

Article

Modeling Requirements for Collaborative Robotic Services

Oscar Stiven Morales Zapata ^{1,2}, Yaney Gomez Correa ¹, Leopoldo Rideki Yoshioka ¹
and Jose Reinaldo Silva ^{1,*}

- ¹ College of Engineering, Universidade de São Paulo, São Paulo 05508-010, Brazil; oscar.s.morales.z@usp.br (O.S.M.Z.); ygc8104@usp.br (Y.G.C.); leopoldo.yoshioka@usp.br (L.R.Y.)
² Faculty of Science and Engineering, Universidad Autonoma de Manizales, Carrera 9a, 19-03, Manizales 170001, Colombia
* Correspondence: reinaldo@usp.br

Abstract: Collaborative robots have experienced low acceptance in applications, especially in industry. This fact has attracted the attention of researchers and practitioners, who point to different causes for this limited acceptance. One of the main reasons is the difficulty in converging on suitable methods for modeling collaborative interactions between robots and their surrounding context during the requirements phase. These interactions must be elicited and modeled during the requirements stage to maximize value creation through collaboration. Formal verification is necessary, taking into account the risks of human-robot interaction. However, such modeling is often absent in collaborative robot design, and choosing an appropriate approach remains an open problem. This paper addresses this problem using a model-based requirements cycle where the value creation is detached to provide direct analysis, possible optimization, and formal verification. The general process integrates with the general model-based requirements engineering of the remaining system. This service system approach relies on a goal-oriented requirements approach, and specific algorithms were developed to transfer goal-oriented diagrams into Petri Nets—to provide formal process verification. A case study illustrates the application of the proposed method on a collaborative robot used in a university hospital environment.

Keywords: systems design; requirements engineering; collaborative robots; human–robot interaction; value co-creation



Citation: Zapata, O.S.M.; Correa, Y.G.; Yoshioka, L.R.; Silva, J.R. Modeling Requirements for Collaborative Robotic Services. *Eng* **2023**, *4*, 2941–2959. <https://doi.org/10.3390/eng4040165>

Academic Editor: Huanyu Cheng

Received: 9 September 2023

Revised: 28 October 2023

Accepted: 3 November 2023

Published: 21 November 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

According to Federico Vicentini, collaborative robots “is an umbrella term that conveys the general idea of proximity between machines and humans for a useful task” [1]. This definition summarizes the current trend generalizing the use of collaborative robots in different application domains from manufacturing [2,3] to health applications [4], including RPA (Robotics Process Applications) [5,6]. However, widening the scope of human-robot to a generic cognitive human-machine interaction would raise a challenge on how to design these systems with *humans-in-the-loop* [7,8].

Therefore, there is a demand to revise collaborative robot design to achieve a more comprehensive understanding, beyond its functionality, focusing on the value added by collaboration with humans or other machines. The value co-creation added by collaboration must be treated as the focus for mixed robotic and human systems instead of a functional composition of individual machines and humans in a hybrid network [9,10].

There is a convergence among researchers and practitioners about a generalized terminology covering a broad scope of collaborative robotic systems. Its primary design factors are actuation, control, physical interaction, usability, and productivity [1,11]. For service collaborative robotics, a perceptron is crucial to increase the ability to sense the state of the environment and capture the environment state and its influence to achieve collaborative goals [12].

Together, all these features fit the service concept derived from Service Science. An automated service provider depends on coupling with a service client to achieve value creation resulting from collaboration. Therefore, the provider is designed to offer a service coupling for a hybrid network of humans and machines.

Service Science [13] emerged with IBM researchers about 2004 and has conceptually and scientifically developed, gaining new topics such as Service Engineering, including robotics application [14]. Service Engineering started proposing different methods for requirements modeling and formalization [15]. Some authors proposed an approach based on objectives (Goal-Oriented Requirements) [16], revisited here as an alternative to provide a human-centered method for collaborative automated systems (a generalization of collaborative robots). The method generates a model-based requirement cycle formalized in Petri Nets—a suitable representation to model the behavior and communication of an automata network.

Collaborative robots or “cobots” are fundamental for the future of factories and complex human tasks. The combination of robot characteristics (repeatability and tirelessness) and flexible user experience improves productivity and can be critical to the success of automated processes. Cobots are flexible, perform the switch from different tasks, and are easy to install, program, and reprogram, even by non-experts. Existing technologies for sensing and communication [17] incorporate perceptron devices such as laser sensors and vision, recognizing gestures and human expressions or attending voice commands while avoiding collision and performing other superposed tasks. Human safety is one of the most critical considerations in human-robot interaction, preventing collisions between humans and robots operating within a shared space and considering all possible ways to avoid harming a person [18]. Completing the mission will require path and motion planning to provide input to the collaborative features [19,20].

Two major approaches for robotic task execution have emerged [21]: path/motion planning, which is hardly applicable to human-robot collaboration because of the unpredictable and dynamic nature of humans, and sensor-based control (more efficient and flexible for human-robot interaction, since it closes the perception-to-action loop at a lower level than path/motion planning).

This work focuses on interactive and sensor-based collaboration, including a user experience based on actions. Therefore, the approach fits AI planning methods where actions lead to the system’s goal. However, even before that, the primary concern is collaboration design, which demands three layers: path/motion planning, sensor-based control, and collaborative interaction with humans and other machines (which also uses part of the sensor-based layer). In this work, we focus on the last one.

Our contribution starts with the requirements analysis, providing rationales to guide deployment or used to revise a previous project to fit the proper user experience that maximizes value generation. We propose a goal-oriented method instead of a functional approach, which uses a formal representation in Petri Nets for verification. A generalization of this proposal will allow the application of the same approach to other application domains.

2. Requirements for Modeling Collaborative Robots

The human factor demands dynamic changes during the design and in revision, maintenance, or innovative enhancements. Cobots often rely on special-purpose hardware and operating software and attend requirements directed toward specific purposes. Consequently, specifications for user interactions need to be modeled more closely, resulting in more flexibility and design reusability. Additionally, bringing intelligence to the project requires tightening robotic systems’ sensing, processing, and acting with (AI) planning capabilities.

In this scenario, the specification of requirements plays a crucial role in embodying robotic system functionalities into a proper and complete application [22]. An approach for the requirements in a generic system domain is unsuitable due to the diversity and complex interactions involved. There are some attempts to map generic features, even if

missing a formal verification, that mention difficulties dealing with uncertainty, defining appropriate models, integrating requirements models, including reusability, synchronizing with reference architectures, addressing NFR (Non-Functional Requirements). In practice, different applications present a different subset of difficulties.

Wang [17] summarizes the relationship involving humans and robots in a shared domain as,

- *coexistence*—when a robot and a human are in the same physical space but do not overlap each other’s workspace;
- *interaction*—when a human and a robot share the same workspace and communicate with each other, but the human and the robot work on the same task stepwise, in sequential order;
- *cooperation*—developed among human and robot agents who have their autonomy, expressed in terms of goals, objectives, utility or profit,
- *collaboration*—when there is a joint activity of humans and robots in a shared workspace to accomplish a set of given working tasks cooperatively.

Different collaborative interactions between humans and robotic systems should be defined (during requirements specification) and modeled in this initial design phase and must be formally verified.

This paper proposes a goal-oriented approach to model human-robot interactions with humans and a surrounding environment to achieve goals defined by collaboration requirements. Requirements models are represented by goal-oriented service-based diagrams, transferred to XML, and verified using Petri Nets formalism. Notice that most goal-oriented methods and platforms usually address software design. Here, we start to work to adapt these systems to model automated systems that involve hardware and software, following the Model Base Requirements Engineering Approach [23].

Collaborative robotics can benefit from a goal-oriented approach for requirements, model-based, designed over the path/motion and sensor-based layers [4]. Therefore, we will briefly present the basic concepts of Goal-Oriented Requirements Engineering (GORE) in the following sub-section to highlight the choice for a goal-oriented approach and justify the Model-based Requirements Engineering cycle proposition.

2.1. The Requirements Cycle in the Design of Collaborative Automated Systems

Goal-oriented requirements rely on using a set of diagrams composing a framework called KAOS. The KAOS model was initially developed by the University of Oregon (Eugene, OR, USA) and the University of Louvain (Belgium) in 1990 [24] to address the design of software systems. It focuses on goals (objectives) instead of functionality and can postpone the need to balance functional and non-functional requirements at the beginning (before requirements were already synthesized). Goals describe the desired states often reached with the collaboration of dynamic agents instantiated by machines or humans. This network approach makes KAOS particularly suitable for designing service systems.

The KAOS model comprises four diagrams: the objective, object, operation, and responsibility [24]. In this article, we adapt KAOS representation for systems modeling, possessing a social goal instantiated by the diagram’s primary goal. The domain definition needed to extend to encompass collaborative or concurrent systems, following the INCOSE proposition [25]. Expectations in the model are adapted to include observable events that may not be controllable. With these adaptations, the goal is to provide a framework for requirements by defining the responsibilities of system agents and establishing effective communication with different stakeholders. Operations were assigned to agents (eventually humans) responsible for achieving the requirements, utilizing resources (objects in the system) if necessary.

The motivation for the authors to propose a KML representation is to provide an input description that can generate automatic documentation (not explored in this article) and to allow a semantic transference algorithm to Petri Nets place/transition, one of the contributions of this article. Closing a requirements engineering life cycle is not possible without this semantic transference [4].

Service design has a human-centered approach directing its iterative exploration, ideation, reflection, and implementation process. Typically, the inception process promotes user and stakeholder participation in the co-creation of solutions [26]. Therefore, the main challenge relies on searching for new solutions or implementing a better set of sub-processes that satisfy the requirements while leading to better performance. Frequently, different processes demand greater flexibility in the tasks developed. The parties involved (suppliers and consumers) should be aware of objects or agents (people, machines, resources, among others) that can change the system's state. Therefore, service design is responsible for modeling, analyzing, and specifying design processes that effectively cover a set of goals called service requirements.

Value co-creation is the final goal of service systems, reinforcing collaboration, focused on human-robot interaction, or including the interaction with other automated systems. Recently, value co-creation gained more momentum because of the underlying logic of organizations [27]. Cover tools and procedures support new techniques applied to different service domains, such as robotic restaurants, where robots interact, assist, and serve customers in activities such as hosting, taking orders, delivering food, and cleaning.

In the value co-creation process, the consumer is the actor in focus. The co-creation process involves active interaction between the user and the product-service provider. The value generated by this interaction cannot be obtained by the action of any of the parts separately and is associated with the satisfaction of the system's objectives and expectations. This paper will use the KAOS operations diagram to model user interactions and service-generated values.

Goal-oriented requirements are modeled diagrammatically using KAOS (Knowledge Acquisition Object System) following the requirements elicitation phase and continue during the cycle depicted in Figure 1. Figure 1 has two additional modules, addressing new models to the original MBRE model. A goal model following the proposed method can improve the first elicited model. Once the model is enhanced, the refinement cycle will proceed, repeating the process each time a cycle is completed. Therefore, the user becomes a dynamic agent instead of a static end-user.

KAOS representation has stood since the beginning of this century [28]. It comprises four related (but not redundant) modeling diagrams: goal, object, operation, and responsibility, which could be an advantage compared to other general proposed methods, such as UML. Another advantage of KAOS is that it collapses the dichotomy between functional and non-functional requirements into goals - at least at the goal modeling stage. On the other hand, recent developments have adapted UML to represent systems in general, formalizing it in SysML.

This work is part of the research to enhance KAOS to represent systems in general, using Petri Nets as a sound formal representation. The advantage is the better scalability of Petri Nets and the possibility of using many other developments in formal verification and workflow analysis that use this formalism. Another possibility is to use hierarchical or high-level nets, which this article will not address.

Semantic transference between semi-formal requirements models to Petri Nets attracted much attention, starting with the attempt to capture UML diagrams [29,30]. Later, the possibility to do that using other diagrams (such as KAOS) and Petri Nets appeared, capturing all interaction among agents and the system [4]. The Petri Net resulting from this method can be formally analyzed structurally or by its behavior, using properties such as invariants, liveness, reversibility, or necessary conditions to be deadlock-free. A revision process will look for gaps that justify enhancing the requirement's initial model.

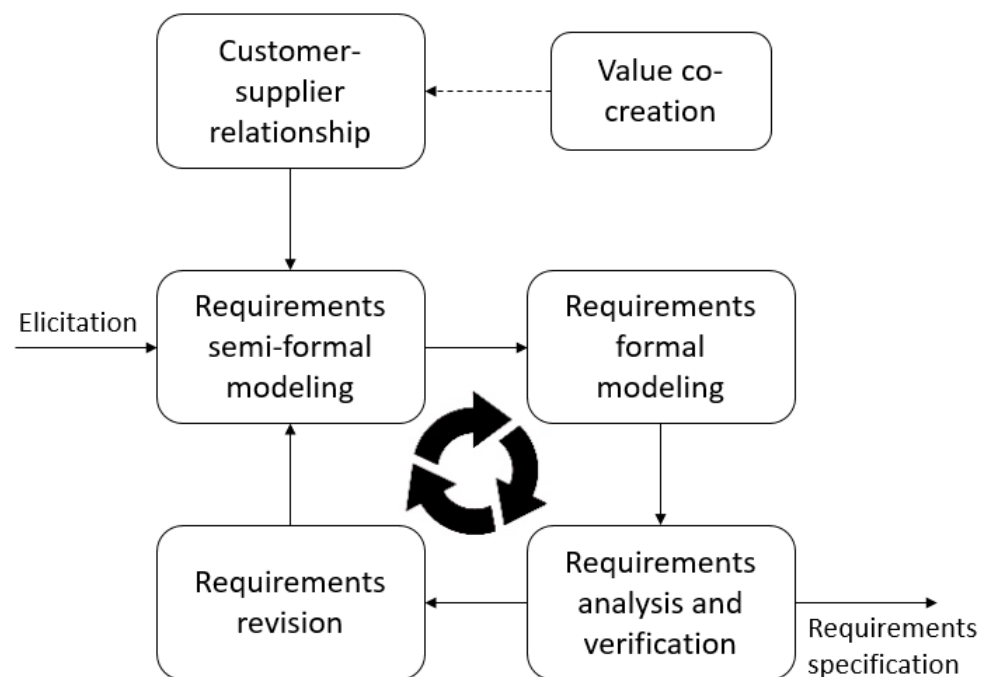


Figure 1. Proposal for modeling and requirements analysis.

Thus, once completing a requirements modeling, documented in a specific *eXtensible Markup Language* (XML) for KAOS, called *KAOS Markup Language* (KML), they could be transferred to Petri Nets by an algorithm that automates this process. This article contributes to both KML and the transfer algorithm.

We apply KML in the general requirements model and the enhancement modeling provider-consumer or human-robot interaction. The KML approach can provide the necessary attributes - in a hierarchical and structured way - to describe all the relationships between the objectives, objects, responsibilities, and operations diagrams.

The KML document used to document diagrams in different abstraction levels has two blocks: one with the metadata for the document and the other with the XML code for KAOS.

1. Prolog: Contains a sequence of processing instructions and/or document type declaration. It can be divided into two parts:
 - A KML declaration defines the KML version, the encoding type, and if it is an independent document (`<?kml version="1.0" encoding="ISO-8859-1"?>`) (https://www.w3schools.com/charsets/ref_html_8859.asp (accessed on 3 November 2023)).
 - The document type declaration defines the type of the document. (Optional).
2. Body: The document information organized as a single tree of marked elements.

The general structure of KAOS and their equivalent KML description (Table 1) are described below:

Table 1. Guideline of KAOS elements-KML representation.

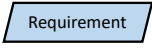
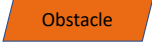
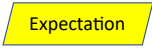
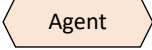
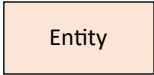
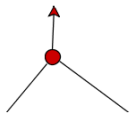
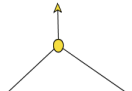
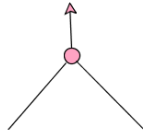
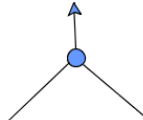
KAOS Elements	Elements Description	KML Standard Code
	Describes an objective to be achieved by the system or one of the subsystems. Special tags are printed blue.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Goals
	A low-level objective with clear satisfaction criteria can be used in test routines. Satisfaction of the requirements should be reflected (using the objectives diagram) in the justification of other objectives related to these requirements.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Requirement
	An unfavorable condition or constraint to the satisfaction of goals, usually originating in the context or domain of the overall goal.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Obstacle
	Used to designate the existence of a conflict between objectives or an object that prevents the satisfaction of the objective.	<code><ConflictTo>....</ConflictTo></code>
	Denote results expected by the agents. It does not have clear criteria for satisfaction. Therefore, their satisfaction, partially defined, does not affect the satisfaction of the associated requirements. However, satisfaction of expectations can influence the satisfaction of agents, which is important if they are part of the coupling of services.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Expectation
	Denote systems external to the system under consideration that may be collaborative or supportive. In the case of support systems, they can be associated with invariants or with a hypothesis, in the case of the collaborative system (an external objective to be satisfied), and in this case, it must contribute to the global objective of the system being modeled.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Domain
	Active physical objects (system agents), humans, machines, or logical objects (software agents), are capable of performing operations that eventually change the state of the system.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Agent

Table 1. Cont.

KAOS Elements	Elements Description	KML Standard Code
	Represent passive and independent objects. Passive objects cannot perform operations; and are independent if they follow the object-oriented model with “behavioral completeness”. Its attributes, whose admissible values must be specified, define the state of the system.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Entity
	Use it to start or stop operations. They can be external or produced by operations (they become the output of the operation).	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Event
	Used to connect (responsibility) to the responsible agent for requirements or expectations.	<code><ResponsibilityOf>....</ResponsibilityOf></code>
	Used by different agents to enforce requirements. The operations can create objects, trigger state transitions, and activate another series of operations by calling another event or operation.	<code><Id>...</Id></code> Identifier of element <code><Name>....</Name></code> Name of the element <code><Type>....</Type></code> Diagram elements type: Operation
	Collector node used to connect objective with its sub-goals. Each connection contributes to the satisfaction of the objective in an inclusive way (AND node). Conventionally, if there is a partial order between the sub-goals, it should be represented from left to right. Alternative connections (OR nodes) will be represented by independent connectors.	<code><ToRefineAnd>....</ToRefineAnd></code> or <code><ToRefineOr>....</ToRefineOr></code>
	Collector node used to connect the agent with the goals, expectations, or requirements for which this agent is responsible. The same convention applies to inclusive and alternative connections.	<code><AssignedTo>....</AssignedTo></code>
	Used to connect the operations with requirements (A requirement is said to be “closed” when this triangular Responsibility-Operationalization-Performance relationship has been established)	<code><OperationalizationOf>....</OperationalizationOf></code>

We detach two elements in the description of KML: *expectation* and *domain*, redefined to fit an extended version of KAOS (the KAOS+). The enhancement in *expectation* covers the need to specify the expected behavior of service customers following the coupling. *Domain* element had a secondary role in KAOS and was enhanced to represent the external systems in the design context, which is also crucial in customer coupling.

A document with an extension .kml can operate transference between platforms. A Petri Net is synthesized by an algorithm transferring KML to PNML - Place/Transition nets are the target in this work. We call this process KML2PNML transfer (depicted in Figure 2). The resulting PNML file goes to any Petri Net system environment for formal verification. The code for this algorithm is available on GitHub <https://github.com/osmz/KML2PNML>, accessed on 3 November 2023.

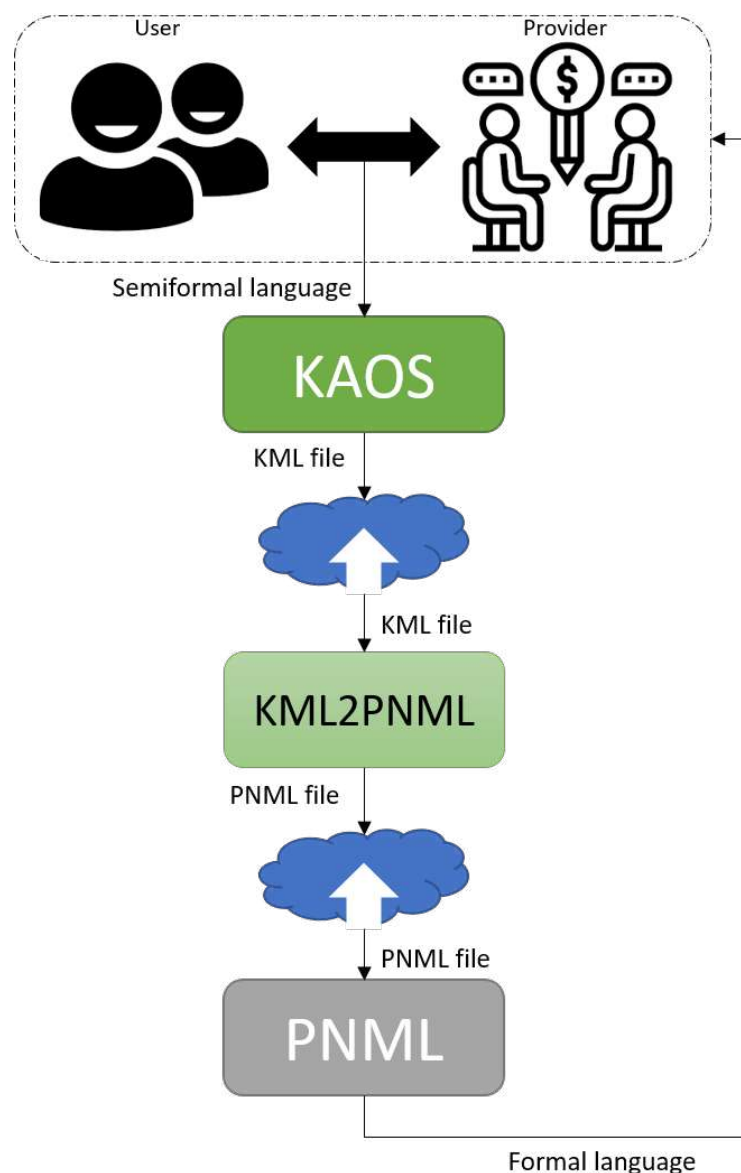


Figure 2. KML2XML transfer proposal.

2.2. The Goal-Oriented Service Approach

The goal-oriented treatment for requirements engineering has been discussed since the late 90s and used in different academic and commercial tools. MBRE aims to add traceability and reusability to improve diagram analysis and validation and introduced its

use to design service networks. Therefore, it includes migrating the whole process to the cloud [31].

The proposed method can also contribute to automating the documentation and reuse of service systems, exploring the fact that the formal model in Petri Nets is a schema applicable to different domains. Soundness for the transference relies on ISO/IEC standard, which defines Petri Nets and the transfer language PNML [32]. PNML is part of Petri Net Standard ISO/IEC 15.909 (<https://www.iso.org/standard/67235.html> (accessed on 3 November 2023)), adopted since 2004.

Therefore, the suitability for using the Goal-Oriented Service Approach is justified by its main characteristics, summarized as follows:

- (i) Using goal-oriented modeling is suitable to describe a service network architecture involving the interaction with different automation systems and collaborative robots.
- (ii) The method focuses on all service interactions, especially the human-robot relation, responsible for the value co-creation.
- (iii) The proposal revises KAOS diagrams to improve service network requirements and design.
- (iv) We developed an algorithm to transfer KAOS diagrams to KML, which can automate documentation, support reusability, and fit the requirements cycle described in Figure 1.
- (v) We developed an algorithm to transfer KML to PNML to get a formal model for verification. The formal model verifies the intended behavior, as well as its structure.

The proposed method starts with a preliminary model for the requirements elicited, modeled in KAOS+ diagrams, standing for the system-as-is. Such a system will be the input for the MBRE cycle, generating refined models that improve the system and its collaborative coupling. After the necessary refinement cycles, a specification, called the system-to-be, is ready for a solution-matching phase. In each cycle, formal verification can support the intention to evolve by conservative extensions, adding performance to the process.

3. A Case Study of Collaborative Robots in a Hospital Environment

The role of collaborative robots grew during the pandemic when the logistics of hospitals and medical centers were affected by the need for isolation and extra care to avoid contamination. It was clear that besides consuming human support, a large amount of “services” could be managed by collaborative machines and sometimes by regular service-delivering automation.

The term service has a specific meaning, taken from the Service Science, related to the interaction carried out by a minimum of two agents: the service provider and the service consumer. In some situations, the same agent can occupy a duplicated mission, for instance, managing and controlling smart-grid systems based on co-production. In such cases, the service consumer is also a producer [33]. A collaborative robot working in a hospital environment has different categories of service consumers but deals with one at a time. These consumers are clearly defined and do not perform a dual role simultaneously. The chosen case study observed that a transportation robot must deal with human agents and automated systems (machines), reinforcing the demand for flexibility and careful design [34].

The case study comprises a robot developed by Professor Yoshioka (co-author of this paper) using a control functional approach without considering collaborative modeling in the requirements phase. Therefore, the goal (objectives) related to collaboration and coupling with service consumers were mixed with control. The prototype works very well but still needs attention concerning value co-creation and the diversity of situations where the robot could intervene in a critical environment such as a hospital.

The project is a partnership between the University Hospital Innovation Center (the University Hospital can attend to 258 patients besides the emergency unit) and the College of Engineering. The idea was to have a flexible, autonomous robot to automate logistic

demands in a hospital, transporting items from place to place or from the automated pharmacy to distinct destinations. The robot could be quickly sanitized after going to sensitive regions concerning contamination. The robot's mission would be to transport medication or goods (small equipment and medical resources, including personal protective equipment).

The first step consists of capturing the legacy system requirements (the system-as-is) to revise it and design a new version (the system-to-be). The intention is not to make a direct comparison, as the requirements follow different approaches, but to focus on the service network's architecture, especially the collaboration modeling.

3.1. The System-As-Is

The collaborative robot has as functional specification a mobile system weighing 40 kg, capable of carrying up to 20 kg. It is 120 cm high and can move with a maximum of 5 km/h. Figure 3 shows the robot's layout with open and closed doors for the cargo compartment. The robot has doors that open and close automatically, high-capacity rechargeable batteries, sensors for the perception of the environment, a wireless communication system, a high-performance hardware platform, and navigation software based on Artificial Intelligence (AI).



Figure 3. The collaborative mobile system functionally designed and implemented.

Concerning collaboration, the robot must interact with humans and an automated system that dispenses medicines automatically. As mentioned, delivering medicines from the internal pharmacy is one of the most demanding actions in the hospital processes. This process is fully automated, as shown in Figure 4.



Figure 4. The automated system to dispense pharmacos.

The hospital communicates with the Automated Logistic Center by Wi-Fi to synchronize the demands for medication pickup. Thus, a user can request medication from the automated pharmacy delivered using the robot service or request that an item be transported from one place to another in the hospital. Suppose the robot is not involved in another mission. In that case, it goes to the pharmacy (or the pickup point), which internally prepares to transfer the requested medication to the robot storage compartment. The robot will deliver the cargo (the medication or an item) to the specified destination. The supervision center stores all mission data for further performance analysis.

Figure 5 captures the functional approach summarized in Section 3. We can focus on the goal diagram to identify some critical features that clarify the introduction of collaboration: The “Hospital Logistic Automation” is divided into two objectives, “Collaborative Robot” and “Automatic Medication Dispenser”, each presenting automated functions such as Movement, Collision-Detection, Collision-Avoid, and Planning. Subgoals refine objectives and finally attribute responsibility to agents. We will focus on the (human) agents and their interaction with the system. There is a conflict related to the failure to deliver the order and three actions that will depend on the user (expectations).

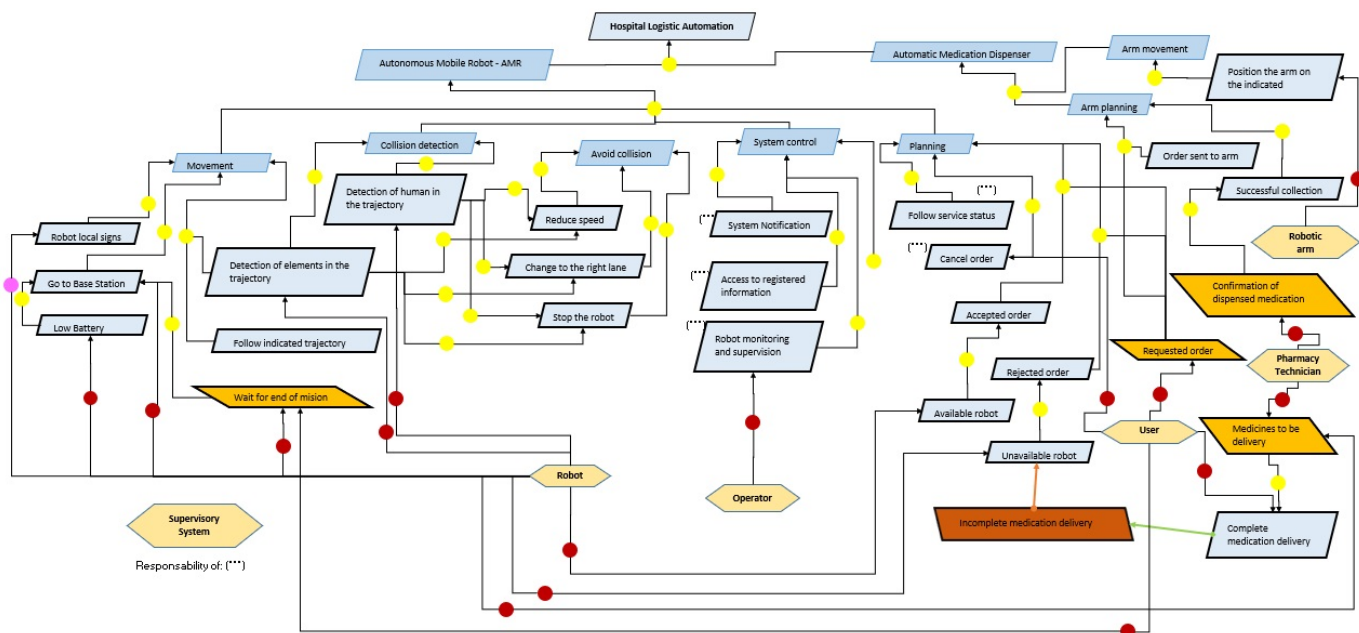


Figure 5. Goal-oriented model for the original project.

Objects are either active (which can change the system state, as humans and machines) or passive (resources and material). They appear explicitly in the goal diagram (Figure 5) as the yellow hexagon. Passive elements could be physical (as the medicine to be delivered from the pharmacy or the items to be) or virtual - as the requests or confirmation to dispense medicines. Active objects (agents) are artificial (the collaborative robot, the robotic arm of the dispenser), virtual (the supervisory system, the control in the automatic dispenser), or human (the final user, the operator of the logistic center, the operator in the pharmacy). Collaboration means different journeys involving two or more of these agents.

The goal model depicts the relationship and dependencies among the objectives (goals). Following the original specification strictly, we focus on the relationship between human agents requesting the transportation of equipment, PPE, or other medical resources except for medication. The (human) user journey does not appear explicitly in the original goal-oriented version of the original project version, which is a consequence of a design where the collaboration results from the control instead of motivating the control algorithm. Therefore, collaboration performance could be questioned even if the system works.

In the original system, the profusion of direct connections can bring unnecessary constraints that obfuscate the value creation relation, the focus of a collaborative system. In Figure 5, the supervisor is passive since all the control emerges from direct mobile communication. As a result, there is also a lack of structuration, reflecting a secondary role for the human agent and a weak co-creation value.

The following section will show a different model, with a service-oriented approach, focusing on the (human) user and its journey to clarify collaborative automation.

3.2. The System-to-Be: Introducing Human–Robot Collaboration

Before focusing on the value creation and interaction between the service producer and customer, it is necessary to review the system-as-is to improve modularity and identify the points for collaboration. The requirements review reinforces the separation between three services: the collaborative robot, the Logistics Automation Center, and the Automated Medication Dispenser (service network). In this architecture, the Logistics Automation Center orchestrates the whole process. It interacts with the (human) user by receiving and acknowledging requests, which are also stored for future flow and demand analysis, as originally specified.

The service interaction between the Logistic Center and the Collaborative Robot is now based on sending *mission plans* and receiving acknowledgments about the mission accomplishment using a protocol controlled by the first.

A *mission plans* is composed of:

- (i) A *pre-plan* requesting the robot to move from its current position to the point where it should get some item to deliver.
- (ii) The *plan*, fitting classic AI definition, where the robot follows the trajectory between the pickup and delivery point, avoiding obstacles and managing emerging occurrences (such as being out of power). The routing system is embedded in the robot control and is not part of the value creation.
- (iii) The *post-plan*, regarding the plan mentioned above, analyzes its impact on the hospital workflow and logistics.

However, the focal point of this article is how to approach collaborative requirements for an environment populated by humans and machines using the service network approach. Requirements specifications should reflect the integration, as it occurs in systems design [25]. Collaboration would benefit from a service-oriented approach, providing better solutions for further innovations or maintenance. Finally, applying a goal-oriented service approach allows specifying, during the requirements phase, the coupling and value co-creation that arise when dealing with humans and machines.

Based on the original requirements, represented in the system-as-is shown in Figure 5, a revision for the project resulted in the system-to-be (Figure 6), corresponding to a service-oriented network view for the same system. The overall interaction of the automated logistics system is mediated by the operator agent, who attends to the user by receiving and handling requests. A human user interacts with the robot only in specific transactions concerning the loading/unloading of the items transported or delivered and following the robot during the execution of the plan. In conflicting situations competing for space, the human is a privileged obstacle. Meaning the robot must stop until the path is free.

The goal diagram corresponding to the system-to-be (Figure 6) was substantially reduced compared to the original system-as-is (Figure 5) due to the implementation of the service network, which offers the possibility to make use the same targets in different actions if the processes are the same (folding). The new distribution refines the identified objectives better, leading to a better understanding of what is desired and expected from the product-service. Another significant difference is the integration of “Supervisory Sys” and “Supervisory System” in the system-to-be, avoiding redundancy. The comparison should consider that the review started from a model extracted from functional specifications, characterized by mixing functionality, resources, actions, and conditions, which undoubtedly influenced the objectives. This limitation will be removed in other diagrams representing operations, linking objects and agents in a responsibility relation.

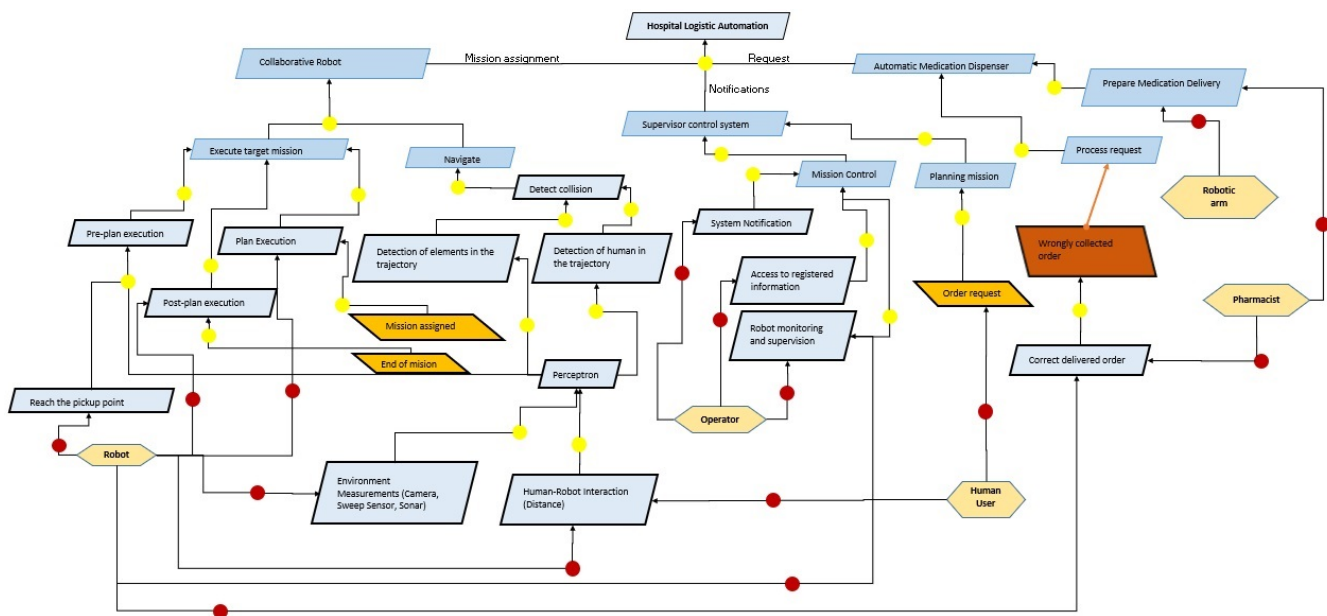


Figure 6. Goal-oriented model for the revised specifications.

The revision led to the following enhancements:

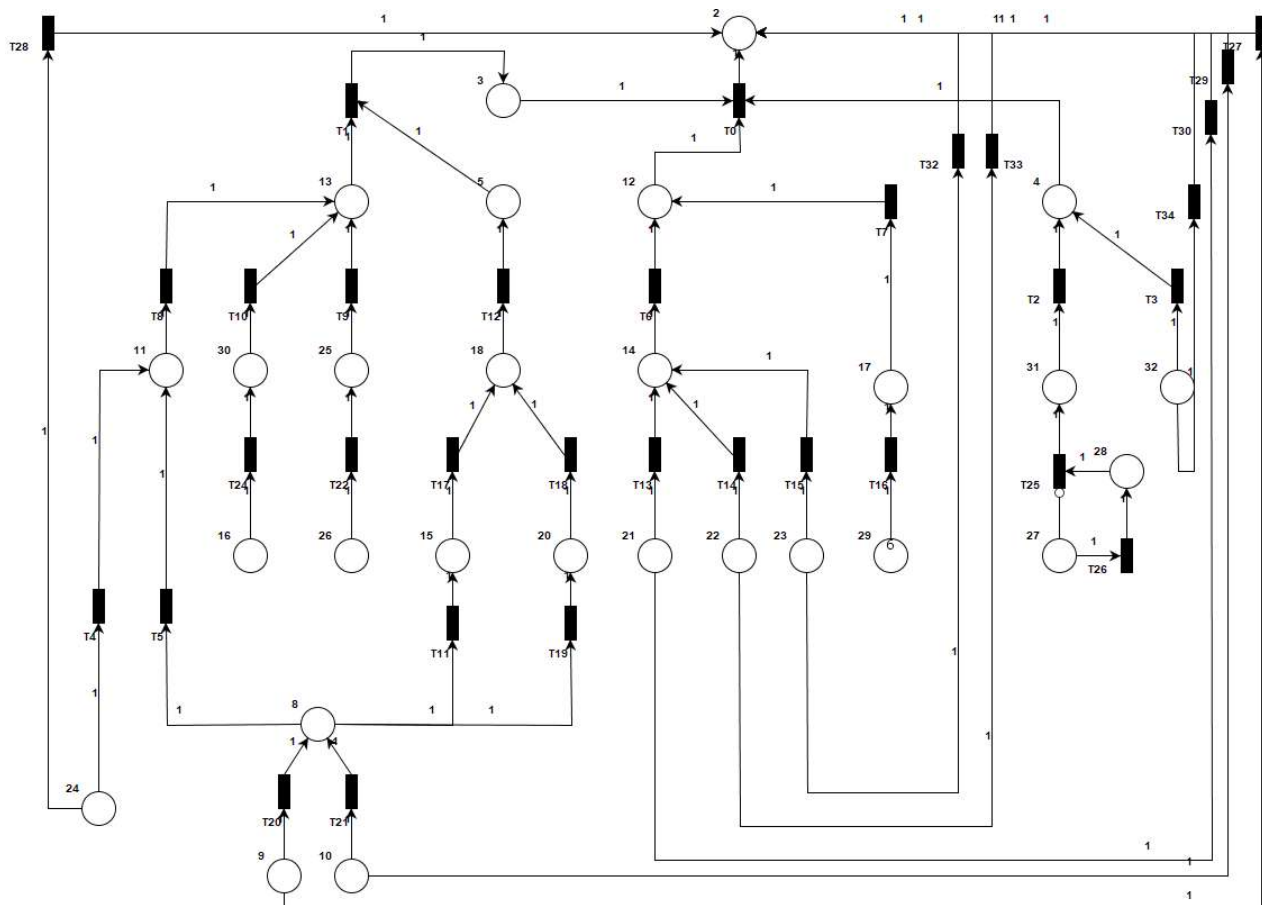
- (i) Identify and separate services (Autonomous Mobile Robot—AMR, Supervisor control system, and Automatic Medication Dispenser) to focus on service-oriented network architecture.
- (ii) Collapse functionalities, actions, rules, and conditions into targets (including possible non-functional requirements). Objectives represent goals identified in the user-supplier interaction and must be satisfied by the robot collaborative missions orchestrated by the Hospital Logistic System.
- (iii) Identifies the essential communication between the logistic services and the agents responsible for issuing and receiving goods. In other words, it allows us to attribute/responsabilize each identified requirement to an agent. It is essential to mention that the lowest level of refinement reaches a stage where all actions match one or more agents (the responsibility diagram).

As shown in Figure 6, one of the objectives is to model logistics automation in the hospital, composing a Collaborative Robot and the Medication Dispenser belonging to the Hospital Pharmacy. Therefore, it is a service composed of other services. The value co-creation is already included (which was not explicit in the original project, although it was in the background specification). A goal-oriented service approach includes the collaboration among services that we added to the former conceptualization of the system. This modeling appeared before as a restriction, more than a requirement specification, opening space for a revision.

Consequently, details of service relations (and their operation) were not explicitly analyzed. Also, the system was conceived interactively but needed to be modeled and analyzed formally. Therefore, it is possible that considering the former project as a collaborative service of services leads to some collaboration flaws. Some details of the execution route of specific goals will appear explicitly in the operations diagram, where the user journey concerning the collaboration between humans and robots will be clarified.

Figure 7 corresponds to the Petri net generated from the KML2PNML transfer. The network allows properties and workflow analysis, particularly observing conflicts or possible interference between identified requirements. Identifying which requirements depend on an external action, known as expectations, is also possible. For a proper understanding of the Petri net, it should be mentioned that the locations (represented by circles) represent requirements, objectives, sub-objectives, and expectations. A Petri Net is

a bipartite graph, and a place (circle) cannot connect to another place without a transition in between (represented by rectangles). Arcs represent the flow direction.



2	Hospital Logistic Automation
3	Collaborative Robot
4	Automatic Medication Dispenser
6	Robot
7	"Robotic arm"
19	Pharmacist
11	Pre-plan execution
12	Supervisor control system
13	Execute target mission
15	Detection of elements in the trajectory
16	End of mission
5	Navigate
14	Mission Control
17	Planning mission
18	Detect collision
20	Detection of human in the trajectory
21	System Notification
22	Access to registered information
23	Robot monitoring and supervision
8	Perceptron
9	Environment Measurements (Camera, Sweep
	Sensor, Sonar)
10	Human-Robot Interaction (Distance)
24	Reach the pickup point
25	Plan Execution
26	Mission assigned
27	Wrongly collected order
28	Correct delivered order
29	Order request
30	Post-plan execution
31	Process request
32	Prepare Medication Delivery

Figure 7. Petri Net of goal-oriented model for the revised specifications.

A reliable net has been synthesized to support the verification of requirements and analyze the structural and dynamic properties of the model. Every system modeled in Petri Nets admits a unique algebraic representation given by matrices representing the structure of the model. The requirements model can be transferred to a transition system (T-System) represented by the state that follows (Equation (1)).

$$M_i = M_0 + A^T \sigma_{i-1} \quad (1)$$

where: M_0 : initial state, A^T : transposed incidence matrix, σ_{i-1} : initial marking vector.

3.3. Modeling the Human–Robot Interaction

The case study of a collaborative logistics robot will illustrate the method concerning the analysis of value co-creation. However, the justification for its use is practical and conceptual, as mentioned above. Since the proposed method concerns requirements, it is not possible to have a complete formalization.

The collaborative interaction between a service provider and customer presupposes a work area for this relationship, considering a (human) agent and a robotic system. That means to identify all relations between these elements in the system-to-be.

In the system-to-be, this points to a relation between the user and the Planning goal of the Collaborative Robot to request or cancel an order and feedback sent to the automated medication dispenser. Since orders must be orchestrated together, that suggests a change to concentrate user relationship with the Supervisory System.

Assuming that change in the system-to-be, we can now concentrate on the collaborative relation between human agents and the robot, enhanced to avoid overwhelming master-slave commands. An Operation Diagram can be synthesized only for this relation, as shown in Figure 8.

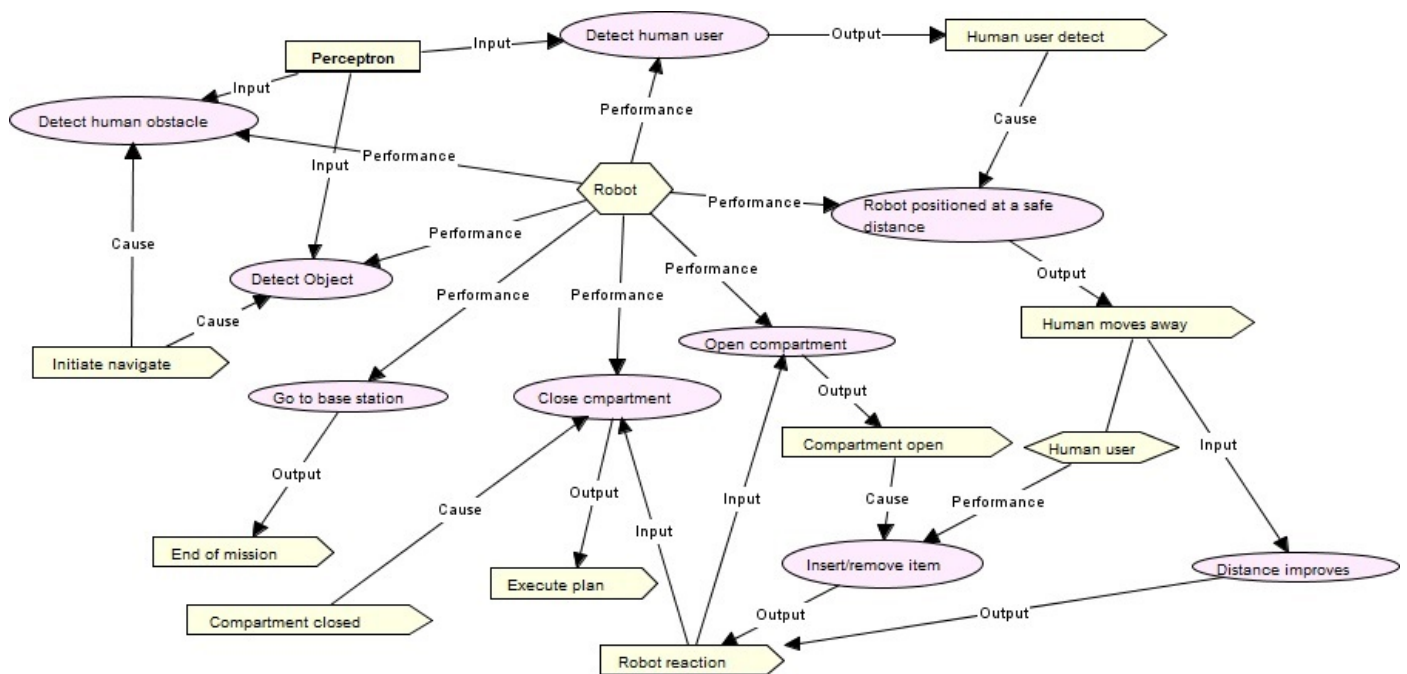


Figure 8. Operation diagram.

The operation diagram Figure 8 shows two different processes for human-robot relation: In the first, the robot detects an obstacle and recognizes (using visual robot control) that it is a human. The robot completely stops its movement and waits for the human to move. This process is shown as a “deadlock” in the diagram since this does not constitute collaboration. The other possibility composes a collaborative process starting with the robot positioning “at a safe distance” in front of the user (it uses the visual human detection

capability of the robot and the control to stop at that distance to avoid harming the users when the compartment opens automatically).

At a secure distance, the robot opens the compartment automatically and waits for the user to reduce this distance and eventually insert an item into the compartment. If the user does so and moves away from the robot (get to a safe distance again), the robot closes the compartment and prepares to leave to perform its mission (executing the remaining plan sent by the Supervisory). Once the mission finishes, it goes to a base station to recharge or wait for the next call.

Figure 9 shows the Petri net model synthesized using the algorithm transferring from KML to PNNML and visualized in any Petri net environment. In this work, we used the Pipe3, v.4.3, (available at <https://www.softpedia.com/get/Others/HomeEducation/PIPE2.shtml>, last accessed on 21 October 2023). The net formally presents the control sequence, starting with the navigation to accomplish the mission.

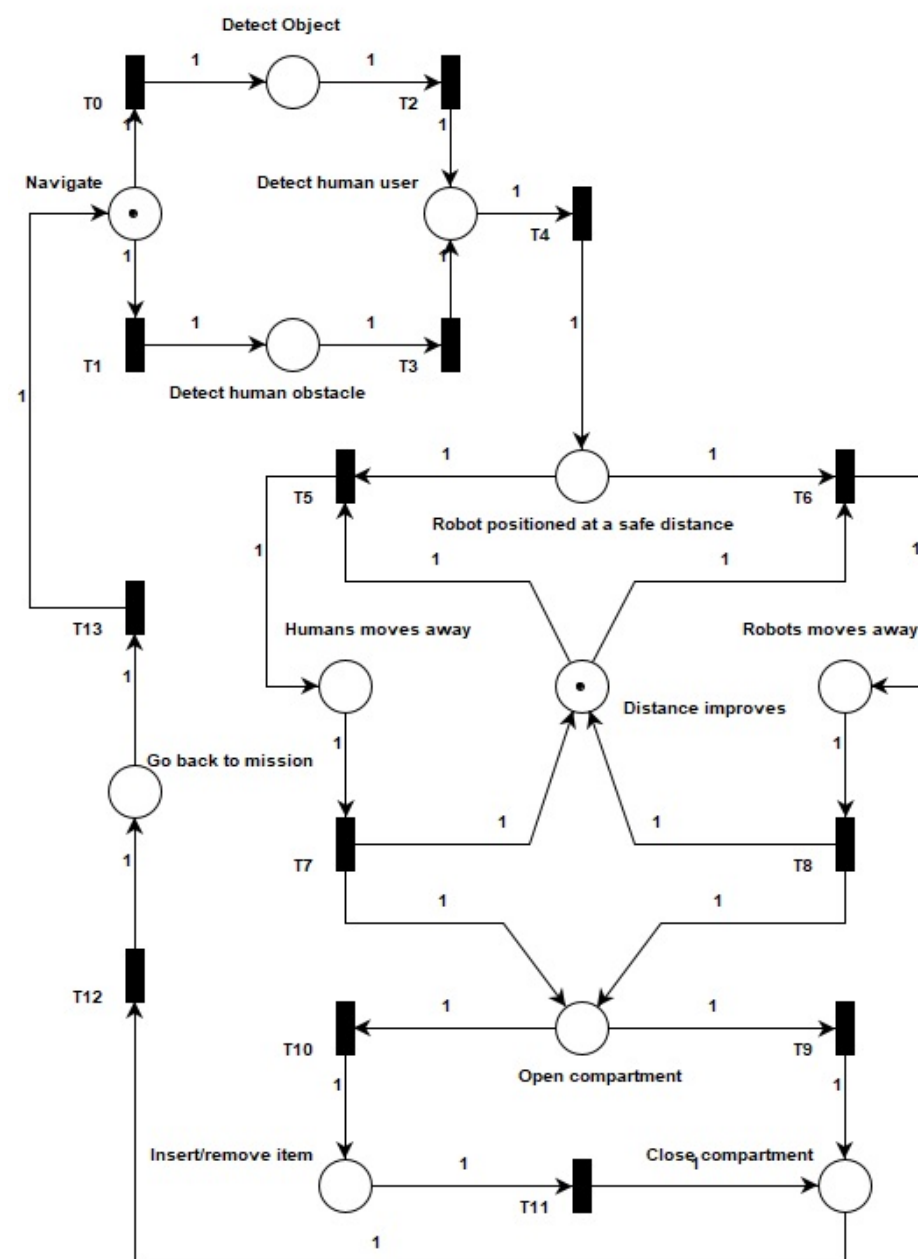


Figure 9. Petri Net of operation diagram.

The locality of transitions T5, T6, T7, and T8 shows the management of safe conditions to open/close the compartment, where a pre-marked place signals that a safe distance between the robot and the human must be respected to allow automatic actuation opening or closing the compartment. The Petri Net can formally verify this condition.

However, it is possible for the human (for any reason) to move away, setting the safety distance without inserting or retrieving an item in the compartment. In this case, the robot returns to its mission empty. Although possible, this is not desirable. The suggestion to avoid this case is to insert a sign from the (human) user canceling the mission. Concerning the design process, this could be done by changing the Petri Net and synthesizing the KAOS+ to proceed with the MBRE refinement.

The relation is almost linear, and a short invariant analysis will show that the summation of all marks is $\sum m_i = 2$, denoting a conservative net. The only problem is the conflict generated by firing transitions T9 and T15, leading to an empty mission.

4. Conclusions

This paper applies the goal-oriented service approach to collaborative robotics. A step-wise cycle of requirements development called Model-based Requirements Engineering (MBRE) served as a basis [4] to support a systemic method for synthesizing concise and close requirements for the coupling between the service system and the customer. The benefit is to have a service architecture covering all its communication, including collaboration, integrated with the general model that could be formally verified, focusing on value creation.

Using XML to transfer requirements models from general proposed diagrams (such as UML) to Petri Nets is nothing new and follows the tendencies of ISO/IEC standards. The novelty is to apply that to goal-oriented diagrams (like KAOS) and contribute to preparing the result to generate documentation and reusability support (using XML). Another significant contribution is assembling value creation in the whole requirements cycle in the MBRE, adding system modeling to the service network concept.

Adapting this framework to collaborative robotics poses the new challenge of combining control and perceptrons with mission workflow and Artificial Intelligence (planning), in addition to the characterization of collaboration. In the case study, the proposal dealt with a robot initially focused on control automation without leaving aside the analysis of the model to include the mission workflow to analyze and detail the collaborative operation.

The distance between the robot and the human user characterizes collaboration. However, error recovery mainly demands improving perceptron to avoid empty missions. Missing this improvement would result in unwanted situations of undelivered (or unloaded) items. The interaction should be enhanced by adapting the robot control with perceptrons that can detect and analyze (human) movements, suggested for further developments.

Independently of the sophistication level of the robotic system, investing in a preliminary design process can bring new ideas and open possibilities (and functionalities) to be included in the design process. Revising requirements elaborated by other approaches—as developed above—is also an option but opens the possibility to emerging requirements that conflict with the implementation adopted. Intense conflicts did not appear in the case study because the original design was transparent, flexible, and based on a lean solution.

For future research, we propose to work with the formalization of the method in a framework (so far, it is a collection of tools) implemented in the cloud. As for the application in collaborative robotics, the challenge is to apply the same approach to systems with multiple collaborative problems, starting with adding motion detection, voice communication, or cognitive anticipation of user intentions. Therefore, scalability is also a concern, and using High-Level Nets is possible [35]. That allows us to explore symmetry and fold the net.

Another exciting possibility is to explore knowledge engineering for planning to extend the possibilities of the automatic planning system to cope with multiple missions or changes in the current plan [15].

Author Contributions: The authors O.S.M.Z., Y.G.C. and J.R.S. contributed for the conceptualization of the requirements engineering process and KML+, to the method directed to value creation and the KML2PH software developed, as well as to the formal analysis in Petri Nets. L.R.Y. contributed with the conceptualization, development and implementation of the collaborative robot and its control, as well as to the direct validation of requirements method and its results associated to the case study. All authors worked in the writing of the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The code for the algorithms mentioned in this work is available in <https://github.com/osmz/KML2PNML>.

Acknowledgments: The authors would like to thank CAPES for financing graduate student scholarships. Also, for the research infrastructure created by the Graduate Program in Mechanical Engineering of the Polytechnic School of the University of São Paulo. The authors acknowledge the University Hospital Innovation Center for supporting the project with the case study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vicentini, F. Collaborative Robotics: A Survey. *J. Mech. Des.* **2021**, *143*, 040802. [CrossRef]
2. Francesco, P.; Paolo, G.G. AURA: An Example of Collaborative Robot for Automotive and General Industry Applications. *Procedia Manuf.* **2017**, *11*, 338–345. [CrossRef]
3. Proia, S.; Carli, R.; Cavone, G.; Dotoli, M. A Literature Review on Control Techniques for Collaborative Robotics in Industrial Applications. In Proceedings of the 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), Lyon, France, 23–27 August 2021; pp. 591–596. [CrossRef]
4. Silva, J.R.; Macedo, E.C.T.; Correa, Y.G.; Medeiros, R.P. A Multilayer Proposal to a Smart Home Applied to Healthcare. *Polytechnica* **2021**, *4*, 1–14. [CrossRef]
5. Cabello, R.; Escalona, M.J.; Enríquez, J.G. Beyond the Hype: RPA Horizon for Robot-Human Interaction. In *Lecture Notes in Business Information Processing*; Springer: Cham, Switzerland, 2020; Volume 393, pp. 185–199. [CrossRef]
6. Cabello Ruiz, R.; Jiménez Ramírez, A.; Escalona Cuaresma, M.J.; González Enríquez, J. Hybridizing humans and robots: An RPA horizon envisaged from the trenches. *Comput. Ind.* **2022**, *138*, 103615. [CrossRef]
7. Maurice, P.; Schlehuber, P.; Padois, V.; Measson, Y.; Bidaud, P. Automatic selection of ergonomie indicators for the design of collaborative robots: A virtual-human in the loop approach. In Proceedings of the 2014 IEEE-RAS International Conference on Humanoid Robots, Madrid, Spain, 18–20 November 2014. [CrossRef]
8. Maurice, P.; Padois, V.; Measson, Y.; Bidaud, P. Human-oriented design of collaborative robots. *Int. J. Ind. Ergon.* **2017**, *57*, 88–102. [CrossRef]
9. Gualtieri, L.; Rauch, E.; Vidoni, R. Emerging research fields in safety and ergonomics in industrial collaborative robotics: A systematic literature review. *Robot. Comput. Integr. Manuf.* **2021**, *67*, 101998. [CrossRef]
10. Shaumburg, H.; Korabiev, V.; Laszlo, U. *Technological Transformation: A New Role for Human, Machines and Management*; Springer International Publishing: Cham, Switzerland, 2020; Volume 157. [CrossRef]
11. Saenz, J.; Elkmann, N.; Gibaru, O.; Neto, P. Survey of methods for design of collaborative robotics applications—Why safety is a barrier to more widespread robotics uptake. In Proceedings of the 2018 4th International Conference on Mechatronics and Robotics Engineering, Valenciennes, France, 7–11 February 2018; pp. 95–101. [CrossRef]
12. Lv, Z.; Qiao, L. Deep belief network and linear perceptron based cognitive computing for collaborative robots. *Appl. Soft Comput.* **2020**, *92*, 106300. [CrossRef]
13. Qiu, R. *Service Science*; John Wiley & Sons: Hoboken, NJ, USA, 2014.
14. Čaić, M.; Odekerken-Schröder, G.; Mahr, D. Service robots: Value co-creation and co-destruction in elderly care networks. *J. Serv. Manag.* **2018**, *29*, 178–205. [CrossRef]
15. Silva, J.R.; Silva, J.M.; Vaquero, T.S. Formal Knowledge Engineering for Planning: Pre and Post-Design Analysis. In *Knowledge Engineering Tools and Techniques for AI Planning*; Vallati, M., Kitchin, D.E., Eds.; Springer: Berlin/Heidelberg, Germany, 2020; pp. 47–66.
16. Horkoff, J.; Aydemir, F.B.; Cardoso, E.; Li, T.; Maté, A.; Paja, E.; Salnitri, M.; Piras, L.; Mylopoulos, J.; Giorgini, P. Goal-oriented requirements engineering: An extended systematic mapping study. *Requir. Eng.* **2019**, *24*, 133–160. [CrossRef] [PubMed]
17. Wang, L.; Gao, R.; Váncza, J.; Krüger, J.; Wang, X.V.; Makris, S.; Chrysosolouris, G. Symbiotic human-robot collaborative assembly. *Cirp Ann.* **2019**, *68*, 701–726. [CrossRef]

18. Lasota, P.A.; Fong, T.; Shah, J.A. A Survey of Methods for Safe Human-Robot Interaction. *Found. Trends Robot.* **2017**, *5*, 261–349. [[CrossRef](#)]
19. Kiani, F.; Seyyedabasi, A.; Nematzadeh, S.; Candan, F.; Çevic, T.; Anka, F.A.; Randazzo, G.; Lanza, S.; Muzirafuti, A. Adaptive Metaheuristic-Based Methods for Autonomous Robot Path Planning: Sustainable Agricultural Applications. *Appl. Sci.* **2022**, *12*, 943. [[CrossRef](#)]
20. Tuba, E.; Strumberger, I.; Zivkovic, D.; Bacannin, N.; Tuba, M. Mobile Robot Path Planning by Improved Brain Storm Optimization Algorithm. In Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8. [[CrossRef](#)]
21. Cherubini, A.; Navarro-Alarcon, D. Sensor-Based Control for Collaborative Robots: Fundamentals, Challenges, and Opportunities. *Front. Neurobotics* **2021**, *14*, 576846. [[CrossRef](#)] [[PubMed](#)]
22. Albuquerque, D.; Castro, J.; Souza, A. A Requirement Definition Framework for Robotic Systems Domain: An exploratory study. In Proceedings of the Workshop in Requirements Engineering, Rio de Janeiro, Brazil, 5–6 September 2018.
23. Zhen, L.; Li, R.; Liu, X.; Chen, Y. A New Practical Robust Control Design for Model-based Uncertain Collaborative Robot. In Proceedings of the 2023 International Conference on Advanced Robotics and Mechatronics, (ICARM), Sanya, China, 8–10 July 2023; pp. 768–773. [[CrossRef](#)]
24. van Lamsweerde, A.; Darimong, R.; Letier, E. Managing conflicts in goal-driven requirements engineering. *IEEE Trans. Softw. Eng.* **1998**, *24*, 908–926. [[CrossRef](#)]
25. Cloutier, R.J.; Hutchison, N. *SeBok: Guide to the Systems Engineering Body of Knowledge*, 2nd ed.; INCOSE: San Diego, CA, USA, 2022.
26. Liu, Z.; Ming, X.; Song, W.; Qiu, S.; Qu, Y. A perspective on value co-creation-oriented framework for smart product-service system. *Procedia CIRP* **2018**, *73*, 155–160. [[CrossRef](#)]
27. Proper, H.A.; Bjeković, M.; Feltus, C.; Razo-Zapata, I. On the development of a modelling framework for value co-creation. *CEUR Workshop Proc.* **2018**, 2239, 122–132.
28. van Lamsweerde, A. Goal-oriented requirements engineering: A guided tour. In Proceedings of the 5th IEEE International Symposium in Requirements Engineering, Toronto, ON, Canada, 27–31 August 2001; pp. 249–263.
29. Basile, F.; Chiacchio, P.; Del Grosso, D. Modelling automation systems by UML and Petri Nets. In Proceedings of the 2018 9th International Workshop on Discrete Event Systems, Gothenburg, Sweden, 28–30 May 2008; pp. 308–313. [[CrossRef](#)]
30. Zhang, H.-X.; Zhu, L.-Z. Building Dynamic Model in UML Using Colored Petri Nets. In Proceedings of the 2009 International Symposium on Computer Network and Multimedia Technology, Wuhan, China, 18–20 January 2009; pp. 1–4. [[CrossRef](#)]
31. Silva, J.R.; Vital, E. Towards a Formal Design to Service-Oriented Cloud Manufacturing. In *Proceedings of the Congresso Brasileiro de Automática*; Bernardon, D.P., Vargas, F.L., Eds.; Blucher: Santa Maria, Brazil, 2020; pp. 8–16.
32. Kindler, E. The Petri Net Markup Language and ISO/IEC 15909-2: Concepts, Status, and Future Directions. In *Design of Complex Automation Systems*; Schnieder, E., Ed.; ifak—Institut für Automation und Kommunikation: Braunschweig, Germany, 2006; pp. 35–55.
33. Orellana, M.A.; Silva, J.R.; Pellini, E.L. A model-based and goal-oriented approach for the conceptual design of smart grid services. *Machines* **2021**, *9*, 12. [[CrossRef](#)]
34. Roy, R.B.; Lillrank, P.; Sreekanth, V.K.; Torkki, P. *Designing Service Machines*; Springer: Singapore, 2019. [[CrossRef](#)]
35. Silva, M. 50 years after the PhD thesis of Carl Adam Petri: A perspective. *Ifac Proc. Vol.* **2012**, *45*, 13–20. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.