML-algorithm. We extended a ML-algorithm to be able to present the actual decisions taken in the form of fuzzy sets [23]. The ML-algorithm is based on the agglomerative merging algorithm described by Flach [9], and combined with the use of Shannon entropy [18]. The fuzzy set only contained the features from the stated decisions. We used the fuzzy set to validate the compliance to the stated decisions.

We are currently investigating how responses from interfaces can be exposed as a common information graph. An information graph will make it easier to provide the evaluation and learning, knowledge and belief layer with information from several systems without the need to understand the dependencies between these systems.

5. Discussion

The OODA-loop is from the beginning aimed for humans. The orient phase is the only phase which has been expanded by Boyd. It includes cultural traditions, genetic heritage, previous experience, new information, and analyses & synthesis [17]. The orient phase influences how the other phases are carried out [12]. Some of the information in the orient phase are used with no further reflections about if the information is valid in a certain context, e.g. cultural traditions [12].

When using the OODA-loop with software components, the orient phase might be regarded as a big ball of mud [8]. By making the evaluation, learning, knowledge, and beliefs explicit, we create a separation-of-concern of what is influencing the different phases. The logic used in the different phases will still be specific to each phase. The evaluation and learning, as well as the knowledge and beliefs, can be shared by phases or specific to a phase.

In the original OODA-loop the orient phase contains implicit and explicit knowledge gathered during a humans life time. It is not feasible for a software component to learn everything through its interactions with the environment. Instead we use of a define layer and an execute layer which makes it possible to bootstrap a software component. The arrow marked with a B, in Figure 1, is used to gather needed information from sources external to the software component, without using the components primary interaction interfaces. Since the original OODA-loop was not designed for collaborations [4], the arrow marked with a B can be used to retrieve information needed for collaborations between different software components.

Our idea of extending the OODA-loop with an intent phase is supported by a paper by Consoli et al. [4], who present how the relationship between the Belief-Desire-Intention framework and the OODA-loop can be enabled.

Our work regarding rule generation, correspondence checking of the define layer and the execute layer, capability matching, adaptive learning capabilities, and the use of information graphs, can be regarded as generic methods for the different phases in the extended OODA-loop. However, the software architecture of an intent-driven system has to be further investigated. We need to extend the generic methods with knowledge representation and reasoning capabilities [3]. This need is also valid for the evaluation and learning capabilities.

6. Conclusion and Future Work

We argue that the extended OODA-loop is a vital building block to an intent-driven system. However, we will continue to investigate how to create a software architecture with reusable components for intent-driven systems by addressing the issues discussed in Section 5. Our primary focus is to extend the generic methods with knowledge representation and reasoning capabilities.

References

[1] Bladon, P., Hall, R., Wright, W., 2002. Situation assessment using graphical models. Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997) 2, 886–893. doi:10.1109/ICIF.2002.1020903.

[2] Blasch, E., Kessler, O., Morrison, J., Tangney, J., 2012. Information Fusion Management and Enterprise Processing, in: Proceedings IEEE National Aerospace and Electronics Conf. (NAECON), 2012.

[3] Brachman, R.J., Levesque, H.J., 2004. Knowledge Representation and Reasoning. 1st ed., Elsevier.

[4] Consoli, A., Tweedale, J., Jain, L., 2008. Aligning cognitive models using AC3M. Proceedings - 1st International Conference on Emerging Trends in Engineering and Technology, ICETET 2008 , 882–886doi:10.1109/ICETET.2008.249.

[5] Derler, P., Lee, E., Vincentelli, A.S., 2012. Modeling Cyber Physical Systems, in: Proceedings of IEEE, pp. 13–28.

[6] DiMario, M., Boardman, J., Sauser, B., 2009. System of Systems Collaborative Formation. IEEE Systems Journal 3, 360–368. URL: http://ieeexplore.ieee.org/document/5229218/, doi:10.1109/JSYST.2009.2029661.

[7] Ditzel, M., Kester, L., Broek, S.V.D., 2011. System design for distributed adaptive observation systems. 14th International Conference on Information Fusion , 1–8.

[8] Evans, E., 2015. Domain-Driven Desig Reference. https://domainlanguage.com/wp-content/uploads/2016/05/DDD_Reference_2015-03.pdf Last checked 2019-03-13. URL: https://domainlanguage.com/wp-content/uploads/2016/05/DDD{_}Reference{_}2015-03.pdf.

[9] Flach, P., 2012. Machine Learning. Cambridge University Press.

[10] Fusano, A., Sato, H., Namatame, A., 2011. Study of multi-agent based combat simulation for grouped OODA Loop. SICE Annual Conference 2011 , 131–136.

[11] Hevner, A.R., March, S.T., Park, J., Ram, S., 2004. Design Science in the Information Systems Research. MSI Quarterly 28, 75–105.

[12] Maccuish, D.A., 2012. ORIENTATION : KEY TO THE OODA LOOP THE CULTURE FACTOR. Journal of Defense Resource Management 3, 67–74.

[13] Mccauley-bell, P., Freeman, R., 1996. Quantification of Belief and Knowledge Systems in Information Warfare, in: Proceedings of the 5th IEEE international conference on Fuzzy Systems, 1996., pp. 1579–1585 (Volume:3).

[14] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518, 529. URL: https://doi.org/10.1038/nature14236http://10.0.4.14/nature14236https://www.nature.com/articles/nature14236{#}supplementary-information.

[15] Pask, G., 1976. Conversation Theory. Elsevier.
[16] Quinlan, J.R., 1986. Induction of Decision Trees. Machine Learning 1, 81–106. doi:10.1023/A:1022643204877.
[17] Richards, C., 2011. Boyd's OODA Loop, in: Proceedings of Lean Software and Systems Conference 2011, pp. 127–136. URL: https://sslserver.com/jvminc.com/boydsrealooda{_}loop.pdf.
[18] Shannon, C.E., 1948. A Mathematical Theory of Communication. Bell System Technical Journal 27, 379–423. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6773024, doi:10.1002/j.1538-7305.1948.tb01338.x.
[19] Sheth, A., Anantharam, P., Henson, C., 2013. Physical-cyber-social computing: An early 21st century approach. IEEE Intelligent Systems 28, 78–82. doi:10.1109/MIS.2013.20.
[20] Sholes, E., 2007. Evolution of a UAV autonomy classification taxonomy, in: IEEE Aerospace Conference Proceedings, pp. 1–16. doi:10.1109/AERO.2007.352738.
[21] Silvander, J., Wilson, M., Wnuk, K., Svahnberg, M., 2017. Supporting Continuous Changes to Business Intents. International Journal of Software Engineering and Knowledge Engineering 27, 1167–1198. URL: http://www.worldscientific.com/doi/abs/10.1142/S0218194017500449, doi:10.1142/S0218194017500449.
[22] Tanik, U.J., Begley, A., 2014. Applied Cyber-Physical Systems. Springer New York, New York, NY. URL: http://link.springer.com/10.1007/978-1-4614-7336-7{_}11http://link.springer.com/10.1007/978-1-4614-7336-7, doi:10.1007/978-1-4614-7336-7.
[23] Zimmerman, H.J., 2001. Fuzzy Set Theory - and its applications. fourth ed., Springer New York LLC.