



HAL
open science

A Meta-Model for integrating Human Factors into System Engineering

Kahina Amokrane, Andreas Makoto Hein

► **To cite this version:**

Kahina Amokrane, Andreas Makoto Hein. A Meta-Model for integrating Human Factors into System Engineering. Human System Integration Conference, Nov 2021, San Diego, United States. hal-03355804

HAL Id: hal-03355804

<https://hal.science/hal-03355804v1>

Submitted on 8 Jan 2025

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



A Meta-Model for integrating Human Factors into System Engineering

Kahina Amokrane-Ferka
IRT SystemX
8 Avenue de la Vauve, Palaiseau, France
+33 1 69 08 06 17
kahina.amokrane-ferka@irt-systemx.fr

Andreas Hein
IRT SystemX
8 Avenue de la Vauve, Palaiseau, France
+33 1 69 08 06 17
andreas.hein@irt-systemx.fr

Copyright © 202X by Kahina Amokrane-Ferka . Permission granted to INCOSE to publish and use.

Abstract. With the emergence of autonomous systems, it is no longer sufficient to only consider the interactions between humans and systems but to assign roles and allocate functions to both. Systems engineering methods, which aim for designing such systems need to take this paradigm shift into account. Domains such as human system integration aim to work towards this objective from a systems engineering perspective. However, one shortcoming of existing systems engineering approaches is that it do not explicitly take the human aspect into account. The objective of this paper is to integrate elements from ergonomics modeling languages into SysML/UML, a systems modeling language in widespread use. We conclude that integrating human factors into SysML/UML significantly extends the expressivity regarding ergonomic concepts for designing autonomous systems which collaborate with humans.

Introduction

Until now, systems have been designed to address user needs, which is reflected in common systems engineering standards and handbooks [Haskins, 2007] [Kapurch, 2010]. In these systems the human is considered as an external component or just an actor who interacts with the system through human-machine interfaces. Today, with the emergence of autonomous systems, needs have evolved towards human/system collaboration and cooperation to achieve a common goal. Therefore some of the human tasks can be performed by the autonomous system [Boy 2017] [Hoc 2000].

For a successful cooperation, on the one hand the human must understand the decisions of the system. On the other hand, the system must understand the human (their behavior and characteristics) and this system must be designed to reason or to behave like a human. For example, in an autonomous car, the car needs to understand how the human behaves during handing over command over the vehicle and the car needs to behave in such a way that the human can understand its next actions. To that end, the human and the system must have the same reference model describing the tasks to be performed [Boy 2017]. This reference model is known as the task model in ergonomics. It is based on ergonomist activity languages [Leplat 2004].

For successful human-system integration, human factors, especially the task model, must be considered during the whole life cycle of systems engineering [Madni 2011] [Boy 2014] [Handley 2008]. Therefore, considering the elements of ergonomic activity description languages in system design would form a solid basis for integrating human factors into the systems engineering process. In the

following, we opt for this approach of integrating ergonomic language elements into a systems engineering language instead of developing a common language for ergonomics and systems engineering. The main reason is that we can start with existing languages in both domains and can exploit lightweight extension mechanisms to these languages instead of developing a new language.

Until the last decade, on the one hand, humans were studied in ergonomics/human factors, a discipline that seeks to improve the usability of systems according to human capabilities and limitations. On the other hand, system design was studied in systems engineering [Boy 2014] [Millot 2018]. Then, a convergence of these two domains occurred through the field of Human System Integration (HSI). The main objective of HSI is to provide methods and tools that support the systems engineering community by ensuring that humans are taken into account in a logical and efficient way all along the system design process [Madni 2011] [Boy 2016].

In the HSI field, there have been several attempts to integrate human factors in early system design stages. On the one side, relevant works [Handley 2008] [Madni 2011] [Boy 2014] focused on high level questions, with the aim of capturing and identifying the HSI research problem (the what). However, they did not define the tools to achieve this goal, nor the methods to be used (the how). On the other side, some studies contributed to very specific frameworks such as MODAF [Baker 2006], DODAF [Bruseberg 2007] and NATO [Handley 2008]. Another minority of works, even if they show how to integrate human factors but their attachment to systems engineering processes and methodologies lead to a superficial consideration of ergonomic studies and references. Among these works we can quote the integration of human errors [Hardman 2009], the medialization of human tasks using the SysML activity diagram [Handley 2019] [Orellana 2018] [Hause 2017], or the use case diagram [Liaghati 2020]. Therefore, our challenge is to integrate human factors into the systems engineering process by considering them as studied in ergonomics, from the beginning of systems engineering process. And finalizing the modeling by a systems engineering language to better integrate them. Knowing that despite the different diagrams provided by SysML, none of them is particularly oriented to supporting the human factors.

Thereby, through this work, we propose a step towards properly integrating human factors into systems engineering. This approach contribute to process and language integration level. It begins by analyzing human activity descriptions languages from an ergonomic perspective. Then we extract their principal elements and relations. With these elements we build our meta-model. We use the UML/SysML profiling mechanism for developing a domain-specific language based on our meta-model. We subsequently apply the domain-specific language to a maritime defense case study.

Literature survey

Our literature survey covers task analysis and HSI with a focus on modeling languages. We survey the task analysis literature to extract key concepts from existing task modeling languages that are required for modeling human activities. The objective is to later transfer into a systems modeling language. In addition, we survey the HSI literature for identifying gaps in the use of systems modeling languages for integrating human factors into systems engineering

Task analysis

The design of any interactive system requires task analysis and modelling that the user must and/or can do, as well as all the objects in the world, to which these tasks relate, and the events that can occur. The consideration of human factors is now recognized as necessary from the early stages of the interactive systems engineering process. This consideration involves, in the vast majority of cases, an analysis, description and modeling of the user's task [Lucquiaud 2002].

The task model leads to a twofold interpretation depending on whether one adopts the system or the cognitive point of view. It can: represent the actual functioning of the system, i.e. the set of functions

that the system will have to provide; model the knowledge that the user will have in order to use the system; model the prescribed task (what an operator has to do from the experts' point of view) or the activity (what the operator does in the area) [Amokrane 2010].

There are several languages/ formalisms developed in ergonomics to describe such tasks and the corresponding activity by humans at work. Among them MAD [Scapin 1989] which is an object-oriented formalism where the task is an object. MAD is used to represent the hierarchical dimension of the planning of human activities. Hence, the decomposition of a task into sub-tasks is not just a simple logical decomposition, but it aims to reflect the structure of the expert mental representation of the task.

The Groupware Task Analysis (GTA) [Van Der Veer 1996] is going one step beyond by adding supplementary facets to the task description: agents (participants) who are the actors of a task with a given role within an organization. Agents may correspond to people, either individuals or groups, but may also refer to systems. Work: identified by a task, and structured as a set of sub-tasks and executed actions. Work Situation: analyzing a task context from the viewpoint of the situation means detecting and describing the environment (physical, conceptual, and social) and the objects in this environment.

CTT (ConcurTaskTrees) [Paternò 2004] is a hierarchical task decomposition. It allows designers to concentrate on the activities that users aim to perform. The particularity of CTT is that it introduces four types of tasks: user task, abstract tasks that can be decomposed into sub-tasks, System tasks and interaction tasks.

HAWAI-DL (Human Activity and Work Analysis for sImulation-Description Language) [Amokrane 2008] provides a hierarchical description of tasks, allowing activities to be represented based on the whole of the agent activity, and not only the required procedure. Comparing with other languages, HAWAI-DL allows to describe tasks that are a priori hierarchically independent (i.e. that have no hierarchical relations between them at the time it is described). It includes also conditions under which a task should be performed such as "Tolerated Conditions of Use" (TCU) and their associated risks, are explicit in the task description. TCU is a concept that derives from ergonomic research, and also from the field of human reliability.

Human System Integration (HSI)

HSI is "an interdisciplinary technical and management processes for integrating human considerations within and across all system elements; an essential enabler to systems engineering practice". [INCOSE 2020]

Embedding human factors into SE processes can be performed at the process, methods or tool/language level.

At process level, Chua and Feigh [Chua 2011] proposed an extension of the NASA systems engineering process by integrating human factors elements at each of the four steps. Another idea consists of standardizing terminology between systems engineering and human factors practices. Hardman, et al. [Hardman 2009] suggest clarifying the terminology used within DoD. Orellana [Orellana 2014] proposed an ontology to integrate the semantics of the two domains.

Efforts at the methods level aim at supporting the integration of human factors in systems engineering with a focus to improve one of the existing design methods or by proposing a new one based on the analysis of the existing in systems engineering. Hardman [Hardman 2012] proposes an empirical methodology for human integration in the SE technical processes. Bruseberg [Bruseberg 2007] and Baker [Baker 2006] proposed extensions to MoDAF and DoDAF Frameworks, respectively, by integrating new views related to human factors. To unify the proposals between DoDAF and MODAF; NATO (North Atlantic Treaty Organization) has been created [Handley 2008]. Thereafter, Hause

[Hause 2017] inspired by MoDAF and DoDAF human views points to integrate a set of Human views called Personnel Views in UAF).

The most thorough way to integrate human factors into the SE process is to address integration at the tool/language level. Works on this side are rare, we can cite [Boy 2014] who advocates the use of modeling and simulation to consider the integration of human factors in SE process. Bodenhamer [Bodenhamer 2017] has extended the system diagrams to include human as an internal system interface. Or again, the use of UML/SysML to model NATO human views [Handley 2019].

While process-level integration efforts support increased communication and the merging of ergonomics and systems engineering, they do not guarantee or even provide a mechanism to directly integrate human considerations into systems engineering products. At methods level, they are still inspired from those of systems engineering, so the consideration of human factors remains limited. Concerning languages, the existing attempts aim at integrating human factors based on SysML, whereas it is far from being the appropriate language, especially for the modeling of human activity. Concerning DODAF, MODAF and NATO frameworks, none of them has become an internationally recognized standard or has been officially integrated into the corresponding architectural framework

Human factors meta-model for systems engineering

We propose a meta-model for human factors for the context of systems engineering. The meta-model is a precondition for integrating human factors, more specifically tasks, into a systems modeling language. The meta-model provides a formal description of key concepts necessary for modeling tasks. For this purpose, we extract task-related concepts from ergonomics languages, notably HAWAI-DL. Finally, we introduce elements of the meta-model into a well-known systems modeling language, the Systems Modeling Language (SysML), in the form of a domain-specific language. To customize SysML, we use the profiling mechanism for developing a domain-specific language.

From human activities to the meta-model

To build our meta-model we used the main elements of the HAWAI-DL meta-model [Amokrane 2010], one of the human activity description languages. This language is very expressive in terms of activity description, inspired by several languages such as GTA, MAD, then enriched by new elements such as human errors and common performance deviations. The main concepts taken into account for our meta-model are task, agent, role, event, object and conditions [Amokrane 2008]. We then extended the meta-model with elements from other studied languages, such as the concept of action from GTA. Then, based on our objectives, in particular, representing human-system cooperation, we added new concepts such as the distinction between human agent and system agent, and we distinguished between collaborative, cooperative, and individual tasks Figure 1.

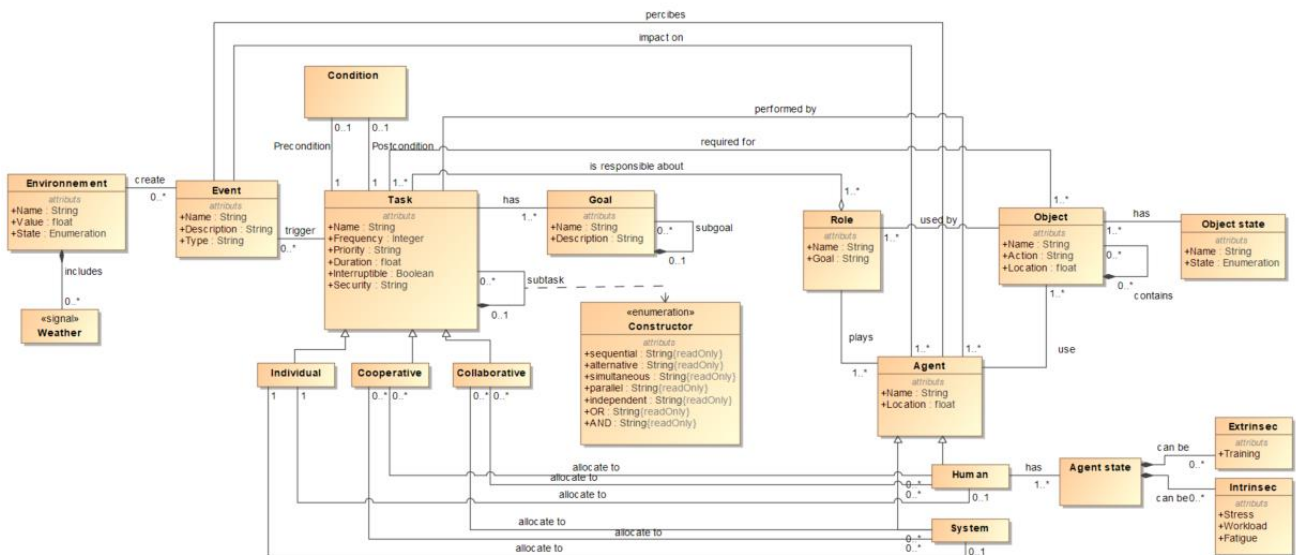


Figure 1: Human Task meta-model

The description of some principal elements are below:

Task: a task is performed or must be performed by an agent to achieve a certain goal. A task can correspond to prescribed one (what an agent must do) or to effective task (activity: what the agent perform effectively). A task usually changes something in the world. It can be complex i.e. decomposable into subtasks, or simple (action) i.e elementary decomposition. An action is a physical or cognitive process that can be accomplished by an agent [Amokrane 2010] [Van Der Veer 1996]. A complex task reflects the organization of the task according to the hierarchical organization, and temporal, logical organization through constructors. For the purpose of task allocation, the following sub-classes are added: Cooperative task which is a mutual engagement of participants to solve the problem together [Roschelle 1995]; collaborative task which is accomplished by the division of work among participants, each person is responsible for one part of the problem solving; and individual task, executed by one agent (human or system).

Constructors: a constructor is an important element of a task model. The constructor describe the logical and temporal decomposition of a complex task into subtasks (see description of each one in Table 1 and 2

Agent: an agent is an active entity who realize a task. It can play one or more roles and perform one or more tasks [Amokrane 2008]. Human-machine interaction moves into human-machine cooperation when the machine becomes highly automated. In this case, it is more appropriate to talk about agent-agent cooperation [Boy 2017]. Thus, we distinguished between two subclasses “human agent” and “system agent”.

Role: a role is a significant collection (set) of tasks performed by one or more agents. A role is therefore responsible of tasks that surround it [Amokrane 2008].

Goal: a goal indicates the reason of the task. Like the task, a goal is decomposable into sub goals up to low level goals [Van Der Veer 1996].

Object: it is used to model diverse objects. An object can be concrete or abstract having an independent existence, i.e. it can be manipulated without the need to know other objects [Scapin 1989].

Event: An event is a change in the state of the world at a given moment. The change can be internally generated by an internal concept such as object, task, agent, or can reflect changes in external concepts such as a change of the weather. An event can trigger one or several tasks, and can affect one or several agent’s characteristics [Van Der Veer 1996].

Environment: the environment is the universe in which the agent evolves. It is built up by all the objects of the world, agents, etc.

Condition: the condition indicates the context in which a task can or must be performed or not, and the expected state of the objects and/or the result produced after the completion of the task. A task can be triggered by an event that occurs in the environment (trigger condition) [Amokrane 2008].

SysML Domain-Specific Language

The objective of our meta-model is to facilitate the transition from an ergonomic language to a system language. For our study, we chose SysML/UML, a systems modeling language in widespread use [Friedenthal 2015]. We introduce some subtle but important stereotypes Figure 2. In the following, we describe the principal stereotypes which we added.

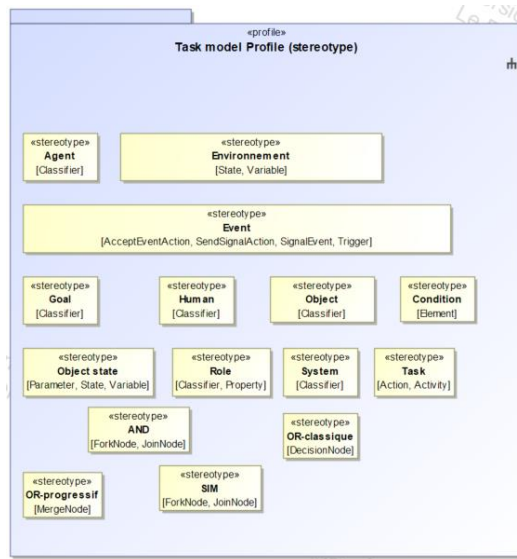


Figure 2: Task model concepts with different stereotype

Task stereotype: from SysML, it can inherit from the concepts of activity because each one of task and activity are used to describe the behaviour of an agent or a block. And also from the concept of action which is the elementary behavior of a block in SysML or agent in human factors.

Constructor stereotype: To describe temporal and logical relations between subtasks of a complex task, we use existing activity diagram concepts for three of them. For the sequential (serial) (SEQ) we use the logical organization of activity diagram as shown in Table 1. The Alternative (ALT) constructor can be modeled using the decision node. Then, Parallel (PAR) can be modeled using fork and join elements (see .

Table 1). However, the SysLM models are far from to be adequate to model human activity, among other some constructors. To overcome this lack constructors stereotypes are modelled because there equivalents in SysML, do not exist as shown in Table 2.

Table 1: HAWAI-DL constructors with their equivalent example in SysML



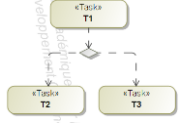
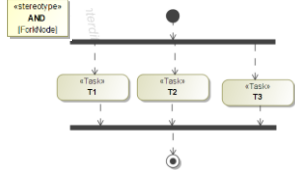
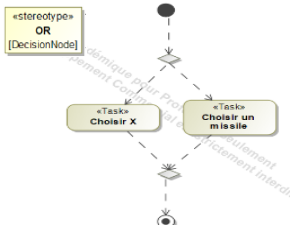
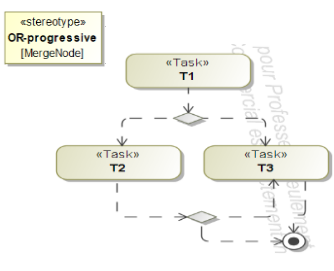
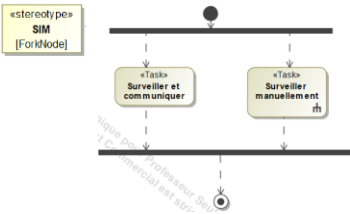
Constructor	SysML modelling	Description
Sequential (SEQ)		The subtasks must be performed one after another in right order
Parallel (PAR)		Subtasks of the same complex task must be performed at the same time without time constraints
Alternative (ALT)		One of the subtasks of a complex tasks chosen according to certain conditions.

Table 2: HAWAI-DL constructors with their equivalent stereotypes in SysML

Constructor	SysML modelling	Description
Independent (AND) « AND » stereotype		All subtasks must be performed in any order
Classic OR « Classic OR » Stereotype		One of the subtask can be chosen without any condition
Progressive OR « Progressive OR » stereotype		One of subtask must be done, if it is not successful (the expected goal is not reached), a second subtask must be done, and so on until the goal is reached
Simultaneous (SIM) « SIM » Stereo- type		Subtasks must be performed simultaneously (start at the same time and finish at the same time).

Agent stereotype: can inherit from “Classifier” which allow to distinguish others sub-classes, in our case Human and System.

Human agent stereotype: is used to model a human agent. It has several extrinsic (hierarchical status, group dynamics, communication, etc.) and intrinsic (stress, workload, fatigue, health, competence, knowledge, etc.) characteristics. “Human agent” stereotype is used to model the concept of human agent. This stereotype can inherit from the Actor concept with a significate evolution.

System agent stereotype: is used to model each software or hardware entity which perform tasks.

Role stereotype: can inherit from “Classifier” that allow to distinguish several roles.

Object stereotype: can inherit from « ObjectFlow » which used to specify data flow in input or output of an action

Object State stereotype: is used to model the different states of an object for example the state “Open” and “close” of the object door. Compared to SysML concepts, the “Object State” stereotype class can inherit from Parameter which feed into the subject state. State has the same definition in SysML and our approach. And it correspond also to Variable meta-classes. Compared to SysML concepts, the Event stereotype can inherit from : “event” class that has the same signification with “event” in human activity languages, namely specification of some occurrence that may potentially trigger effects by an object; “AcceptEventAction” which allows to execute an action flow when an event is occurred the same principal like triggering an action by an event in the task model; “Send-SignalAction” which is a concept that allow to transmit the information of event occurrence to run the action; “Signal” in block diagram and activity diagram represent the event defined in task model itself. It can inherit from the “Trigger” concept of state machine which defines the types of events that can initiate a transition between states.

Environment stereotype: can inherit from the block definition diagram and state machine diagram to model all the changing states in the world.

Condition stereotype: can inherit from “AcceptEventAction” which represents the required conditions for the completion of a task and allows from “SignalEvent” which represents the post and pre conditions of two successive actions respectively.

Validation of the domain specific language

Our case study is within the domain of naval defense. Our case focuses on an anchored frigate whose main mission is reconnaissance and protection of the open sea. In this context, if a suspicious object is detected it must be identified. If it is identified as an enemy or attacks, our system must attack back or respond accordingly Figure 3. This mission involves 5 main roles: supervisor, identifier, decision-maker, commander and effector. Some of the tasks related to these roles can be performed by an autonomous system such as supervision or identification.

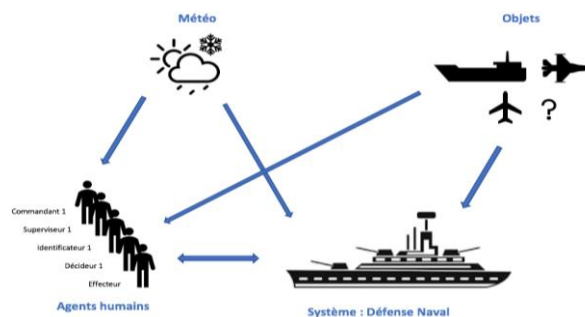


Figure 3: Illustration of our use case

To validate our SysML domain specific language we model the task (what must be done) related to naval defence using HAWAI-DL then we have succeeded to model this task in SysML thanks to the proposed stereotypes.

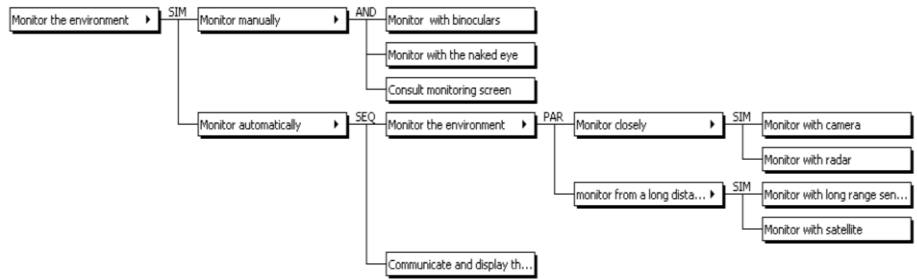


Figure 4: The task "Monitor the environment" in HAWAI-DL

Figure 4 shows the “Monitor the environment” task which is a complex task that must be done simultaneously by a system Agent and Human agent. Each agent must do several tasks in a different manner to achieve its objective. For example the human agent must perform three different tasks in any order (AND constructor). And the system Agent must perform two tasks in sequential (SEQ) and the first one can be achieved by doing two subtasks in parallel (PAR). The (Figure 5) show the modeling of the same task in SysML based on an activity diagram and the different constructor Stereotypes that we add. We successfully allocate the different subtasks to the corresponding agents (Human: supervisor 1 and system: supervisor system).

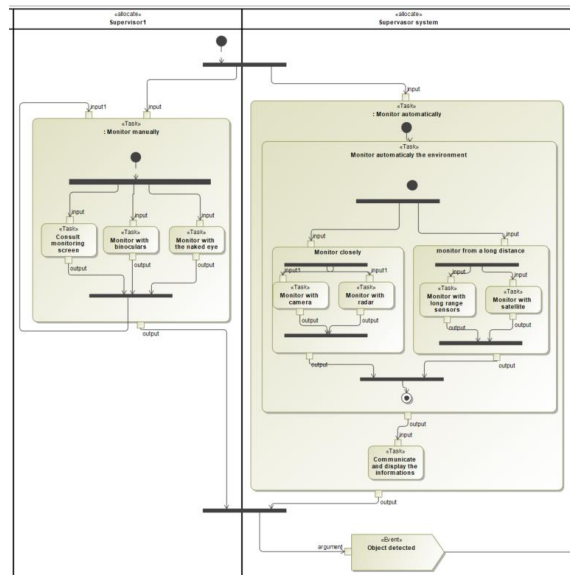


Figure 5: The task "Monitor the environment" in SysML

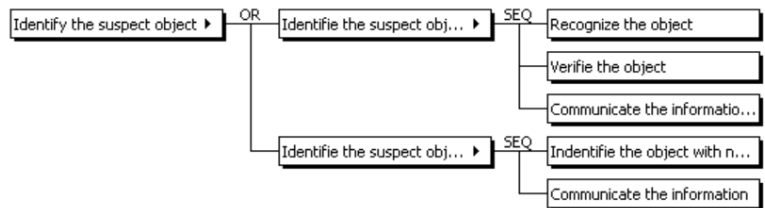


Figure 6: The task "Identify the suspect object" in HAWAI-DL

Figure 6 shows an example of the omplexe task ”Identify the suspect object” that can be performed in two ways (subtasks) automatically by the system OR manually by the human. To acheive one of the two subtasks, three or two subsubtasks, respectively must be performed in sequence. The medeling of this task in SysML is done (Figure 7) with classical modeling of an activity model in

addition to the allocation of each subtask to the relevant agent (System : Identification system ; or human : Identifier1).

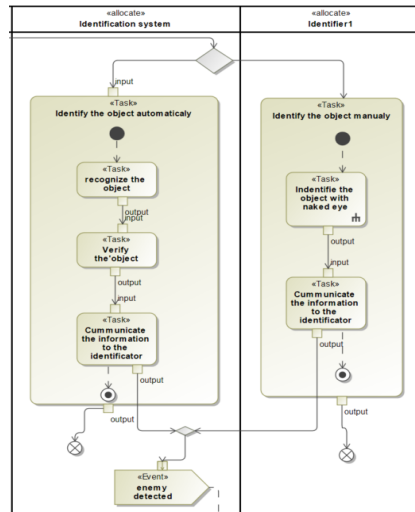


Figure 7: The task "Identify the suspect object" modeled in SysML

The application to our use case allowed us to verify the usefulness of our meta-model. We note that our meta-model covers successfully the "constructor" part (the temporal and logical relationships between the subtasks of a complex task). By contrast, the triggering event/task triggered part remains to be validated. In addition, the allocation of collaborative or cooperative tasks still remains to be done. In the current version, allocation is feasible at a sufficient level of decomposition that allows allocation to a single agent (subsystem). However, alternative allocation mechanisms that allow for a dynamic allocation to multiple agents might be possible, using a different formalism. This would allow for the allocation of collaborative or cooperative tasks.

Conclusion and perspectives

The human task analysis being an important step in the interactive and collaborative systems design, then its consideration in the process of systems engineering allows a full integration of human factors. To do so, we propose a domain-specific language, an extension to the SysML language by the main concepts of human task and the factors that result from it, early in the systems engineering process.

Our approach enabled us to gradually move from an ergonomic language to a systems engineering language. We started by extracting the main concepts of ergonomic languages, and then built our meta-model with these concepts and the relationships between them. This meta-model gives a global vision of the system and a detailed level of abstraction. Then, we transposed these concepts to concepts in SysML/ UML profiling mechanisms and proposed a domain-specific language, using the profiling mechanism.

The domain-specific language proposed allows system architects to understand the human task by themselves, then to analyze and model it with the involvement of an ergonomist without the problem of terminology, nor of understanding between the two domains.

For future work, we plan to extend the integration of the notion of "condition", one of the main concepts of ergonomic languages. Also in the current model we are unable to represent cooperative tasks, due to the limitation of SysML to allocate a task to more than one agent. Also, a demonstration of the language's usefulness in a wide range of systems engineering projects would strengthen its level of validation.

References

- Amokrane, K & Lourdeaux, D & Burkhardt, JM 2008, 'Learner behaviour tracking in a virtual environment, Proceedings of virtual reality international conference VRIC2008.
- Amokrane, K 2010, 'Suivi de l'apprenant en environnement virtuel pour la formation à la prévention des risques sur des sites Seveso', UTC Doctoral dissertation.
- Baker, K & Pogue, C & Pagotto, J & Greenley, M 2006, 'Human Views: Addressing the human element of capability-based decisions', TTCP HIS Symposium, DSTO, Australia.
- Bodenhamer, A 2012, 'Adaptations in the US Army MANPRINT process to utilize HSI-inclusive system architectures', *Procedia Computer Science*. vol. 8, pp. 249–254.
- Boy, GA 2017, *The handbook of human-machine interaction: a human-centered design approach*, CRC Press.
- Boy, GA & Narkevicius, JM 2014, 'Unifying human centered design and systems engineering for human systems integration', *Complex Systems Design & Management*, Springer, Cham, pp. 151–162.
- Bruseberg, A & Lintern, G 2007, 'Human factors integration for MoDAF: Needs and solution approaches', *Proc of INCOSE*.
- Chua, ZK & Feigh, KM 2011, 'Integrating human factors principles into systems engineering', *IEEE/AIAA 30th Digital Avionics Systems Conference*, pp. 6A1–1.
- Friedenthal, S & Moore, A & Steine, RA 2015, 'Practical Guide to SysML, A Practical Guide to SysML', Elsevier.
- Handley, HA & Smillie, RJ 2008, 'Architecture framework human view: The NATO approach', *Systems Engineering*, vol. 11 No 2, pp. 156–164.
- Handley, HA 2019, 'The Human Viewpoint for System Architectures', vol. 35. Springer, ISBN 978-3-030-11629-3.
- Hardman, N & Colombi, J & Jacques, D & Hill, R & Miller, J 2009, 'The challenges of human considerations in the systems engineering technical processes', *Proc. 7th Ann. Conf. Syst. Eng*, pp. 1–8
- Hardman, N & Colombi, J 2012, 'An empirical methodology for human integration in the SE technical processes', *Systems Engineering*, vol. 15 no. 2, pp. 172–190.
- Haskins, C & Forsberg, K & Krueger, M 2007, *INCOSE Systems Engineering Handbook*, International Council On Systems Engineering INCOSE.
- Hoc, JM 2000, 'From human–machine interaction to human–machine cooperation', *Ergonomics*, vol. 43 no. 7, pp. 833–843, <https://doi.org/10.1080/001401300409044>.
- INCOSE 2020, *Systems engineering vision*, Systems Engineering Vision Working Group of the International Council on Systems Engineerings.
- Kapurch, S 2010, *NASA Systems Engineering Handbook*.
- Leplat, J 2004, 'L'analyse psychologique du travail', *European review of applied psychology*, vol 54, no 2, pp. 101-108, <https://doi.org/10.1016/j.erap.2003.12.006>.
- Liaghati, C & Mazzuchi, T & Sarkani, S 2020, 'A method for the inclusion of human factors in system design via use case definition', *Human-Intelligent Systems Integration*, pp. 1–12, <https://doi.org/10.1007/s42454-020-00011-1>.
- Lucquiaud, V & Scapin, D & Jambon, F 2002, 'Outils de modélisation des tâches utilisateurs: exigences du point de vue utilisation', In *Proceedings of the 14th Conference on l'Interaction Homme-Machine*, pp. 243–246.
- Madni, AM 2005, 'Integrating humans with and within complex systems. *CrossTalk*, 5.
- Millot, P & Boy, GA 2012, 'Human-machine cooperation: a solution for life-critical Systems?', *Work*, vol. 41, no, Supplement 1, pp. 4552–4559.
- Orellana, DW & Madni, AM 2014, 'Human System Integration Ontology: Enhancing Model Based Systems Engineering to Evaluate Human-system Performance', *CSER*, pp. 19–25.
- Orellana, DW & Madni, AM 2018, 'An Architecture Profile for Human–System Integration', *Disciplinary Convergence in Systems Engineering Research*, Springer, Cham, pp. 395–406.

- Roschelle, J & Teasley, SD 1995, 'The construction of shared knowledge in collaborative problem solving', In *Computer supported collaborative learning*, pp. 69-97, Springer, Berlin, Heidelberg.
- Scapin, DL & Pierret-Golbreich, C 1989, 'MAD: Une méthode analytique de description des tâches', *Actes du colloque sur l'ingénierie des Interfaces Homme-Machine*.
- Van Der Veer, GC & Lenting, BF & Bergevoet, BA 1996, 'GTA: Groupware task analysis-Modeling complexity', *Acta psychologica*, vol. 91, no. 3, pp. 297-322.
- Watson, M. E & Rusnock, CF & Colombi, JM & Miller, ME 2017, 'Human-centered design using system modeling language', *Journal of Cognitive Engineering and Decision Making*, vol. 11, no 3, pp. 252-269.
- Hause, M & Wilson, M 2017, 'Integrated Human Factors Views in the Unified Architecture Framework', In *INCOSE International Symposium*, vol. 27, no. 1, pp. 1054-1069.
- Paternò, F 2004, 'ConcurTaskTrees: an engineered notation for task models', *the handbook of task analysis for human-computer interaction*, pp. 483-503.

Biography



Kahina Amokrane-Ferka is a senior research engineer in Human Machine Interaction and is currently working as a system architect at IRT SystemX, France. Her education includes a Computer Science degree from UMMTO University, Algeria, and then a Master's research degree from Evry Val d'Essonne University, France. She received a PhD degree in Information Technologies and Systems from UTC of Compiègne, then worked as an academic researcher for over four years. Her research interests include knowledge management, human factors, intelligent systems, simulation, system engineering, system modeling and human system integration.



Andreas Hein is a senior research engineer and system architect at IRT-SystemX, working on autonomous transportation systems. He was formerly an assistant professor for systems engineering at the Industrial Engineering Lab at CentraleSupélec – Université Paris-Saclay. He obtained his Bachelor's and Master's degree in aerospace engineering from the Technical University of Munich and conducted his PhD research at the same university and at the Massachusetts Institute of Technology.