Journal of Ambient Intelligence and Humanized Computing (2021) 12:9667–9692
https://doi.org/10.1007/s12652-020-02713-5

ORIGINAL RESEARCH

# Customisable assistive plans as dynamic composition of services with normed-QoS

Claudia Di Napoli[2] · Patrizia Ribino[1] · Luca Serino[2]

Received: 1 August 2020 / Accepted: 18 November 2020 / Published online: 3 January 2021
© The Author(s) 2021, corrected publication 2021

**Abstract**

Ambient assisted living solutions (AAL) aim to improve the quality of life, especially of elderly with cognitive and physical disabilities, by providing assistive tasks such as medication management, reminders of daily activities or medical tests. However, the loss of physical functions (such as mobility, eyesight and hearing), as well as the loss of cognitive functions (starting from amnesia to severe disabilities as dementia or Alzheimer), require appropriate solutions in order to offer personalised assistive tasks according to user needs and characteristics. In this paper, we propose a novel methodology that is based on the integration of a service-oriented approach with normative reasoning to automatically generate assistive tasks customised for different target user's profiles, and deployable in any AAL environment. The formalisation of the conceptual steps of the methodology is provided, together with a framework implementing them. The advantages of dynamic customisation obtained by decoupling the functional and not functional aspects of assistive technology are shown through a validation scenario.

**Keywords** Ambient assisted living · Customisable assistive plans · Norms · Quality of service · Service composition

## 1 Introduction

Ambient assisted living (AAL) environments are becoming a more and more realistic opportunity due to the advances in Information and Communication technology that allow equipping environments with smart devices providing functionalities not available since recently. These smart environments may provide ubiquitous support to people, allowing to automatically control smart devices to evaluate the health state of users, to avoid dangerous situations, and to improve environmental comfort (Woźniak and Połap 2019; Lauraitis et al. 2019; Ribino et al. 2020; Wozniak et al. 2020). Among the several applications of AAL, those in the healthcare domain are very attractive since they provide support to the elderly in their everyday life, partially relieving caregivers and/or nurses tasks. Nevertheless, their applicability in real settings is still far from reaching its potential because of the difficulties in the acceptance of assistive technology, above all, when dealing with fragile classes of users as the elderly.

In this context, personalisation based on specific characteristics and needs of each user is a crucial requirement in order to make assistive care supported by modern technology effective. The possibility to adapt the way the assistive tasks are delivered depending on unforeseen events causing either a change of the user's state or devices availability can further improve the acceptance level of technology due to the adoption of a full user-centric vision. In addition, adaptation allows deploying assistive care applications in several AAL environments, such as nursing home, private houses and hospitals that can be equipped with different technological devices, without relying on an all in one solution.

A technologically supported assistive task is usually a composition of atomic sub-tasks so that it can be accomplished by services offered by different devices (i.e., their providers). With the increasing development of the Internet of Things technology, several devices may be available with the same functionality, but different non-functional characteristics. Hence, it becomes essential to select services that can, on the one hand, satisfy user features and needs, and,

✉ Patrizia Ribino
patrizia.ribino@icar.cnr.it

Claudia Di Napoli
claudia.dinapoli@cnr.it

Luca Serino
luca.serino@cnr.it

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, C.N.R., Palermo, Italy

[2] Istituto di Calcolo e Reti ad Alte Prestazioni, C.N.R., Naples, Italy

on the other hand, be provided by the devices available in a specific AAL environment may provide. So, the provision of customised assistive tasks should rely on the choice of an appropriate set of services/devices satisfying not only functional objectives of the composite task but also preferences, needs and characteristics of the target user and the environment.

In this paper, we propose a methodology to automatically generate assistive tasks that can be deployed in any AAL environment, relying on the integration of a service-oriented approach with normative reasoning to allow for dynamic customisation of such tasks with respect to both user characteristics and available devices. The methodology consists of representing non-functional characteristics of services in terms of quality of service (QoS) attributes, and the preferences and needs of users in terms of norms tailored to different classes of users. Services providing the required functionalities and the QoS attributes values compliant with norms for classified users are automatically selected at runtime, so composing personalised assistive plans. A framework to realise the proposed methodology is provided, and it is used to implement a validation scenario that allows describing its main features.

The rest of the paper is organised as follows. Section 2 reports some related works on QoS-based service selection. Section 3 introduces concepts related to the service-oriented approach for assistive technology and an overview of normative reasoning. Section 4 shows how the service-oriented approach is used in the proposed methodology. Section 5 presents the proposed methodological approach and the norm-based specification framework. In Sect. 6, an application scenario is illustrated, while an experimental validation of the approach reported in Sect. 7. Finally, in Sect. 8, discussion and conclusions are presented.

## 2 Related works

The concept of quality of service has become an important selection criterion for addressing the problem to select the most suitable services for performing a composite task. Usually, the considered QoS attributes refer to non-functional service characteristics that can be domain dependant or domain-independent, and they are related to objective factors under the control of their providers. Typical QoS attributes are execution time, reliability, availability and cost. For these attributes metrics and aggregation functions should be defined for designing automatic methods to select services that meet the required criteria (Ardagna and Pernici 2007; Aggarwal et al. 2004). There are several approaches proposed in the literature for performing QoS-based service selection and composition that use global or local selection criteria (Hong and Hu 2009), and propose static or dynamic methods (Rao and Su 2004).

A global approach selects the best composition according to global constraints. Services are selected in such a way the QoS required for the entire composite task, known as end-to-end QoS constraint, is achieved.

The problem to find a combination of services whose QoS values, once aggregated, meet the QoS end-to-end constraints is known to be NP-hard. It has been formulated as an optimisation problem solved with static approaches assuming that fix QoS values for each service are provided, so relying on prior knowledge of the component service QoS values. Some works propose algorithms to select services using linear integer programming methods to maximise a utility value given by the weighted sum of QoS attributes as in Ardagna and Pernici (2007). Local constraints are included in the linear programming model used to satisfy global QoS constraints. Nevertheless, the computation complexity increases exponentially with the number of component services. In Alrifai and Risse (2010) mixed integer programming is used to find the optimal decomposition of global QoS constraints into local constraints, and then local selection finds the best services that satisfy these local constraints. Some studies tried to extend the multi-dimension multiple-choice 0–1 knapsack problem (MMKP) with heuristic algorithms to get near-optimal solutions for service selection problems (Yu et al. 2007).

Several service selection frameworks based on fuzzy multicriteria decision making (MCDM) methods have been proposed in the literature. In Masdari and Khezri (2020) MCDM techniques are adapted to handle fuzziness of users' preferences and QoS properties to provide a more effective ranking of available services to select the most appropriate ones according to QoS factors. In Qi et al. (2018), a discrete differential evolution algorithm for global quality composition optimisation is proposed, evaluating the typical QoS attributes that are service response time, availability and reliability.

Recently, agent-based methods were also developed for QoS-aware service composition. In Naseri and Navimipour (2019) a new hybrid approach based on an agent-based method to compose services by identifying the QoS parameters is proposed. Then, a particle swarm optimisation (PSO) algorithm is employed for selecting the best services based on a fitness function. In Chaib et al. (2018), a model of adaptive service composition in ambient intelligence environment based on multi-agent system is presented. The agents are associated with each object of the environment, and they can coordinate their competences to provide suitable QoS values that meet some requirements, considering the context. An intelligent lighting system of a room office containing several and different light sources is presented as an exemplification of the approach. The main objective of the system is to illuminate a space with the desired lighting intensity to satisfy the lighting needs with minimum energy consumption.

Heuristic algorithms, such as genetic ones, have also been proposed to solve QoS-aware service selection finding approximate solutions as they are faster to reach than optimal ones (Canfora et al. 2005). Besides, they can be applied with non-linear QoS aggregation functions, while traditional approaches, as linear integer programming, require linearization. In Thangaraj and Balasubramanie (2020), authors proposed a hybrid meta-heuristics genetic algorithm with tabu search for retrieving the most appropriate services for the end-user based on the QoS objective parameters such as availability of service, response time, throughput and interoperability between the services. Zhang et al. (2014) proposed to decompose a composite service with a general flow structure into parallel execution paths, and to model the service selection problem for each execution path as a multi-objective optimisation problem, solved with an ant colony optimisation algorithm. In Asghari et al. (2020), privacy is included as another QoS service factor for cloud service composition.

Local approaches select the best candidate services, one at a time, by associating each component task to the best candidate service, as proposed in Zeng et al. (2004).

As concerns static and dynamic approaches, they differ for the time when service selection occurs: at design-time for static approaches, at runtime for dynamic ones with the dynamic binding of selected services. The dynamic approach is more promising than the static one since it limits failures due to the unavailability of services when needed (D'Mello and Ananthanarayana 2010). Other approaches also handle service failure by providing methods for relaxing QoS constraints (Lin et al. 2011) or redundant services when not suitable services are available to meet QoS requirements (Kokash and D'Andrea 2007; Ardagna and Mirandola 2010).

From the literature review arises that an open research field is the definition of new techniques for personalised QoS-based service selection (Muthuraman and Venkatesan 2016). In such context, QoS attributes refer not only to objective factors, depending on their service providers, but also to subjective information depending on their users. Such types of QoS attributes require first the acquisition of user preferences that can either be learned, as proposed in Wang et al. (2007), or be provided by domain experts, and then a decision phase for selecting the suitable QoS-based services. In Ouadah et al. (2019), a hybrid approach to rank services is proposed that mixes methods from the MCDM field, and it evaluates aggregation of weighted QoS criteria that are both objective and subjective.

In the present work, QoS attributes refer to both objective and subjective QoS attributes. The subjective ones refer to the interaction modalities of users with devices providing component services of assistive tasks. The objective ones refer to the energy consumption and the execution time of component services. The knowledge about user preferences is assumed to be provided by professionals in the healthcare domain. The service selection decision relies on a novel approach grounding on the idea to express QoS constraints in terms of norms (i.e., option, obligations or prohibitions) that characterise category of users and the environment, thus creating personalised assistive plans by choosing services in compliance with such prescribed *Normed-QoS*.

## 3 Background

In this paper, we propose a methodology to customise assistive plans to specific user's characteristics, that integrates a service-oriented approach with normative reasoning whose concepts are here briefly introduced.

### 3.1 Service-oriented for assistive technology

An assistive task delivered by a human is usually described as an atomic functionality since a single provider delivers it. Instead, when using assistive technology, several devices are necessary to provide a task, each one responsible for providing a specific functionality. A natural approach to represent an assistive task as a composition of functionalities delivered by different devices is the service-oriented one (Papazoglou et al. 2007). It allows representing a complex functionality as an orchestration of services, each one considered as an autonomous computational entity under the control of decoupled and distributed providers that can be the devices responsible for their provision. These functionalities rely on standard technologies that guarantee their interoperability and execution on distributed computing units.

Moreover, the concept of service allows adopting a uniform representation of different functionalities necessary to deploy a complex application composed of heterogeneous components from a functional point of view. This feature is crucial to manage the diversity of devices that are usually employed to provide assistive services, requiring approaches that are not constrained by a specific technology.

### 3.2 Normative reasoning and deontic logic

Generally speaking, norms can be seen as normative rules used for governing conducts, procedures or state of affairs within a particular sphere of knowledge. The normative reasoning is the reasoning conforming to or based on norms. It is formally studied through the Deontic Logic (Cocchiarella 2015), which is the study of the logical relationships among propositions asserting that specific actions or state of affairs are obligatory, forbidden or permitted. From the application point of view, deontic logic is the logic that deals with actual as well as the ideal behaviour of systems (Wieringa and Meyer 1993). Oppositely from the classic logic, when working with norms, we have to consider that they are neither true nor false. Hence, a typical approach is to consider the deontic logic as a logic of normative propositions. The underlying concept is that, even if norms are neither true nor false, someone may state that something ought to be done (according to norms): the statement *John ought to pay the taxes* is, then, true or false description of a normative situation (Alchourrón 1969).

The Traditional Scheme of deontic logic is based on five normative statuses (McNamara 2006): *it is obligatory that* (OB), *it is permissible that* (PE), *it is impermissible that* (IM), *it is gratuitous that* (GR), *it is optional that* (OP). The Traditional Definitional Scheme (TDS) takes the first as basic, and defines the rest as follows: $PEp \leftrightarrow \sim OB \sim p$ , $IMp \leftrightarrow OB \sim p$ , $GRp \leftrightarrow \sim OBp$ , $OPp \leftrightarrow (\sim OBp \& \sim OB \sim p)$. The first statement asserts that something is *permissible* if and only if its negation is not obligatory. The second and the third expression assert that something is *impermissible* if and only if its negation is obligatory, and *gratuitous* if and only if it is not obligatory. Finally, something is *optional* if and only if neither it nor its negation is obligatory. In addition, a proposition may be obligatory, optional, or impermissible, but it can not belong to more than one of these three categories. Moreover, the permissible propositions are those that are either obligatory or optional, and the gratuitous propositions are neither impermissible nor optional. Furthermore, it is generally assumed that if $p$ it is obligatory then $p$ is true, vice versa if $p$ is true, then $p$ is permissible.

For reasoning with norms, several systems have been introduced, but the benchmark system of deontic logic is the so-called Standard Deontic Logic (SDL). SDL is one of the first deontic logics axiomatically specified (Von Wright 1951; Hansen 2008; McNamara 2006). It builds upon propositional logic and it is obtained from the modal logic. For the scope of this work, we adopt the operators *Obligation (O)*, *Option (Op)* and *Prohibition (P)* conforming to the Traditional Definitional Scheme. In particular, let A be a proposition, *OA* means that *it is obligatory that the situation described by the descriptive sentence A is realised*, *PA* means that *it is prohibited that the situation described by the descriptive sentence A is realised*, finally *OpA* means that *it is indifferent that the situation described by the descriptive sentence A is realised.*

## 4 Assistive tasks as Abstract Workflows

In this work, the service-oriented approach is used at a higher level of abstraction to allow for the dynamic personalisation and adaptation of assistive tasks at runtime starting from their abstract representation in terms of functional components at design time (Di Napoli et al. 2017). In fact, when a task needs to be automatically provided by a set of technological devices, it is necessary to functionally split it into sub-tasks, each one provided by a specific device. The rationale of this choice is that it is usually unlikely that a single device, unless ad hoc designed, can deliver a complex assistive task.

The service-oriented approach is here proposed as a development methodology that allows:

- to adopt a uniform representation of the heterogeneous functionalities, from a functional point of view, composing an assistive task;
- to decouple the functionality provided by a service from the provider responsible for its delivery;
- to orchestrate services provided both by the devices composing the AAL environment and by third parties, so relying on external services when necessary;
- to decouple the functional aspect of service from the different modalities it can be delivered with, usually referred to as its non-functional aspects.

The subtasks composing an assistive task are identified according to the functionalities that devices, potentially available in an AAL environment, may provide. The abstract representation of an assistive task in terms of its functionality is referred to as an *Abstract Workflow* (*AW*), a directed graph where each node is an *Abstract Service* (*AS*), i.e. an atomic functionality that can be concretely delivered by devices of an AAL environment, and arcs represent functional dependencies among nodes, determining their execution order (Di Napoli 2009).

An Abstract Service can also be characterised by non-functional aspects referred to as *Quality of Service* that are domain-dependent attributes, such as execution cost, execution time, and so on (Kritikos and Plexousakis 2009). Hence, we extend the Abstract Service definition to a *QoS-based Abstract Service* (*Q-AS*) meaning an Abstract Service associated with a set of QoS attributes types. Here, QoS attributes refer to the different modalities an AS can be delivered with. The types and the corresponding values of QoS attributes are determined depending on the specific functionality and the corresponding device.

Concrete implementations of a QoS-based Abstract Service are referred to as a *QoS-based Concrete Service* (*Q-CS*), and the devices providing them as *Service Provider*s (*SP*).

The proposed approach to decouple abstract functionalities from their concrete implementation allows associating to a single functionality, different implementations that can be delivered by different devices.

A QoS-based Abstract Service is defined as follows:

$$Q\text{-}AS =< id, fname, abs\_output, (qos_1, .., qos_m) > \quad (1)$$

where *id* is the *Q-AS* identifier, *fname* is the name of the functionality it refers to, *abs_output* is an abstraction of the expected output of the corresponding QoS-based Concrete Service, and $(qos_1, .., qos_m)$ is the list of the QoS attributes types associated to the *Q-AS*.

A QoS-based Concrete Service is defined as:

$$Q\text{-}CS =< id, id_{Q-AS}, id_{SP}, url, in, out, (\overline{qos_1}, .., \overline{qos_m}) > \quad (2)$$

where *id* is the *Q-CS* identifier, $id_{Q-AS}$ is the identifier of the corresponding *Q-AS*, $id_{SP}$ is the identifier of the device providing the *Q-CS*, *url* is the endpoint of the *Q-CS*, *in* and *out* are respectively the input and output of the *Q-CS*, and $(\overline{qos_1}, .., \overline{qos_m})$ is the list of the QoS attributes values associated to the list of QoS attributes types of the corresponding *Q-AS*.

Likewise, a *QoS-based Abstract Workflow* (*Q-AW*) is associated to an Abstract Workflow, and it is defined as follows:

$$Q\text{-}AW =< id, TaskName, Tstart, Tend, AW(Q\text{-}AS_{set}, E) > \quad (3)$$

where *id* is the *Q-AW* identifier, *TaskName* is the name of the assistive task represented by the *Q-AW*, *Tstart* and *Tend* identify the time range of the assistive task duration, and $AW(Q\text{-}AS_{set}, E)$ is a directed graph with $Q\text{-}AS_{set}$ containing the nodes of the graph, and *E* the directed arcs. Each arc represents a transition from an origin node to a target node. Nodes with no outgoing arcs are terminal nodes.

In Fig. 1, an example of an Abstract Workflow representing the assistive task to remind a user to take medicines, *Remind Medicine*, is reported in terms of the Abstract Services necessary to perform the task.

The identified functionalities are:

- to identify user localisation (`Find user`);
- to check that the user is in a physical state suitable to follow the usual pharmacological therapy (`Check user Physical state`);
- to check whether the user has already taken the medicine (`Check taken medicine`);
- to check if the time is still compatible with the time range planned for the pharmacological therapy (`Check time in`);
- to remind the user to take the medicine (`Remind medicine`);

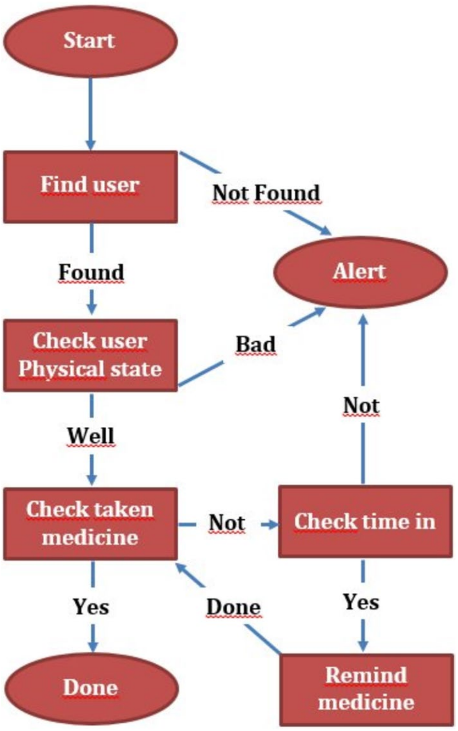


**Fig. 1** The remind medicine Abstract Workflow

- to generate an alert signal in case of detected anomalies (`Alert`);
- to end the assistive task (`Done`).

Each *Q-AS* composing the *Q-AW* can be provided as a *Q-CS* by different devices with different non-func-tional characteristics. For example, *Find user* can be deployed as a service provided by a mobile robot using localisation procedures relying on its sensors, or by a set of cameras installed in the domestic environment. These different devices involve different interaction modalities with the user. So, a QoS type associated with the *Find User* Abstract Service is the level of interaction with the technology that is required to the user.

Starting from the functional representation of an assistive task in terms of a QoS-based Abstract Workflow, the normative reasoning allows to automatically generate the corresponding executable assistive task composed of QoS-based Concrete Services, i.e. concrete implementations of the required functionalities provided by a specific device with specific QoS values. The QoS-based Concrete Services are determined by defining normatively the QoS values that are valid or acceptable for identified classes of users or specific contexts. So, the obtained executable assistive tasks are customised for each user in terms of the QoS values of its components, as the corresponding devices can provide them.

## 5 Normed-QoS for customisable assistive plans

The novel idea grounding on the proposed approach is to define normatively (i.e., in terms of option, obligation or prohibition norms) the quality of service, thus introducing the novel concept of *Normed-QoS* (*N-QoS*).

In this work, norms are conceptualised adopting a typical system of deontic logic that comprises ordinary first-order logic and some deontic operators. Hence, we extend the concepts of *Q-AW* and *Q-AS* to respectively *Norm-oriented Abstract Workflow* (*N-AW*) and *Norm-oriented Abstract Service* (*N-AS*) by introducing a new formalisation based on the first-order logic. This formalisation allows integrating the traditional approach of service composition with a norm-oriented paradigm. In doing so, we differentiate from the classic service orientation approach introducing normative reasoning for the creation of personalised plans. The normative reasoning allows us to select the concrete services that are compliant with a prescribed set of normed-QoS.

### 5.1 Methodological approach metamodel

The methodological approach proposed in this study is based on the metamodel depicted in Fig. 2. It shows the transition from an Abstract Workflow towards a customised plan, referred to as *Customised Concrete Plan* (*CCP*), throughout a Norm-oriented Abstract Workflow model, and a set of normed-QoS. As we can see from Fig. 2, at the abstract layer, a QoS-based Abstract Workflow is defined for performing a required task. At this purpose, starting from the functionalities specified in the Abstract Workflow, a set of

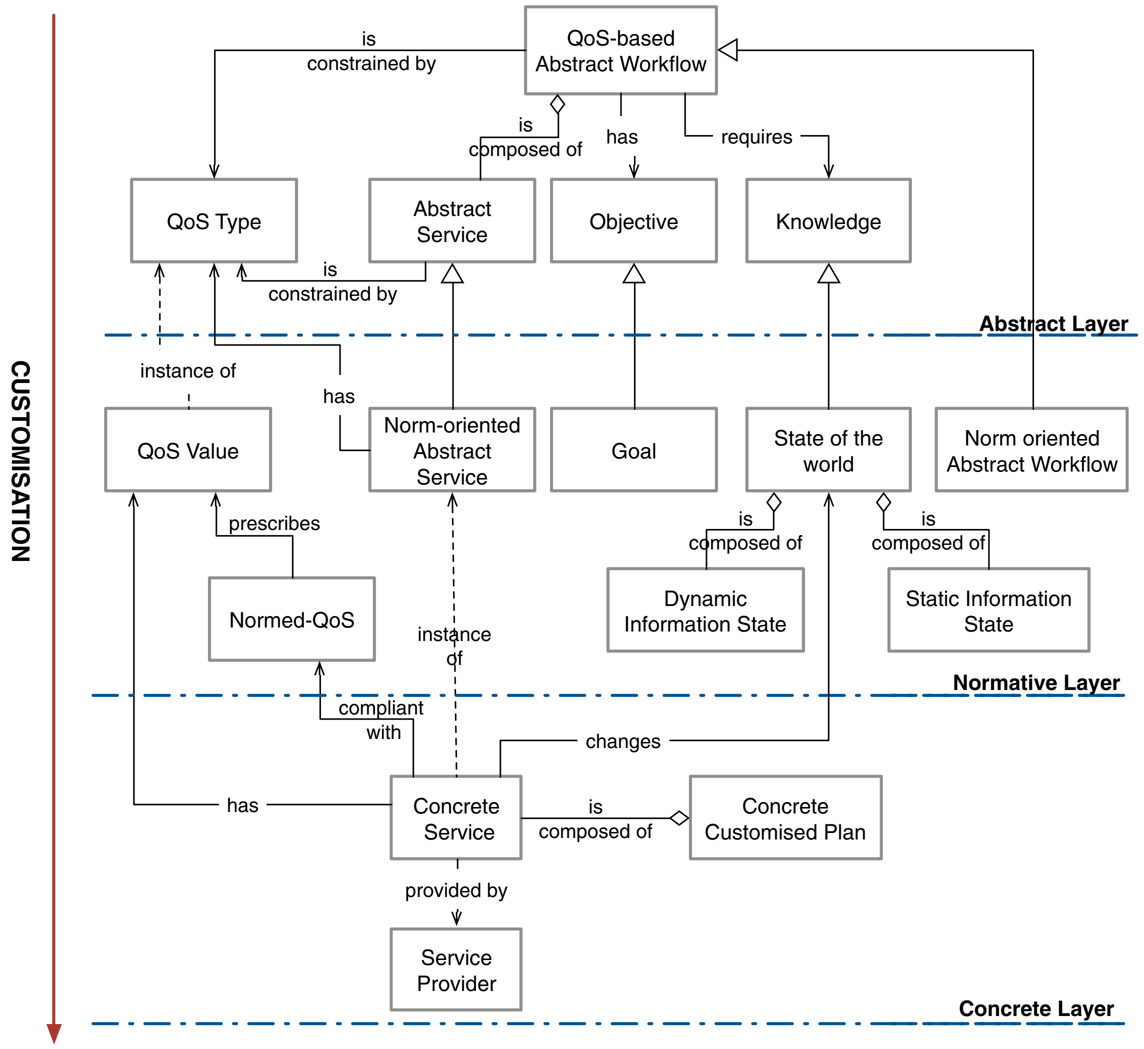


**Fig. 2** Methodological approach metamodel

non-functional requirements have to be identified representing the types of quality of service attributes necessary in order to satisfy user needs or preferences. In some context, such as the healthcare, a set of non-functional requirements may represent a typical or expected pattern (i.e., a norm) that characterises both a category of users, and a particular domain context.

The transition from a *Q-AW* specification to an *N-AW* model is obtained by specialising the elements of the abstract layer (see Fig. 2) with a new set of elements that derive from norm-based theoretical foundations proposed in Ribino and Lodato (2019). In particular, in this work, we propose the novel concept of normed-QoS, thus introducing a Norm-oriented Abstract Workflow. Each element of the normative layer that composes the N-AW model is detailed in Sect. 5.2. Finally, at the concrete layer, the process of customisation is completed through the algorithms introduced in Sect. 5.3 that allow for the generation of Concrete Customised Plans.

## 5.2 Norm-based specification framework

Equation 4 shows the tuple that describes the proposed Norm-oriented Abstract Workflow model:

$$N\text{-}AW = \langle \mathcal{W}^t, \mathcal{G}, N\text{-}AS_{set}, N\text{-}QoS_{set} \rangle \tag{4}$$

where $\mathcal{W}^t$ is the knowledge at a given time *t* modelled by using the concept of *state of the world*; $\mathcal{G}$ is the *Goal* the *N-AW* has to reach; $N\text{-}AS_{set}$ is a set of N-AS ordered according to the linearization of the *Q-AW*; and $N\text{-}QoS_{set}$ is the set of normed-QoS that constraint the way for fulfilling the *N-AW* goal. In detail, the *state of the world* $\mathcal{W}^t$ represents a set of declarative information about events (occurred within the environment), and relations among events at a specific time. Events are defined as occurrences of some facts that can be perceived by or communicated to the system. They can be used to represent information that can characterise the situation of an interacting user, as well as a set of circumstances in which the system operates at a specific time. In this work, we distinguish between static ($\mathcal{SIS}$) and dynamic ($\mathcal{DIS}$) information on the state of the world. A static information state is based on the assumption that it remains constant with respect to time during the system execution, while the $\mathcal{DIS}$ may change. It is due to the effects that the employment of concrete services produces on the state of the world.

**Definition 1** (*State of the world*) Let $\mathcal{D}$ be the set of concepts defining a domain. Let $\mathcal{L}$ be a first-order logic defined on $\mathcal{D}$ with $\top$ a tautology and $\bot$ a logical contradiction, where an atomic formula $p(t_1, t_2..., t_n) \in \mathcal{L}$ is represented by a predicate applied to a tuple of terms $(t_1, t_2..., t_n) \in \mathcal{D}$ and the predicate is a property of or a relation between such terms that can be true or false. A *state of the world* at a given time t ($\mathcal{W}^t$) is a subset of atomic formulae whose values are true at the time t:

$$\mathcal{W}^t = [p_1(t_1, t_2, ..., t_h), ..., p_n(t_1, t_2, ..., t_m)] \tag{5}$$

where the subset of atomic formulae defines both static and dynamic information states as follows:

$$[p_1(t_1, t_2, ..., t_h), ..., p_n(t_1, t_2, ..., t_m)] = \mathcal{SIS} \cup \mathcal{DIS} \tag{6}$$

An example of the state of the world at time *t* could be the following:

$$\mathcal{W}^t = [isUser(user, hearing\_impaired), user\_Location(user, kitchen)]$$

where the first proposition is a static information state, and the second one is a dynamic information state because the user location may change during the time. Definition 1 is based on the close world hypothesis (Reiter 1981) assuming that all facts not in the state of the world are considered false.

The second element of the *N-AW* model is formally defined as follows:

**Definition 2** (*Goal*) Let $\mathcal{D}$, $\mathcal{L}$ and $p(t_1, t_2..., t_n) \in \mathcal{L}$ be the elements previously introduced in Definition 1. Let $t_c \in \mathcal{L}$ and $f_s \in \mathcal{L}$ be formulae that may be composed of atomic formulae by means of logic connectives AND($\wedge$), OR ($\vee$) and NOT ($\neg$). A *Goal* is a pair $\langle t_c, f_s \rangle$ where $t_c$ (*trigger condition*) is a condition to be evaluated over a state of the world $\mathcal{W}^t$ when the goal may be actively pursued, and $f_s$ (*final state*) is a condition to be evaluated over a state of the world $W^{t+\Delta t}$ when it is eventually addressed. A goal is active if Eq. 7 is verified. A goal is addressed if Eq. 8 is verified.

$$t_c(\mathcal{W}^t) \wedge \neg f_s(\mathcal{W}^t) = true \tag{7}$$

$$f_s(\mathcal{W}^{t+\Delta t}) = true \tag{8}$$

A goal is activated when some conditions occur (i.e., the trigger conditions), and it is achieved when a new state of the world contains the final state. In Eq. 8, $\Delta t$ is the frequency at which the system monitors the state of the world.

Resuming the example in Sect. 3.1 that defines an Abstract Workflow for the task of reminding medicine, its goal could be represented according to our formalism as follows:

$$\mathcal{G} = \langle \mathrm{is}Time(T, User, Medicine), remind(User, Medicine, completed) \rangle$$

meaning that the system starts to pursue the goal $\mathcal{G}$ when the trigger condition `isTime(T, User, Medicine)` will be true in a state of the world $\mathcal{W}^t$. The system considers the goal satisfied when the final state `remind(User, Medicine, completed)` will be reached in a state of the world $\mathcal{W}^{t+\Delta t}$.

The third element of the model is $N\text{-}AS_{set}$, where a Norm-oriented Abstract Service is a generalised semantic representation of a QoS-based Abstract Service that can guarantee specific QoS types.

**Definition 3** (*Norm-oriented Abstract Service*) Let $\mathcal{D}$, $\mathcal{L}$, and $p(t_1, t_2, ..., t_n) \in \mathcal{L}$ be the elements previously introduced in the Definition 1. Let $pre\_cond \in \mathcal{L}$ and $post\_cond \in \mathcal{L}$ be formulae composed of atomic formulae by means of logic connectives AND($\wedge$), OR ($\vee$) and NOT ($\neg$). A *Norm-oriented Abstract Service* is defined by the tuple showed by Eq. 9.

$$N\text{-}AS = \langle id, name, pre\_cond, post\_cond, QoS_{set} \rangle \tag{9}$$

where *id* is the *N-AS* identifier; *name* is the *fname* of the corresponding QoS-based Abstract Service it refers to; *pre_cond* is a logic condition that must be satisfied so that the corresponding QoS-based Abstract Service can be chosen, and it is derived from the incoming arcs of the corresponding Abstract Service of the QoS-based Abstract Workflow; *post_cond* is a logic condition that describes as the *abs_output* of the corresponding QoS-based Abstract Service modifies the state of the world; $QoS_{set} = \{qos_1, qos_2, ..., qos_m\}$ is a set of QoS types of the corresponding QoS-based Abstract Service.

Considering the previous example, a Norm-oriented Abstract Service could be represented as follows:

$$\texttt{Check\_User\_Physical\_State(User)} = \langle \texttt{registered(User)}, \texttt{physical\_state(User, well)OR physical\_state(User, bad)}, \texttt{InteractionModality} \rangle$$

The reported *N-AS* corresponds to a QoS-based Abstract Service that is able to check the physical state of registered users. It provides as QoS type the interaction modality referring to a set of predefined features impacting the way users can interact with technology, and their values can be, for example, dialogue or text. As for the *Q-AS*, also the *N-AS* represents an abstraction of the AS `Check_User_Physical_State` that can be performed by an element of $Q\text{-}CS_{set}$ with a given value of the QoS type, for example the dialogue modality.

The last element of the specification framework is the set of normed-QoS, $N\text{-}QoS_{set}$ that constrains the way the goal is achieved.

**Definition 4** (*Normed-QoS*) Let $\mathcal{D}$, $\mathcal{L}$, and $p(t_1, t_2 ..., t_n) \in \mathcal{L}$ be the elements previously introduced in Definition 1. Let $\phi \in \mathcal{L}$ and $\rho \in \mathcal{L}$ be formulae composed of atomic formulae by means of logic connectives AND($\wedge$), OR ($\vee$) and NOT ($\neg$). Moreover, let $D_{op} = \{option, obligation, prohibition\}$ be the set of deontic operators. A *Normed-QoS* is defined by following tuple:

$$N\text{-}QoS = \langle N - AS_{set}, \rho, \phi, d \rangle_{[priority]} \tag{10}$$

where: $N\text{-}AS_{set}$ is the set of Norm-oriented Abstract Services the *N-QoS* is applied to; $\rho \in \mathcal{L}$ is a formula that expresses the QoS values that are prescribed for the QoS types and ruled conditions as well; $\phi \in \mathcal{L}$ is a logic condition (to be evaluated over a state of the world $\mathcal{W}^s$) under which the norm is applicable; $d \in D_{op}$ is the deontic operator applied to $\rho$ defining the type of prescription, where:

$$d(\rho) = \begin{cases} \rho & \text{if } d = obligation \\ \neg\rho, & \text{if } d = prohibition \\ \rho \vee \neg\rho & \text{if } d = option \end{cases} \tag{11}$$

An obligation imposes to obtain the QoS value defined in $\rho$. A prohibition defines $\rho$ as a not-acceptable condition. Finally, options do not have a restrictive role. They are mainly used for indicating desirable but not obligatory QoS values, thus providing alternatives. Moreover, each normed-QoS is associated with a priority that represents a weight indicating its relevance in the particular application domain.

Resuming the previous example, let us suppose to define an assistive plan for reminding medicines for hearing impaired users. It is expected that for this category of users, a fundamental quality of service is the interaction modality with the device. Thus, for personalising the assistive task, a normed-QoS is defined stating that the Interaction Modality must be only visual. In our approach, such a statement is formalised as follows:

$$N - QoS = \langle \texttt{Remind(User, Medicine)}, \texttt{Interaction\_Modality(visual)}, \texttt{isUser(user, hearing\_impaired)}, \texttt{obligation} \rangle_{[1]}$$

Hence, a normed-QoS mainly prescribes acceptable QoS values of the given QoS types for classes of users, and under specific circumstances, thus guiding the selection of norm-compliant QoS-based Concrete Services.

Finally, in our approach, the Norm-oriented Abstract Workflow specification model is represented according to the following template.

```

/** State of the World **/
SIS: set of predicates
DIS: set of predicates

/** Goal **/
TriggerCondition: formula
FinalState: formula

/**Norm-oriented Abstract Services Set **/
ServiceName
Preconditions: formula
Postconditions: formula
QoS Types: set of predicates
...
/**Normed-QoS Set **/
NormID
NormType: deontic operator
Services: list of N-AS
RuledCondition: formula (QoS_Type(QoS_Value)
                AND other conditions)
ApplicabilityCondition: formula
Priority: priority value
...
```

Listing 1: Template of the specification model for a Norm-oriented Abstract Workflow.

It is worth noting that the definitions mentioned above imply a formalisation of a *sphere of knowledge* in which the system operates. In particular, to establish a relationship between the formal specification of goals, the state of the world and the formal representation of the normative framework, it is necessary to refer to the same semantic layer. This requirement can be satisfied, for example, by adopting a knowledge formalisation based on a proper domain ontology. It is out of the scope of the paper how to build such an ontology.

## 5.3 Algorithms for customised concrete plans generation

The *N-AW* is essentially a representation of an assistive task through a set of *N-AS* and *N-QoS* that describe the functionalities and the constraints for producing customised results for target users. Thus, it is not executable, but it can be planned in a variety of ways, so producing a set of customised concrete plans. They are composed of QoS-based Concrete Services provided by different Service Providers that are compliant with the prescribed normed-QoS (see Fig. 2), by including the normative reasoning shown in Algorithm 1.

Given an *N-AW* model, Algorithm 1 firstly generates a sequence of *N-AS*, (named *tempSeq*), that satisfies the goal of the *N-AW*, starting from a state of the world $\mathcal{W}^t$, and constrained by the $N\text{-}QoS_{Set}$. This initial sequence is obtained by means of Algorithm 2 that, from an initial state of the world, reaches a final state of the world containing the goal throughout intermediate states that do not violate the prescribed $N\text{-}QoS_{set}$.

**Algorithm 1:** Customised_Concrete_Plan Generation

```
Data: $N\text{-}AW$
Result: $Customised\ Concrete\ Plan\ (CCP)$
$\langle \mathcal{W}^t, \mathcal{G}, N\text{-}AS_{set}, \mathcal{N}\text{-}QoS_{set} \rangle \leftarrow N\text{-}AW$;
$Plan \leftarrow \varnothing$;
$goal\_states \leftarrow \varnothing$;
$\langle t_c, f_s \rangle \leftarrow \mathcal{G}$;
$result \leftarrow \langle Type, Plan \rangle$;
$goal\_states.append(f_s)$;
$tempSeq \leftarrow norm\_aware\_Planner(N\text{-}AS_{Set}, \mathcal{N}\text{-}QoS_{Set}, \mathcal{W}^t, goal\_states, Plan)$;
 $result \leftarrow find\_concrete\_services(tempSeq)$;
switch $result.Type$ do
    case $Result_1$ do
        $Customised\_Concrete\_Plan \leftarrow result.Plan$;
        break;
    case $Result_2$ do
        $Customised\_Concrete\_Plan \leftarrow null$;
        break;
    case $Result_3$ do
        $\mathcal{N}'_{QoS} \leftarrow Relax\_norm(\mathcal{N}\text{-}QoS_{Set})$;
        $N\text{-}AW' \leftarrow \langle \mathcal{W}^t, \mathcal{G}, N\text{-}AS_{Set}, \mathcal{N}'\text{-}QoS_{Set} \rangle$;
        $Customised\_Plan\_Generation(N\text{-}AW')$;
        break;
for $cs \in Customised\_Concrete\_Plan$ do
    $as \leftarrow get\_abstService(cs)$;
    $\mathcal{W}^E \leftarrow \{\mathcal{W}^t \cup as.postCond \cup as.QoSs\}$;
    $execute(cs)$;
    $\mathcal{W}^t \leftarrow update(\mathcal{W}^t)$;
    if $\mathcal{W}^t \not\equiv \mathcal{W}^E$ then
        $N\text{-}AW' \leftarrow \langle \mathcal{W}^t, \mathcal{G}, N\text{-}AS_{Set}, \mathcal{N}\text{-}QoS_{Set} \rangle$;
        $Customised\_Plan\_Generation(N\text{-}AW')$;
        break;
```

For evaluating if a state of the world, generated by choosing a particular *N-AS*, is violating a normed-QoS, the following condition has to be verified:

$$effect_{N-AS} \wedge d(n_1.\rho_1) \wedge \ldots \wedge d(n_m.\rho_m) \neq \perp \tag{12}$$

where $effect_{N-AS}$ expresses the result of the post-condition of the *N-AS* along with the provided *QoS* on the state of the world. The condition (12) verifies that, by applying the deontic operator of the applicable *N-QoS*, it does not result in a logical contradiction. If no contradiction occurs, the corresponding *N-AS* can be chosen, and Algorithm 2 can proceed to explore new sub-goals, i.e., the preconditions of the *N-AS* previously chosen.

**Algorithm 2:** Norm_Aware_Planner

```
Data: $N$-$AS_{Set}, \mathcal{N}_{QoS}, \mathcal{W}, goal\_states, CurrentPlan$
Result: $tempSeq$
$tempSeq \leftarrow \varnothing$;
if $length(goal\_states) = 0$ then
    return $Plan$;
$i \leftarrow 0$;
while $i < length(goals)$ do
    $current\_goal_state \leftarrow goal\_stets.get(i)$;
    if $achieved(current\_goal_state, \mathcal{W})$ then
        $i \leftarrow i + 1$;
        continue;
    if $unification(currentGoal, \mathcal{W})$ then
        $i \leftarrow i + 1$;
        continue;
    $List\_candidate\_N\text{-}AS \leftarrow \{as \mid as \in N\text{-}AS\} \wedge isRelevant(as.postCond, current\_goal\_state)\}$;
    $List\_applicable\_Normed - QoS \leftarrow \{n \mid n \in \mathcal{N}\text{-}QoS_{Set} \wedge isSatisfied(\phi_n, \mathcal{W})\}$;

    $m \leftarrow length(List\_applicable\_Normed - QoS)$;
    for $as \in List\_candidate\_N\text{-}AS$ do
        $effect_{N-AS} = as.postCond \wedge as.QoSs$;
        if $effect_{N-AS} \wedge d(\rho_{n_1}) \wedge d(\rho_{n_2}) ... \wedge d(\rho_{n_m}) \neq \perp$ then
            $as\_chosen \leftarrow as$;
            $subgoals \leftarrow list(as\_chosen.preCond)$;
            $CurrentPlan \leftarrow CurrentPlan.append(asChoosen)$
            $\mathcal{W}' \leftarrow \mathcal{W}.include(as\_chosen.postCond)$;
            $solution = norm\_aware\_Planner(N\text{-}AS_{Set}, \mathcal{N}\text{-}QoS_{Set}, \mathcal{W}', subgoals, currentPlan)$;

            if $solution = Null$ then
                $remove(as\_chosen, currentPlan)$;
                continue;
            else
                $tempSeq.append(solution)$;
                $tempSeq.append(aschosen)$;
                $Found \leftarrow True$;
    if $\neg Found$ then
        return $Null$
```

Once the sequence of *N-AS* is obtained, Algorithm 1 tries to find a set of corresponding QoS-based Concrete Services, thus generating a customised concrete plan for a target user and/or a domain-specific context.[1]

Hence, three situations may occur. The first case is that a customised concrete plan is found, i.e. for each *N-AS* of *tempSeq*, a Service Provider that supplies a corresponding *Q-CS* with the prescribed QoS value is available. The second case is verified when there are no *Q-CS* that realise one or more *N-AS* of *tempSeq*. The last case is verified when it is impossible to generate a customised concrete plan in compliance with the set of normed-QoS. Even though only obligations and prohibitions that have a restrictive role are included as normed-QoS, priorities are associated with each normed-QoS to determine the ones that are more restrictive than others. So, when it is impossible to generate a customised concrete plan, a recovery mechanism is adopted. In consists in finding an alternative solution by relaxing the constraints imposed by the normed-QoS set starting with the ones with the lowest priority. However, there are some normed-QoS that cannot be relaxed because they are hard constraints for the target user or the domain context. For instance, the prohibition of a visual interaction modality is a hard constraint for visually impaired people. Thus, it is fundamental for this category of people to be assisted by employing technological devices that do not use a visual interaction modality. For this reason, it is imposed by the design that an appropriate priority value is used to indicate that the normed-QoS is not releasable.

After the generation of the customised concrete plan, its QoS-based Concrete Services are executed. The algorithm checks if some unexpected events occurred in the environment by evaluating if the execution of a *Q-CS* has generated an effect on the dynamic state of the world invalidating the proceeding of the execution of the plan. In this case, the algorithm re-plans the customised concrete plan starting from that *Q-CS*, thus adapting the customised concrete plan to the new state of the world.

## 5.4 Steps of the methodological approach

The proposed methodological approach can be summarised with a six steps process, as illustrated in Fig. 3. It consists of creating customised concrete plans starting from a graph representation of an Abstract Workflow, generating a Norm-oriented Abstract Workflow model, and applying the proposed algorithms. In the following are illustrated the six steps.

1. *Norm-oriented Abstract Workflow initialisation:* It consists in identifying the goal of the assistive task and the target users;
2. *Norm-oriented Abstract Service set definition:* It is an iterative procedure to be applied for each node of the initial Abstract Workflow consisting in: *(i)* defining variables for each node that specify the elements that the node handles, *(ii)* specifying appropriate propositions for the endpoint of the arrow in-coming to the node and for the start-point of the arrow out-coming from the node that represent the preconditions and the post-

[1] In this work, the descriptions of the QoS-based Concrete Services available in the environment are stored in a repository where each Service Provider has to be registered.

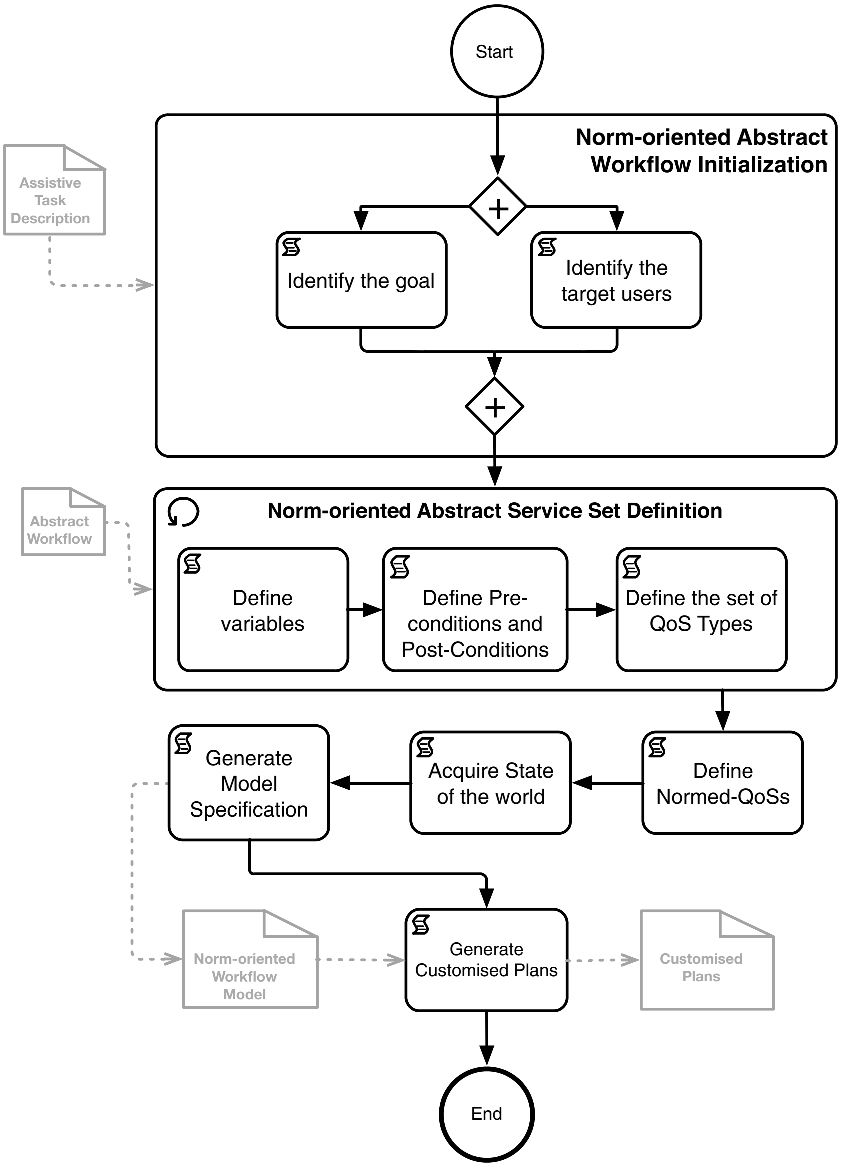

Start
Norm-oriented Abstract Workflow Initialization
Assistive Task Description
Identify the goal
Identify the target users
Norm-oriented Abstract Service Set Definition
Abstract Workflow
Define variables
Define Pre-conditions and Post-Conditions
Define the set of QoS Types
Generate Model Specification
Acquire State of the world
Define Normed-QoSs
Norm-oriented Workflow Model
Generate Customised Plans
Customised Plans
End

**Fig. 3** Steps of the proposed methodological approach

conditions of the corresponding Norm-oriented Abstract Service respectively, *(iii)* and finally associating the set of QoS types that are considered;

3. *Define normed-QoS:* It consists in defining QoS types, and QoS values according to the target users, using the deontic operators (i.e., Option, Prohibition or Obligation), and in assigning a priority to the *Normed-QoS*;
4. *Acquire state of the world:* It consists in identifying the SIS related to a user or a class of users and acquire the DIS;
5. *Generate model specification:* It consists in generating the Norm-oriented Abstract Workflow model starting from to the Norm-oriented Abstract Service set, the goal, the state of the world, and the normed-QoS;
6. *Generate customised concrete plan:* It consists in applying the algorithms presented in Sect. 5.3 to the Norm-oriented Abstract Workflow model.

## 6 Application scenario

In this section, we show an application scenario based on the previous example of the Remind Medicine assistive task, in order to describe the entire process underlying the proposed methodology. The system is implemented in Python for its portability and its easy integration with platforms composed of heterogeneous devices including robotic systems. In addition, compared to other high-level programming languages, it allows fast prototyping and it is one among the

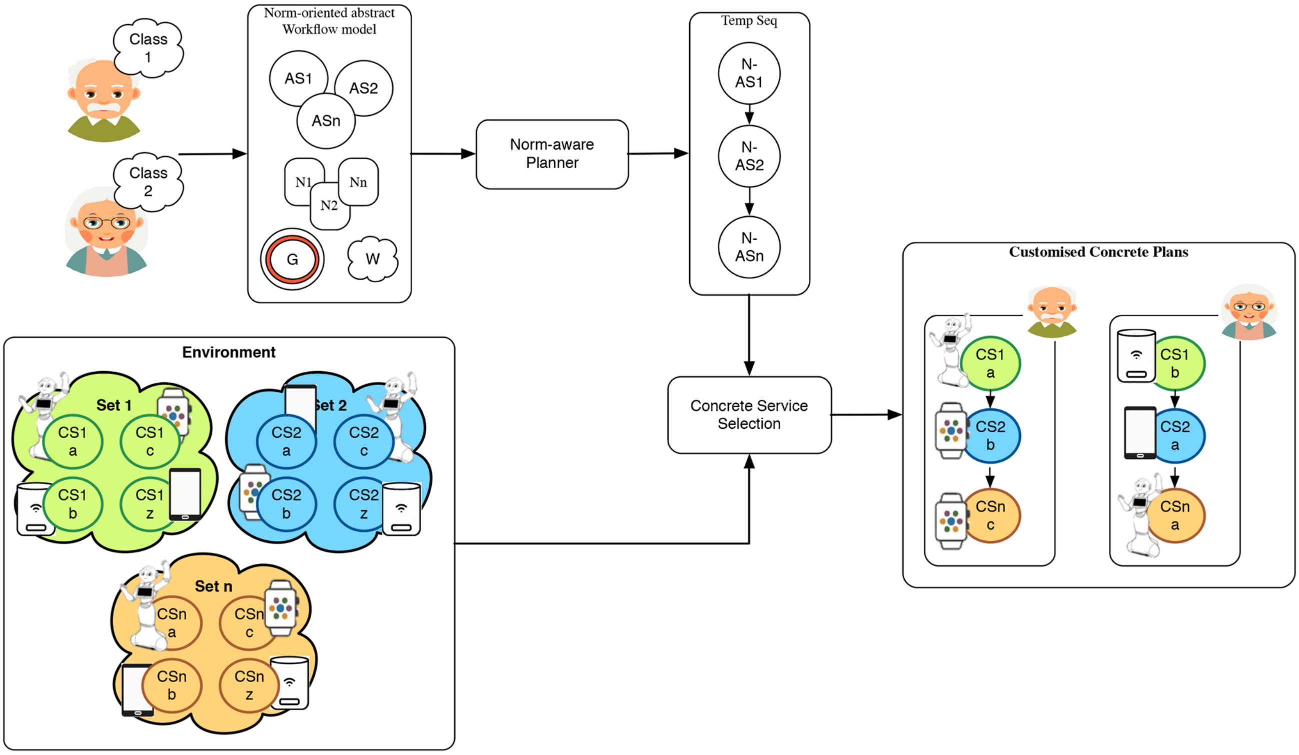


**Fig. 4** Functional schema of the proposed approach

most popular and efficient programming languages for IoT. The repository of concrete services is stored in an SQLite database.

Let us suppose that in a healthcare home, there are several guests with different characteristics (i.e., diseases, psychological traits, gender, age). Let us also assume that the healthcare home is endowed with a plurality of devices such as robots, digital assistants, monitors that can provide different concrete services. Guests are equipped with smartphones, smart-watches or other kinds of personal digital devices. Each guest has to follow a pharmacological therapy that is a plan of medicines that have to be taken at specific hours of the day.

In this application scenario, a useful assistive task for the healthcare home guests is to remind them to take medicines. Such an assistive task can be represented by the graph illustrated in Fig. 1. Because of the different characteristics of guests, the subtasks of the assistive task should be provided by concrete services with QoS values that are more suitable to each guest's profile, so increasing user satisfaction and acceptance of the assistance employing technological devices.

Figure 4 shows the functional schema of the proposed approach to produce concrete plans that are customised according to a particular user or a category of users starting from the Norm-oriented Abstract Workflow specification. Hence, the Norm-oriented Abstract Workflow model is produced following the steps illustrated in Sect. 5.4.

Firstly, to define the goal of the *N-AW*, we have to identify the trigger condition and the final state that it has to produce. As said above, a medicine usually has to be taken at the hour of the day established in the pharmacological therapy. Thus, the precondition of the goal can be identified by the proposition `IsTime(T, User, Medicine)` where `T` is a variable indicating the hour of the day, `User` is a variable for identifying the guest of the healthcare home, and `Medicine` is a variable for identifying the specific medicine the user has to take.

The final state of the goal can be identified by the proposition `Remind(User, Medicine, completed)`. It indicates that the task to remind a `User` to take her/his `Medicine` at time `T` is completed.[2]

[2] It is worth noting that the choice of the appropriate propositions to adopt to identify the elements of the model comes from an opportune ontological definition of the application domain that, as previously said, is out of the scope of this paper.

$$G = \langle \mathtt{IsTime(T, User, Medicine), Remind(User, Medicine, completed)} \rangle$$

The second step is the identification of the target user. According to the domain, the target users of the assistive task are the guests of the healthcare home. In this example, let us suppose that there are both visually impaired and irritable guests.

Then, for each node of the *AW* shown in Fig. 1, we identify the essential elements for defining the corresponding Norm-oriented Abstract Service. Let us consider the node `FindUser`. Its functionality is to identify user localization. Hence, a variable associated with such service is a `User` identification key. The preconditions of such service can be `IsTime(T, User, Medicine)` and `Guest(User)`, meaning that the user has to be found if (s)he is a guest of the healthcare home, and it is the time to take her/his medicine. The postcondition of the service is that the user has been found. According to the first-order logic, it can be expressed with the formula `Found(User)`.

Finally, the set of QoS types that we consider for this assistive task is composed of: the degree of interference, meaning the disturb that the adoption of some device may cause to the guest; the level of interaction required, i.e., the level of intrusiveness; the interaction modality, e.g., dialogue, text; the time the service may spend for completing its execution; and finally the amount of energy a service requires to execute. So, we consider QoS attributes that are strictly related with the target users, and two QoS attributes (the execution time and the energy consumption) that could be of interest for the administration of the healthcare home.

Hence, the Norm-oriented Abstract Service can be represented as follows:

```
FindUser(User)
Preconditions: IsTime(T,User,Medicine),
               Guest(User)
Postconditions: Found(User)
QualityOfServices:
InterferenceLevel(InterfLev1),
InteractionLevel(InteractLev1),
InteractionModality(InteractMod1),
ExecutionTime(ExecTime1),
EnergyWaste(EWLev1)
```

We assume that irritable users need that assistive tasks are delivered with a low level of interaction to limit their adverse reactions. While visually impaired users need that assistive tasks are delivered only through a dialogue modality and not a textual one. These assumptions are translated into norms having the highest priority. So, for irritable users, an obligation imposes a low level of interaction, while for visually impaired users a prohibition imposes a visual modality.[3]

Moreover, we consider that it is obligatory to save energy consumption during the day because of its increased cost. So, services with high energy consumption should be chosen in the evening to save costs.

For the sake of simplicity, in the current release of the software, normed-QoS labelled with *Priority (10)* are considered by the system hard constraints that can not be releasable.

```
Normed-QoS(id1)
NormType: Prohibition
Services: All
QoSs: InteractionModality(visual)
ApplicabilityCond: IsUser(visually-impaired)
Priority: Priority(10)
```

```
Normed-QoS(id2)
NormType: Obligation
Services: All
QoSs: InteractionLevel(low)
ApplicabilityCond: IsUser(irritable)
Priority: Priority(10)
```

```
Normed-QoS(id3)
NormType: Obligation
Services: All
QoSs: EnergyWaste(low)
ApplicabilityCond: Timeslot(daytime)
Priority: Priority(3)
```

Moreover, we also assume that for irritable users, interactive services may generate discomfort if they take a long time to execute. So, short running services should be preferred. For this reason, another norm is introduced describing this situation (normed-QoS(id4)). Conversely, for

[3] As previously said, the norms have to be defined by a domain expert. The normed-QoS set defined in this application scenario has to be considered as an example.

visually impaired guests, the execution time of service is not a concern (normed-QoS(id5)).

```
Normed-QoS(id4)
NormType: Obligation
Services: All
QoSs: ExecutionTime(low)
ApplicabilityCond: IsUser(irritable)
Priority: Priority(7)
```

```
Normed-QoS(id5)
NormType: Option
Services: All
QoSs: ExecutionTime(low)
ApplicabilityCond: IsUser(visually-impaired)
Priority: Priority(0)
```

To create assistive plans for irritable and visually impaired guests, the static information state is respectively:

```
SIS: Guest(User), IsUser(irritable)
```

```
SIS: Guest(User), IsUser(visually-impaired)
```

meaning that in the healthcare home there are both irritable and visually impaired guests.

The Norm-oriented Abstract Workflow model defined for the application scenario to generate Concrete Customised Plans for irritable users is the following:

```

/** State of the world **/
SIS: Guest(User), IsUser(irritable)
DIS: IsTime(T, User, Medicine),
      Timeslot(daytime)

/** Goal **/
TriggerCond:IsTime(T, User, Medicine)
FinalState:Remind(User, Medicine, completed)

/** Norm-oriented Abstract Services Set **/

FindUser(User)
Precond:IsTime (T, User , Medicine),
          Guest(User)
Postcond:Found(User) OR Failure(User,
            Medicine, not_found)
QualityOfServices:
        InterferenceLevel(InterfLev1),
        InteractionLevel(InteractLev1),
        InteractionModality(InteractMod1),
        ExecutionTime(ExecTime1),
        EnergyWaste(EWLev1)

Check_User_Physical_State(User)
Precond: Found(User)
Postcond:Is_Physical_State(User, well)
      OR(Is_Physical_State(User, bad)
      AND Failure(User, Medicine, not_healthy))
QualityOfServices:
        InterferenceLevel(InterfLev2),
        InteractionLevel(InteractLev2),
        InteractionModality(InteractMod2),
        ExecutionTime(ExecTime2),
        EnergyWaste(EWLev2)
Check_Taken_Medicine(User, Medicine)
Precond: Is_Physical_State(User, well)
      OR Done_Remind(User, Medicine)
Postcond: Took(User, Medicine)
      OR NOT Taken(User, Medicine)
QualityOfServices:
        InterferenceLevel(InterfLev3),
        InteractionLevel(InteractLev3),
        InteractionModality(InteractMod3),
        ExecutionTime(ExecTime3),
        EnergyWaste(EWLev3)

Check_Time_In(User, Medicine)
Precond: NOT Taken(User, Medicine)
Postcond: In_Time(User, Medicine, yes)
      OR (In_Time(User, Medicine, no)
      AND  Failure(User, Medicine, timeout))
QualityOfServices:
        InterferenceLevel(InterfLev5),
        InteractionLevel(InteractLev5),
        InteractionModality(InteractMod5),
        ExecutionTime(ExecTime5),
        EnergyWaste(EWLev5)

Remind_medicine(User, Medicine)
Precond:  In_Time(User, Medicine, yes)
Postcond: Done_Remind(User, Medicine)
QualityOfServices:
        InterferenceLevel(InterfLev6),
        InteractionLevel(InteractLev6),
        InteractionModality(InteractMod6),
        ExecutionTime(ExecTime6),
        EnergyWaste(EWLev6)

Alert(User, Medicine, Msg)
Precond:  Failure(User, Medicine, Msg)
Postcond: Anomaly(User, Medicine, Msg)
      AND Remind(User, Medicine, completed)
QualityOfServices:
        InterferenceLevel(InterfLev8),
        InteractionLevel(InteractLev8),
        InteractionModality(InteractMod8),
        ExecutionTime(ExecTime8),
        EnergyWaste(EWLev8)

Terminate_Process(User, Medicine)
Precond: Took(User, Medicine)
Postcond: Remind(User, Medicine, completed)

/** Normed-QoS Set**/

Normed_QoS(id1)
NormType: Prohibition
Services: All
QoSs: InteractionModality(visual)
ApplicabilityCond: IsUser(visually-impaired)
Priority: Priority(10)

Normed_QoS(id2)
NormType: Obligation
Services: All
QoSs: InteractionLevel(low)
ApplicabilityCond: IsUser(irritable)
Priority: Priority(10)

Normed_QoS(id3)
NormType: Obligation
Services: All
QoSs: EnergyWaste(low)
ApplicabilityCond: Timeslot(daytime)
Priority: Priority(3)

Normed_QoS(id4)
NormType: Obligation
Services:  All
QoSs: ExecutionTime(low)
ApplicabilityCond: IsUser(irritable)
Priority: Priority(7)

Normed_QoS(id5)
NormType: Option
Services: All
QoSs: ExecutionTime(low)
ApplicabilityCond: IsUser(visually-impaired)
Priority: Priority(0)

```

Listing 2: N-AW Specification Model

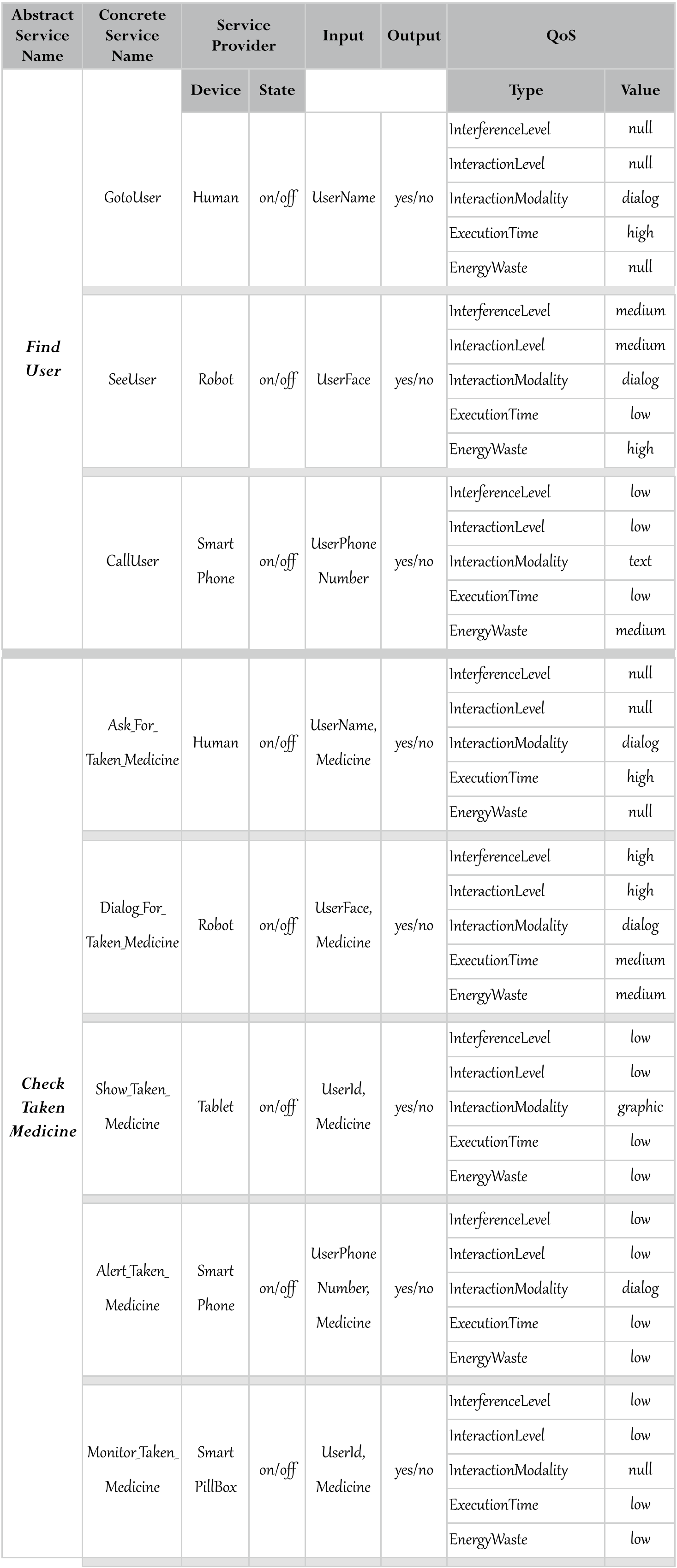


**Fig. 5** Concrete services repository

| Abstract Service Name | Concrete Service Name | Service Provider | | Input | Output | QoS | |
|---|---|---|---|---|---|---|---|
| | | Device | State | | | Type | Value |
| ***Find User*** | GotoUser | Human | on/off | UserName | yes/no | InterferenceLevel | null |
| | | | | | | InteractionLevel | null |
| | | | | | | InteractionModality | dialog |
| | | | | | | ExecutionTime | high |
| | | | | | | EnergyWaste | null |
| | SeeUser | Robot | on/off | UserFace | yes/no | InterferenceLevel | medium |
| | | | | | | InteractionLevel | medium |
| | | | | | | InteractionModality | dialog |
| | | | | | | ExecutionTime | low |
| | | | | | | EnergyWaste | high |
| | CallUser | Smart Phone | on/off | UserPhone Number | yes/no | InterferenceLevel | low |
| | | | | | | InteractionLevel | low |
| | | | | | | InteractionModality | text |
| | | | | | | ExecutionTime | low |
| | | | | | | EnergyWaste | medium |
| ***Check Taken Medicine*** | Ask_For_ Taken_Medicine | Human | on/off | UserName, Medicine | yes/no | InterferenceLevel | null |
| | | | | | | InteractionLevel | null |
| | | | | | | InteractionModality | dialog |
| | | | | | | ExecutionTime | high |
| | | | | | | EnergyWaste | null |
| | Dialog_For_ Taken_Medicine | Robot | on/off | UserFace, Medicine | yes/no | InterferenceLevel | high |
| | | | | | | InteractionLevel | high |
| | | | | | | InteractionModality | dialog |
| | | | | | | ExecutionTime | medium |
| | | | | | | EnergyWaste | medium |
| | Show_Taken_ Medicine | Tablet | on/off | UserId, Medicine | yes/no | InterferenceLevel | low |
| | | | | | | InteractionLevel | low |
| | | | | | | InteractionModality | graphic |
| | | | | | | ExecutionTime | low |
| | | | | | | EnergyWaste | low |
| | Alert_Taken_ Medicine | Smart Phone | on/off | UserPhone Number, Medicine | yes/no | InterferenceLevel | low |
| | | | | | | InteractionLevel | low |
| | | | | | | InteractionModality | dialog |
| | | | | | | ExecutionTime | low |
| | | | | | | EnergyWaste | low |
| | Monitor_Taken_ Medicine | Smart PillBox | on/off | UserId, Medicine | yes/no | InterferenceLevel | low |
| | | | | | | InteractionLevel | low |
| | | | | | | InteractionModality | null |
| | | | | | | ExecutionTime | low |
| | | | | | | EnergyWaste | low |

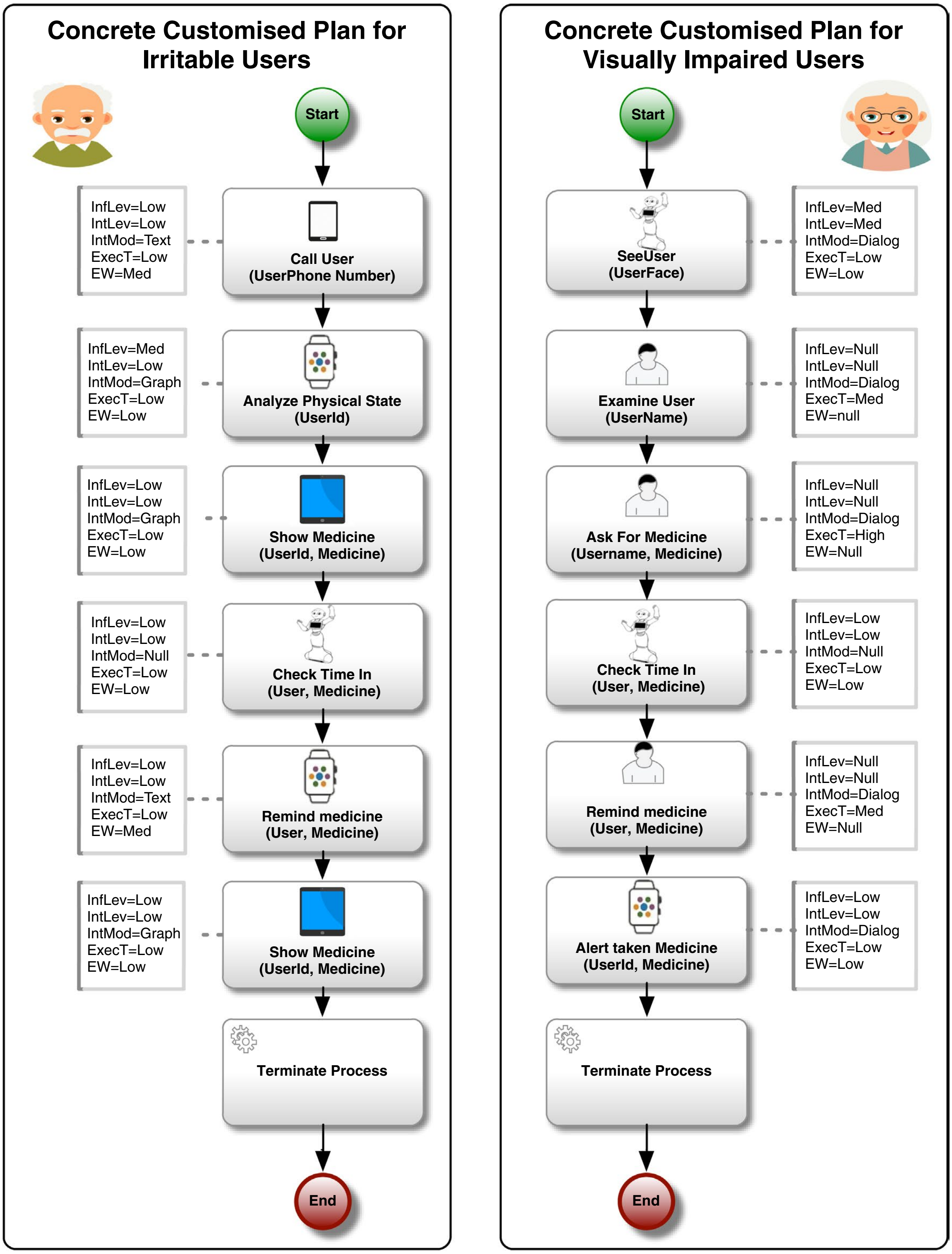


**Fig. 6** Customised concrete plans for irritable and visually impaired users

Finally, in order to apply the proposed algorithms, a set of QoS-based Concrete Services should be available in the healthcare home. We consider twenty-seven QoS-based Abstract Services whose descriptions have been stored in an appropriate service repository. They are grouped into different Abstract Services according to the specific functionality they provide. The table in Fig. 5 shows an excerpt of such a repository. For a specific Abstract Service, there are multiple

QoS-based Concrete Services which are able to fulfil the postconditions of the corresponding Norm-oriented Abstract Service in different ways, as shown in the 2nd column. For each QoS-based Concrete Service, the corresponding Service Provider (i.e., the device) and its availability (i.e., the state) is reported respectively in the 4th and the 5th columns of the table of Fig. 5. The input and output sets of the QoS-based Concrete Services referring to same functionality may be different. Let us consider the `FindUser` Abstract Service. It can be provided by three QoS-based Concrete Services, differing from each other in the way they accomplish the task, and in the input sets. For example, `SeeUser` is provided by a robot, and it needs to know the user's face to find her/him in the healthcare home. While `CallUser` is provided by a smart-phone, and it needs the number of the user's phone. Finally, `GotoUser` may also be provided by a human that needs to know the name of the user to be found. Conversely, the outputs of these concrete services are, in this case, the same.

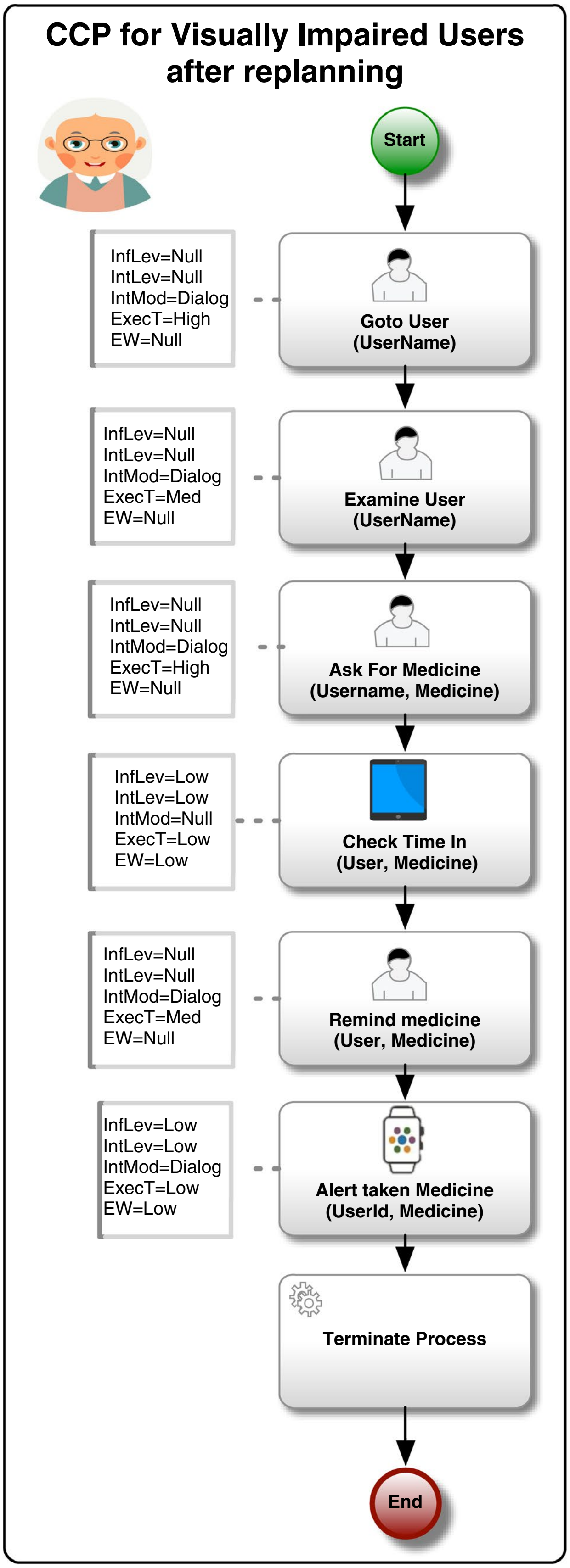
It is worth noting that for the same Abstract Service, the corresponding QoS-based Concrete Services may have different QoS values, as shown in the 6th column of the table of Fig. 5. In addition, for some of these services, QoS values cannot be defined, because not applicable to a specific device, so a *null* value is provided.

For the sake of simplicity, in this example we consider only five QoS Types, namely *InterferenceLevel*, *InteractionLevel*, *InteractionModality*, *ExecutionTime*, and *EnergyWaste*. Each *Q-CS* is associated with different values of these five parameters. For example, the robot and the smartphone provide the same functionality through two different QoS-based Concrete Services, `SeeUser` and `CallUser` that have different QoS values. The first one interacts with the user through dialogue, and it has a medium level of interaction and a medium level of interference. Conversely, the second one finds the user sending a text message to her/his smart-phone. Thus, it provides lower values for the level of interaction and the level of interference with respect to the previous one. They are also characterised by different energy consumption.

Hence, the algorithms applied to the previous Norm-oriented Abstract Workflow model, with the available QoS-based Concrete Services, produce two different customised concrete plans for irritable and visually impaired users respectively (see Fig. 6).

– *Customised concrete plan for irritable guests* It consists of seven *Q-CS*. A smart-phone provides the first service to find the guest with low values of interference level, interaction level and execution time by using a textual interaction modality and a medium waste of energy. A smart-watch provides both the second service to check the physical state of the guest, and the third one to remind the guest her/his medicine with the same value of interaction level and execution time, but different interaction modality (graphical and textual) and energy waste. A tablet provides the service to check if the guest has taken the medicine with its corresponding QoS values, and it is performed twice, before and after reminding the medicine. Finally, the last one is a system service to terminate the process. Such a plan has been obtained by relaxing the normed-QoS(id3) related to energy waste. In fact, there are not QoS-based Concrete Services that satisfy all the obligatory normed-QoS. Thus, the system relaxes the normed-QoS with the lowest priority among the obligatory ones applicable to irritable users and in the domain context. As we can see, by relaxing the normed-QoS(id3), the algorithm may choose `CallUser` that has a medium value of energy waste, and the `Remind Medicine` provided by the smart-phone, thus maintaining the compliance with the highest priority norms for an irritable user (normed-QoS(id2) and normed-QoS(id4)). For space concerns, some service names have been abbreviated in the reported solution.

**Fig. 7** Concrete customised plans for visually impaired users after the ▶ replanning phase



```
Start
-> CallUser(UserNumber) by SmartPhone
 with InterferenceLevel=low
 and InteractionLevel=low
 and InteractionModality=text
 and ExecutionTime=low
 and EnergyWaste=medium
-> Analyze_Phys_State(UserID) by SmartWatch
 with InterferenceLevel=medium
 and InteractionLevel=low
 and InteractionModality=graphic
 and ExecutionTime=low
 and EnergyWaste=low
-> Show_Med_Taken(UserID,Medicine) by Tablet
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=graphic
and ExecutionTime=low
and EnergyWaste=low
-> Check_Time_In(User, Medicine) by Robot
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=null
and ExecutionTime=low
and EnergyWaste=low
-> Remind_med(User, Medicine) by SmartWatch
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=text
and ExecutionTime=low
and EnergyWaste=med
-> Show_Med_Taken(UserID,Medicine) by Tablet
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=graphic
and ExecutionTime=low
and EnergyWaste=low
 -> ProcessFinish(None) by Orchestrator

Solution found by relaxing Normed-QoS(id3)
```

– *Customised concrete plan for visually impaired guests* It consists of the same number of QoS-based Concrete Services as the previous one, but three services are provided by a human because there are not devices that can provide dialogue modality with a low value of energy waste, as required by the norms for this class of users. It should be noted that the preferred level of execution time cannot be obtained in the case of the services provided by the human since this QoS type is not applicable to humans. Conversely, desired values of execution time are satisfied by the remaining QoS-based Concrete Services found by the algorithms.

```
Start
-> SeeUser(UserFace) by Robot
with InterferenceLevel=medium
and InteractionLevel=medium
and InteractionModality=dialog
and ExecutionTime=low
and EnergyWaste=low
-> Examine_User(UserName) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=medium
and EnergyWaste=null
-> Ask_Taken_Med(UserName, Medicine) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=high
and EnergyWaste=null
-> Check_Time_In(User, Medicine) by Robot
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=null
and ExecutionTime=low
and EnergyWaste=low
-> Remind_med(User, Medicine) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=medium
and EnergyWaste=null
-> Alert_Taken_Med(UserPhN,Med) by SmartPhone
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=dialog
and ExecutionTime=low
and EnergyWaste=low
-> ProcessFinish(None) by Orchestrator
```

It should be noted that starting from the same set of QoS-based Concrete Services and the same required assistive task, two different customised concrete plans are generated for two different classes of users with different preferences and needs.

## 6.1 The re-planning phase

The generated plans depend both on the user preferences and needs, and on the environment in which the system operates. The environment may change with time because of some dynamic circumstances may take place, such as the unavailability of some devices because of a fault. When a dynamic circumstance occurs, the dynamic state of the world changes accordingly, since it is updated with variable information gathered from time to time (the update frequency of the dynamic state of the world is fixed in advance). In such a case, the framework can find an alternative plan. In case of a device fault, the change of the dynamic state of the world results in a change of the state of the involved device, so the algorithm replaces the unavailable device with another one that provides the same functionality still compatible with the target users. So, the algorithm re-plans the customised concrete plan starting from that point of failure where the service replacement was necessary. For example, if the robot is out of work, the system produces, for visually impaired users, the following alternative plan by replacing the concrete services provided by the robot with the ones provided by the human and the tablet. In Fig. 7 its graphical representation is also shown.

**Table 1** Patients characteristics

| Patients | Memory Level | Neuroticism Level | Schooling Level |
|---|---|---|---|
| #1 | Medium | Low | Medium |
| #2 | Medium | High | Low |
| #3 | High | High | Medium |
| #4 | High | High | Medium |
| #5 | High | Low | High |
| #6 | High | Low | Medium |
| #7 | High | Low | Medium |

```
Start
-> GotoUser(UserName) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=high
and EnergyWaste=null
-> Examine_User(UserName) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=medium
and EnergyWaste=null
-> Ask_Taken_Med(UserName, Medicine) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=high
and EnergyWaste=null
-> Check_Time_In(User, Medicine) by Tablet
with  InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=null
and ExecutionTime=low
and EnergyWaste=low
-> Remind_med(User, Medicine) by Human
with InterferenceLevel=null
and InteractionLevel=null
and InteractionModality=dialog
and ExecutionTime=medium
and EnergyWaste=null
-> Alert_Taken_Med(UserPN,Med) by SmartPhone
with InterferenceLevel=low
and InteractionLevel=low
and InteractionModality=dialog
and ExecutionTime=low
and EnergyWaste=low
-> ProcessFinish(None) by Orchestrator
```

**Table 2** Repetition frequency w.r.t. memory and neuroticism levels

| Memory Level | Neuroticism Level | |
|---|---|---|
| | Low | High |
| Low | 5 min | 10 min |
| Medium | 10 min | 20 min |
| High | 15 min | 30 min |

Such application scenario showed that the proposed methodology allows to dynamically generate personalised assistive tasks, according to the prescribed norms. Furthermore, if a failure occurs in the generation of a customised plan compliant with all the prescribed norms, the framework allows to generate alternative plans in the case there are norms that can be relaxed. Finally, an adaptation mechanism is provided for the re-planning of assistive tasks when unforeseen events occur invalidating their execution.

## 7 An experimental validation

In order to provide an experimental validation of the proposed approach, the norm-based framework has been tested within the context of the project "User-centered Profiling and Adaptation for Socially Assistive Robotics" (UPA4SAR),[4] whose objective was to develop a robotic system to assist patients with neurological disorders, as dementia, for their daily activities at their homes (Di Napoli and Rossi 2019). In order to improve the level of acceptance of the new technology for a class of so fragile users, both the neurological and personality profile of patients were taken into account to personalise robotic behaviours. Personalisation consisted in adapting the interaction modalities with the robot according to the patient's characteristics and needs. The characteristics were extracted by the teams of neurologists and psychologists based on neurological and personality tests, respectively CDR (Morris 1993) and Neo-Pi3 (McCrae et al. 2005), and other personal information as the age and the schooling level.

The experimental phase of the project consisted in providing timely executed personalised assistive plans for a set of home patients through a social robot acting as an assistant. The robot executed supporting activities such as cognitive entertainment, reminding medicines, and checking daily living tasks (lunch, breakfast, and so on), without any remote control.

We provide an experimental validation of the proposed methodology by applying the framework on the set of real data of patients recruited for the UPA4SAR project. The

[4] A three-year research project funded by the Italian Ministry of Education, Universities, and Research—grant no. 2015KB-L78T.

**Table 3** Interaction modality w.r.t. schooling level

| Schooling Level | Interaction Modality |
|---|---|
| Low | Vocal |
| Medium | Textual/vocal |
| High | Textual/vocal |

rationale of this validation is to verify that the obtained customised concrete plans are the same assistive tasks, in terms of QoS values and devices, executed by the robotic system during the experiments carried out at the patients' homes for the remind medicine task.

For the validation, only the robot interaction modalities were considered as QoS parameters, without considering different devices since for the project experimentation only the robot was used as a service provider. Nevertheless, the framework generalises the personalisation of assistive plans also with respect to different service providers, since every personalisation dimension is uniformly treated as a norm.

The number of patients taking part in the experimental phase was seven. This small number depended on the difficulties to find people available for the experimentation, on the progression of their neurological disorder, and above all on the possibility to deploy the robotic systems in their homes (such as the presence of domestic animals, or carpets, and so on).

Nevertheless, the possibility to use realistic data was an added value for testing the norm-based approach here proposed, since it was possible to check whether the automatically customised concrete plans were compliant with the ones executed during the project experimentation.

The characteristics of patients considered for personalising the remind medicine task were:

- the level of memory split in three ranges Low, Medium, High corresponding respectively to not memory loss, moderate memory loss, and severe memory loss;
- the level of neuroticism personality factor split in two ranges Low, High corresponding respectively to individuals with more or less ability to cope with anxiety or to control reactions;
- the level of schooling split in three ranges Low, Medium, High based on the years of the Italian education system corresponding respectively to primary education (up to 5 years), secondary education (from 8 to 13 years), higher education (13+ years).

The considered QoS parameters for personalisation were the frequency at which the robot interacts with the patient, and the robot interaction modalities (textual or vocal) that have to be modulated according to the characteristics of the patients participating to the experimentation summarised in Table 1.

The teams of specialists that recruited and classified the patients suggested the personalisation criteria. In particular, they suggested that both the level of memory and neuroticism have to be taken into account to set the frequencies for repeating the task to remind medicine, as reported in Table 2.

In addition, they suggested that the type of interaction modality with the robot, textual or vocal, can be correlated to the level of schooling of the patient. In fact, for patients with a low instruction level, a vocal modality was considered mandatory since it is supposed they are not able to read fluently on a technological device. Conversely, both textual and vocal interaction modality were considered suitable for patients with a medium and high instruction level (see Table 3).

These suggestions were translated into a set of 8 norms for the assistive task of reminding medicines, as follows:

```
Normed-QoS(id1)
NormType: Obligation
Services: All
QoSs: RepetitionFrequency(5)
ApplicabilityCond: IsMemory(low)
                   AND IsNeuroticism(low)
Priority: Priority(10)
```

```
Normed-QoS(id2)
NormType: Option
Services: All
QoSs: RepetitionFrequency(10)
ApplicabilityCond: IsMemory(medium)
                   AND IsNeuroticism(low)
Priority: Priority(0)
```

```
Normed-QoS(id3)
NormType: Prohibited
Services: All
QoSs: RepetitionFrequency(5)
ApplicabilityCond: IsMemory(high)
                   AND IsNeuroticism(low)
Priority: Priority(10)
```

```
Normed-QoS(id4)
NormType: Obligation
Services: All
QoSs: RepetitionFrequency(10)
ApplicabilityCond: IsMemory(low)
                   AND IsNeuroticism(high)
Priority: Priority(10)
```

**Table 4** Execution time (ms) of the normative reasoner varying *N-AS* and *N-QoS*

| N-AS | N-QoS | | | | | |
|---|---|---|---|---|---|---|
| | 1 | 2 | 4 | 8 | 16 | 32 |
| 5 | 6 | 8.8 | 10.4 | 16 | 28.9 | 65.9 |
| 10 | 9.9 | 12.5 | 18 | 29.5 | 53.7 | 121.1 |
| 20 | 30.7 | 35.7 | 48.4 | 78.5 | 151.8 | 360.5 |
| 40 | 54.7 | 64.7 | 86.8 | 133.5 | 252.9 | 580.8 |
| 80 | 110.9 | 126.3 | 156.8 | 235.2 | 414.8 | 959.5 |

```
Normed-QoS(id5)
NormType: Obligation
Services: All
QoSs: RepetitionFrequency(20)
ApplicabilityCond: IsMemory(medium)
                   AND IsNeuroticism(high)
Priority: Priority(10)
```

```
Normed-QoS(id6)
NormType: Obligation
Services: All
QoSs: RepetitionFrequency(30)
ApplicabilityCond: IsMemory(high)
                   AND IsNeuroticism(high)
Priority: Priority(10)
```

```
Normed-QoS(id7)
NormType: Prohibition
Services: All
QoSs: InteractionModality(textual)
ApplicabilityCond: IsSchoolingLevel(low)
Priority: Priority(10)
```

```
Normed-QoS(id8)
NormType: Option
Services: All
QoSs: InteractionModality(Vocal)
ApplicabilityCond: IsSchoolingLevel(medium)
                   OR IsSchoolingLevel(high)
Priority: Priority(0)
```

It should be noted that norms apply to all those services providing the QoS parameters the norm refers to. These norms were injected into the framework together with the static information state containing the patients' profiles as follows:

```
SIS: IsPatient(Username), IsMemory(M_Level),
     IsSchoolingLevel(SC_Level),
     IsNeuroticism(N_Level)
```

where *M_Level*, *SC_Level* and *N_Level* refer respectively to the level of memory, schooling and neuroticism.

The framework automatically generated different customised concrete plans for the remind medicine task whose component robotic services (corresponding to the Abstract Services of Fig. 1) have QoS parameters values compliant with the suggestions reported in the Tables 2 and 3. This means that the generation of the customised concrete plans guarantees that personalisation is realised at the level of the single component service. The evaluation allowed to gather on the field validation of the proposed approach by verifying the framework adherence to the expected results. In fact, personalised assistive plans according to users' profiles are automatically generated without relying on ad-hoc designed personalisation for each category of users.

## 7.1 Normative reasoning scalability

To assess the robustness of the proposed approach, an experimental evaluation was carried out to analyse the performances of the normative reasoning. The robustness is

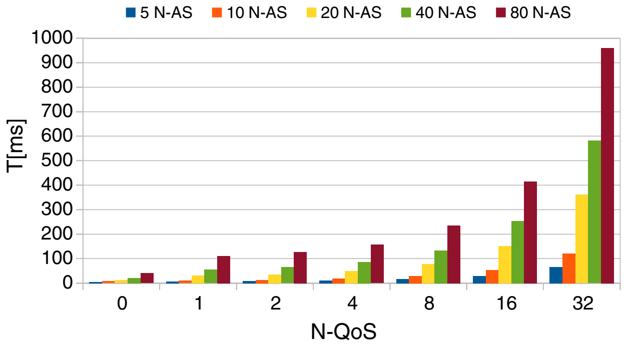


**Fig. 8** Histograms of the normative reasoner execution time

evaluated in terms of scalability of the normative reasoner algorithm by measuring its execution time, varying both the number of Norm-oriented Abstract Services and normed-QoS, with norms applied to all involved services.

For the scalability evaluation, only the execution time of the normative reasoner is considered since the time to bind and execute the concrete services heavily depends on the technology used by their developers. Since services can be provided by third parties, in accordance with the Service-Oriented approach, their performances cannot be included in this evaluation.

The results of the evaluation are reported in Table 4 and in Fig. 8 using a histogram representation.

They show that the execution time for cases with large numbers of *N-AS* and *N-QoS*, such as the case with 80 *N-AS* and 32 *N-QoS* considered in these experiments, is less than 1 second. So, limitation on the number of considered norms is not an issue.

Moreover, in real application scenarios, as in the case of the UPA4SAR project, it is unlikely that assistive tasks are composed of a large number of both services and personalisation dimensions. In fact, in the real case previously reported, the number of Abstract Services composing the assistive task to remind medicine and the number of norms used for personalisation were respectively 6 and 8.

## 8 Discussion and conclusions

The proposed methodological approach to automatically generate assitive plans customised to different classes of users, and to different ambient assisted living environments presents features that can be summarized as follows.

**Personalised QoS-based service selection**: Subjective and objective QoS types are considered, including the typical QoS service attributes that depend on providers, such as *Energy Waste* or *Execution Time*, and attributes related to different categories of users.

**Normed-based customisation**: The proposed approach allows assistive task customisation by decoupling the variable part of the model due to user personalisation, from the service layer, by means of normed quality of service. Hence, any change in the set of norms does not have an impact on the service layer.

**Reduced complexity**: The generation of a customised concrete plan based on QoS satisfaction is carried out starting from a high level of abstraction leading to a concrete one. The abstraction allows to group concrete services that provide equivalent functionalities in a single abstract service, so reducing the search space to find the component concrete services. In this way, a substantially smaller space of solutions is searched for, so making the approach more efficient compared with traditional ones that explore the whole space of concrete service for finding a solution.

**Flexibility**: The possibility to generate different alternatives to provide assistive tasks by including different types of norms allows addressing both the needs of different classes of users, and also the conditions of the specific execution context. The obligation and prohibition norms impose restrictions for categories of users and for domain contexts, while optional norms allows generating more alternative solutions.

**Adaptation**: The adaptation is guaranteed by the possibility to re-plan the customised concrete plan if some changes occur in the environment, by choosing different services that still comply with norms.

**Handling composition failure**: The proposed approach handles composition failure by providing mechanisms for relaxing QoS constraints when not suitable services are available that meet QoS requirements by releasing the corresponding norms.

**Simple deployment**: The proposed methodology can be easily deployed also in pre-existing AAL environments equipped with different technological devices because of the decoupling between abstraction of services and their actual provision. The effort lies on building the Norm-oriented Abstract Workflow specification model, and on specifying the endpoint of the services delivered by the service providers of the AAL environment.

Personalisation based on the characteristics and needs of a fragile class of users, such as elderly, is a fundamental requisite in order to make effective assistive care supported by modern technology in the context of Ambient assisted living.

Our most relevant contribution is a methodology for the generation of assistive tasks that can be deployed in any AAL environment, relying on the integration of a service-oriented approach and normative reasoning. The methodology allows dynamic customisation of assistive tasks depending on the user characteristics, and the devices available in the AAL environment. The novel idea grounding on the paper is to define the quality of services related to the functionality required to deliver an assistive task, in terms of norms that prescribe prohibited, obligatory or optional QoS values. In doing so, it is possible to apply a normative reasoning for the composition of QoS-based Concrete Services that realises an assistive task satisfying both the needs and preferences of classes of users, and environmental conditions. The possibility to decouple the functionality a service provides from how that functionality is delivered (i.e. its quality of service) allows to provide both user customisation, and adaptation to dynamic events. In fact, the composition of services necessary for a customised assistive task is realised reasoning on prescribed norms, without interfering with their functional composition.

In this work, we provide a formal specification of the methodology and a framework that implements it. Of course, knowledge formalisation of the domain under study is necessary, that requires human effort depending on the size of the domain. Such an effort can be mitigated by using automated tools for building ontologies. Instead, the most demanding effort is related to the creation of the Norm-oriented Abstract Workflow specification model that is currently manually generated.

The implemented framework is used in a working application scenario with more devices available as service providers. Moreover, it is tested on real patient data to provide an on field validation to verify its applicability on real application scenarios.

**Acknowledgements** The authors would like to thank the teams of the UPA4SAR project for providing the anonymous data of patients for the experimental validation.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

## References

Aggarwal R, Kunal Verma, Miller J, Milnor W (2004) Constraint driven web service composition in meteor-s. In: IEEE international conference on services computing, 2004. (SCC 2004). Proceedings. 2004, pp 23–30

Alchourrón CE (1969) Logic of norms and logic of normative propositions. Logique et analyse 12(47):242–268

Alrifai M, Risse T (2010) Efficient QoS-aware service composition. In: Binder W, Dustdar S (eds) Emerging web services technology, vol III. Birkhäuser, Basel, pp 75–87

Ardagna D, Mirandola R (2010) Per-flow optimal service selection for web services based processes. J Syst Softw 83(8):1512–1523

Ardagna D, Pernici B (2007) Adaptive service composition in flexible processes. IEEE Trans Softw Eng 33(6):369–384

Asghari P, Rahmani AM, Javadi HHS (2020) Privacy-aware cloud service composition based on QoS optimization in internet of things. J Ambient Intell Humaniz Comput 1–26

Canfora G, Di Penta M, Esposito R, Villani ML (2005) Qos-aware replanning of composite web services. In: Web Services, 2005. ICWS 2005. Proceedings. 2005 IEEE International Conference on, IEEE, pp 121–129

Chaib A, Boussebough I, Chaoui A (2018) Adaptive service composition in an ambient environment with a multi-agent system. J Ambient Intell Humaniz Comput 9(2):367–380

Cocchiarella NB (2015) Notes on deontic logic. Retrieved November 20

Di Napoli C (2009) Software agents to enable service composition through negotiation. Springer, Berlin, pp 275–296

Di Napoli C, Rossi S (2019) A layered architecture for socially assistive robotics as a service. In: 2019 IEEE international conference on systems, man and cybernetics (SMC), pp 352–357

Di Napoli C, Sabatucci L, Cossentino M, Rossi S (2017) Generating and instantiating abstract workflows with QoS user requirements. In: Proceedings of 9th international conference on agents and artificial intelligence, Springer, LNCS, pp 156–170

D'Mello DA, Ananthanarayana VS (2010) Dynamic selection mechanism for quality of service aware web services. Enterp Inf Syst 4(1):23–60

Hansen J (2008) Imperatives and deontic logic: on the semantic foundations of deontic logic, Ph.D. Thesis, University of Leipzig

Hong L, Hu J (2009) A multi-dimension QoS based local service selection model for service composition. JNW 4(5):351–358

Kokash N, D'Andrea V (2007) Evaluating quality of web services: a risk-driven approach. In: International conference on business information systems, Springer, pp 180–194

Kritikos K, Plexousakis D (2009) Requirements for QoS-based web service description and discovery. IEEE Trans Serv Comput 2(4):320–337

Lauraitis A, Maskeliūnas R, Damaševičius R, Połap D, Woźniak M (2019) A smartphone application for automated decision support in cognitive task based evaluation of central nervous system motor disorders. IEEE J Biomed Health Inform 23(5):1865–1876

Lin CF, Sheu RK, Chang YS, Yuan SM (2011) A relaxable service selection algorithm for QoS-based web service composition. Inf Softw Technol 53(12):1370–1381

Masdari M, Khezri H (2020) Service selection using fuzzy multi-criteria decision making: a comprehensive review. J Ambient Intell Humaniz Comput 1–32

McCrae RR, Costa PT Jr, Martin TA (2005) The neo-pi-3: a more readable revised neo personality inventory. J Personal Assess 84(3):261–270

McNamara P (2006) Deontic logic. In: Logic and the modalities in the twentieth century, vol 7. pp 197–288

Morris JC (1993) The clinical dementia rating (CDR). Neurology 43(11):2412

Muthuraman S, Venkatesan VP (2016) Qualitative and quantitative review of QoS based web services selection and composition techniques. In: Proceedings of the international conference on informatics and analytics, pp 1–12

Naseri A, Navimipour NJ (2019) A new agent-based method for QoS-aware cloud service composition using particle swarm optimization algorithm. J Ambient Intell Humaniz Comput 10(5):1851–1864

Ouadah A, Hadjali A, Nader F, Benouaret K (2019) SEFAP: an efficient approach for ranking skyline web services. J Ambient Intell Humaniz Comput 10(2):709–725

Papazoglou M, Traverso P, Dustdar S, Leymann F (2007) Service-oriented computing: state of the art and research challenges. IEEE Comput 40(11):38–45

Qi J, Xu B, Xue Y, Wang K, Sun Y (2018) Knowledge based differential evolution for cloud computing service composition. J Ambient Intell Humaniz Comput 9(3):565–574

Rao J, Su X (2004) A survey of automated web service composition methods. In: International workshop on semantic web services and web process composition, Springer, pp 43–54

Reiter R (1981) On closed world data bases. Readings in artificial intelligence. Elsevier, Amstredam, pp 119–140

Ribino P, Lodato C (2019) A norm compliance approach for open and goal-directed intelligent systems. Complexity

Ribino P, Bonomolo M, Lodato C, Vitale G (2020) A humanoid social robot based approach for indoor environment quality monitoring and well-being improvement. Int J Soc Robot 1–20

Thangaraj P, Balasubramanie P (2020) Meta heuristic QoS based service composition for service computing. J Ambient Intell Humaniz Comput

Von Wright GH (1951) Deontic logic. Mind 60(237):1–15

Wang HC, Lee CS, Ho TH (2007) Combining subjective and objective QoS factors for personalized web service selection. Expert Syst Appl 32(2):571–584

Wieringa RJ, Meyer JJ (1993) Applications of deontic logic in computer science: a concise overview. Deontic Logic Comput Sci 17–40

Woźniak M, Połap D (2019) Intelligent home systems for ubiquitous user support by using neural networks and rule-based approach. IEEE Trans Ind Inform 16(4):2651–2658

Wozniak M, Wieczorek M, Silka J, Polap D (2020) Body pose prediction based on motion sensor data and recurrent neural network. IEEE Trans Ind Inform

Yu T, Zhang Y, Lin KJ (2007) Efficient algorithms for web services selection with end-to-end QoS constraints. ACM Trans Web 1(1):6–32

Zeng L, Benatallah B, Ngu AHH, Dumas M, Kalagnanam J, Chang H (2004) Qos-aware middleware for web services composition. IEEE Trans Softw Eng 30(5):311–327

Zhang T (2014) Qos-aware web service selection based on particle swarm optimization. J Netw 9(3):565

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.