Applying Formal Verification Techniques to Ambient Assisted Living Systems

Kawtar Benghazi, María Visitación Hurtado, María Luisa Rodríguez, and Manuel Noguera

Software Engineering Department,
University of Granada, Spain
{benghazi,mhurtado,mlra,mnoguera}@ugr.es
http://lsi.ugr.es

Abstract. This paper presents a verification approach based on timed traces semantics and MEDISTAM-RT [1] to check the fulfillment of non-functional requirements, such as timeliness and safety, and assure the correct functioning of the Ambient Assisted Living (AAL) systems. We validate this approach by its application to an Emergency Assistance System for monitoring people suffering from cardiac alteration with syncope.

Key words: Ambient assisted living, safety, timeliness.

1 Introduction

Advances in networking, sensors and embedded devices have made it feasible to monitor and provide medical and other assistance to people at home. The general goal of Ambient Assisted Living (AAL) solutions is to apply ambient intelligence technology to enable people with specific demands, e.g. handicapped or elderly, to live in their preferred environment. In order to achieve this goal, different kinds of AAL systems can be proposed and most of them pose reliability issues and describe important constraints upon the development of software systems [13]. In this regard, the specification and verification of non-functional requirements in the early stages of their development cycle is crucial issue [5]. In [7] non-functional requirements for these systems such as robustness, availability, extensibility, safety, security, timeliness, resource efficiency are summarizes. In this paper we focus on safety and timeliness properties. Additionally, AAL systems require clear and precise specifications in order to describe the system behavior and its environment. The formal specification of the system behavior supported by mathematical analysis and reasoning techniques improve the development process and enable the verification of these systems.

The main purpose of this paper is to present a verification approach based on MEDISTAM-RT and timed traces to assure the correct functioning of AAL systems and show the applicability of this methodology in the context of this kind of systems. In the area of specifying and verifying security properties, trace-based formalism have been adopted [4] [6].

R. Meersman, P. Herrero, and T. Dillon (Eds.): OTM 2009 Workshops, LNCS 5872, pp. 381–390, 2009.

Other proposals that use formal methods in AAL systems have been developed. Formal description techniques and particularly the ICO (Interactive Cooperative Objects) notation, based on Petri nets, support usability evaluation, contextual help and incident and accident investigation in health care systems is presented in [9]. In [10], an advanced counseling system in health care is generated out by a formal specification (a process calculus for specifications in space and time). However, verification of non-functional properties such us safety and timeliness have not been considered.

The paper is organized as follows: in the next section we present our approach to formalization of non-functional requirements and formal systems verification. In section 3, a case study highlighting the applicability of the proposed technique is showed. Finally, section 4 concludes the paper with a discussion of the contributions of our approach.

2 Systems Specification and Verification

In this section we explain how we carry out the specification and verification of critical systems starting by modeling the system using a semiformal notation based on UML-RT. The behavior of each one of the components of a critical system architecture has strong requirements that should always be satisfied. The formal specification of this behavior and the requirements these components should fulfill allows to verify that the components work as expected. In our approach, we use CSP+T to specify the behavior of each system component and timed traces to describe the requirements the component must satisfy and to specify the valid sequences of messages that the components interchange through its ports.

Notation:

- A timed trace $s = \langle (s_1, t_1), (s_2, t_2) \dots (s_n, t_n) \rangle$, represents a sequence of timed events (s_i, t_i) than can be observed during the execution of a system.
- $s \uparrow [t_1, t_2]$, represents the projection of the timed traces s on an interval of time $[t_1, t_2]$.
- $\sigma(s)$, represents the elements of a trace s.
- $t : event \mapsto R^+$ is the function that maps events to its time occurrence.

2.1 Formalization of Non-functional Requirement

The properties to be satisfied by a system or a process are defined in terms of timed traces. This definition characterizes some traces as acceptable and some as non-acceptable. A process complies with its specification if all its executions are acceptable, that is, none of its executions by the system violates its specification. Likewise, if $S(tr)$ is a predicate in the timed trace tr, then it is said that P $satisfies$ or $complies$ with $S(tr)$ if $S(tr)$ holds for any timed trace $tr \in F_{TT}(P)$.

$$P \textbf{ sat } S(tr) \Leftrightarrow \forall tr \in F_{TT}(P); S(tr)$$

For example, when smoke is detected in a room (in a time t_s), a firefighting system must check if the door is closed to try to avoid the spreading of the fire, but only when the room is empty. Otherwise, if there are people inside the room, the system should allow people to exit first. We can formalize this situation by means of a property that the system must comply in order to avoid risks: $\forall tr \in F_{TT}(P)$

$$S(tr) = if(closed, t_c) \in \sigma(tr) \Rightarrow (empty, t_e) \in \sigma(tr \uparrow [t_s, t_c])$$

The process $\mathbf{Q} = smoke \bowtie t_s \rightarrow empty \bowtie t_e \rightarrow closed \bowtie t_c \rightarrow Q$ is defined by the set of timed traces

$$\begin{aligned}\mathbf{F_{TT}}(\mathbf{Q}) = &\{\langle\rangle, \langle(smoke, t_s)\rangle, \\ &\langle(smoke, t_s)\rangle^\frown \langle(empty, t_e)\rangle, \langle(smoke, t_s)^\frown(empty, t_e)^\frown(closed, t_c)\rangle\}\end{aligned}$$

and for all $tr \in F_{TT}(Q)$ the property $S(tr)$ is satisfied, that is:

$$\mathbf{Q\ sat}\ \ if(closed, t_c) \in \sigma(tr) \Rightarrow (empty, t_e) \in \sigma(tr \uparrow [t_s, t_c])$$

The specification of a system in the timed trace model or in the general trace model, allows **safety** properties and conditions to be defined. These properties require that *"nothing bad happens"* [11]. In our context, *something bad* is modeled by means of timed traces which do not satisfy $S(tr)$. Thus, the specification of safety properties can be modeled by forbidding the occurrence of a timed trace s during the execution of a process P, or by forbidding the occurrence of an event in a period of time $\forall tr \in F_{TT}(P)$.

$$S(tr) = s\ \mathbf{not\ \ in}\ tr$$

2.2 Formal System Specification Using MEDISTAM-RT

MEDISTAM-RT [2] [1] provides a methodological framework to the formal specification of real–time systems by combining semiformal languages based on UML-RT [8] and formal language based on process algebra[12] (see Fig. 1). This combination is based on the strategy of *integration by derivation* [3], which consists in firstly designing the semi-formal models (UML-RT models) and then obtaining (by applying a set of transformation rules, described in [2]) their equivalents in the formal language CSP+T .

In this methodology, the system is designed in a stepwise refinement manner, where the components are divided hierarchically into subcomponents until obtaining basic components[1]. The behavior of these basic components are separately designed by a Timed State Diagram (TSD), and the behavior of the composite ones are deduced from the behavior of its constituents by following a compositional specification process based on CSP+T [12].

[1] *Indecomposable* components, i.e. those that not contains any subcomponents.

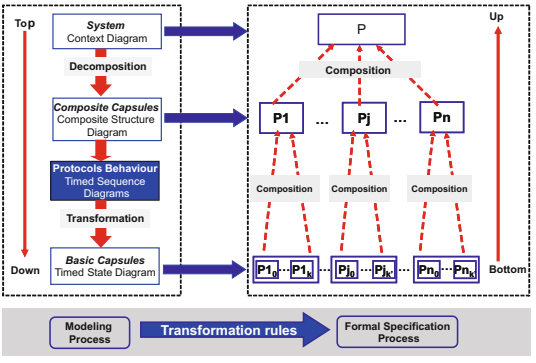


Fig. 1. Methodological framework MEDISTAM–RT

2.3 System Verification

The protocols determines the pattern of interchange of messages between the involved ports, which belong to different components. Actually, a timed sequence diagram (TSeD) representing the behavior of a given protocol gives a general view of the interaction between components. Hence, the protocol TSeD specify the valid sequences and time restrictions of the messages interchange between a component and its environment In this way, a component is correct if the temporal constraints and message order in the TSeD of the protocol is maintained in the component behaviour. As it is shown in figure 2, our verification approach take place in two stages:

1. Verify that the valid sequences obtained by transforming the timed sequence diagram into timed traces *satisfies* the component requirements[2].

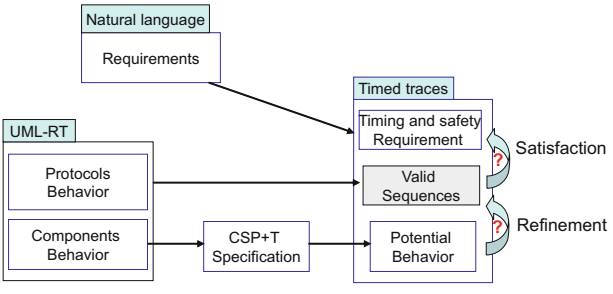


Fig. 2. Verification approach

2. The potential behavior of the system obtained by transforming the timed state diagram into CSP+T (using the set of rules presented in previous work [1]) and

[2] Described by timed traces as well.

then into timed traces (applying the semantic function $F_{TT} : CSP{+}T\ process \mapsto timed\ traces$ defined in [1]) *refine* the valid sequences for the system.

3 Case Study

As a case study we have modeled an Emergency Assistance System for monitoring people suffering from cardiac alteration with syncope. In this case, biomedical sensors that are usually embedded into textiles are used. These sensors are equipped with data storage capabilities and wireless transceiver systems. The data are sent to a Home Care System (HCS) where they are analyzed. In case of anomaly, and considering context information (e.g., patient's medical record), the HCS will determine the degree of seriousness of the situation and will act accordingly, namely:

1. Make a warning call to the patient.
2. Call a health center and monitor ambient conditions of the room or house (i.e., air temperature).
3. Call the Emergency Medical Service and monitor ambient conditions of the room or house.

Home is equipped with a series of sensors and actuators (i.e., temperature sensors and an air conditioner controller). Patients may communicate, by means of a PDA, that they are carrying out certain activities which may influence in their cardiac variations (e.g., because he/she is doing exercises) and are part of the context information. Although the one described it is not a *hard* real-time system, some of their actuations must be carried out within a window of time. Figure 3 depicts the logic system architecture of the system described.

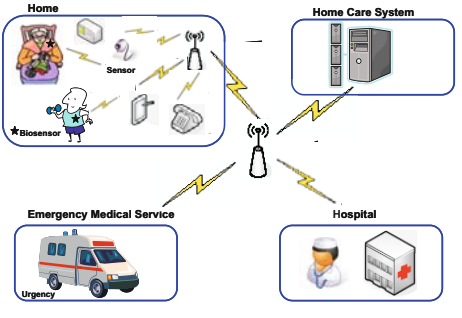


Fig. 3. Logical Architecture of the Emergency Assistance System

3.1 Requirements in Timed Traces

The *CareSystem* should satisfies the following timing and safety requirements:

1. No more than (6, 3, 2) units of time can elapse since it has been detected that a patients pulse is high before notifying the patient, hospital or the urgency respectively (We annotate this time by *tp*). Formally,

$\forall tr \in F_{TT}(CareSystem)$,

$$\mathbf{S_1(tr)} = \begin{cases} tp < t(Advise) \leq tp + 6 \\ \quad and \\ tp < t(Warning) \leq tp + 3 \\ \quad and \\ tp < t(Alert) \leq tp + 2 \end{cases}$$

2. Before sending the advice to the patient or warning to the hospital, the system may ask the patient whether he/she is exercising. Formally, $\forall tr \in F_{TT}(CareSystem)$,

   $S_2(tr) = if(Advise, t(Advice) \in \sigma(tr \uparrow [tp, tp + 6])) \Rightarrow Ask\_if\_exercise \in \sigma(tr \uparrow [tp, tp + 3]))$

   and

   $S_3(tr) = if(Warning, t(Warning) \in \sigma(tr \uparrow [tp, tp + 3])) \Rightarrow Ask\_if\_exercise \in \sigma(tr \uparrow [tp, tp + 3]))$

3.2 CareSystem Specification and Verification

We model the *Emergency_Assistance* system following the MEDISTAM-RT methodology. Table 1 shows the context of the *Emergency_Assistance* system and its internal structure. Table 2 shows the timed sequence diagram of the *CareSystem* which describes the protocols behavior of the *CareSystem* ports.

Table 1. Context and Composite Structure Diagrams

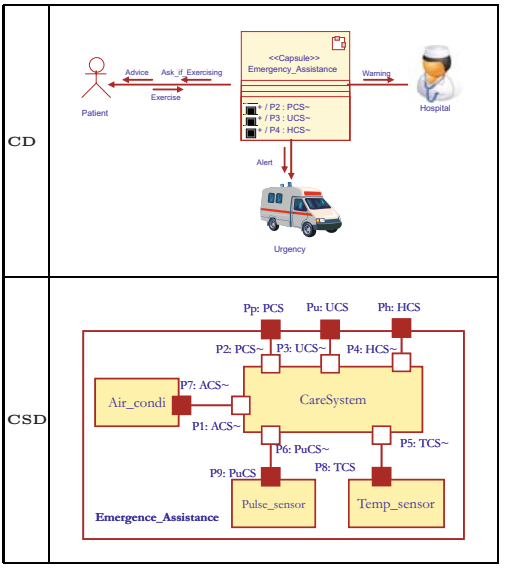


Table 2. *CareSystem* Timed Sequence Diagram

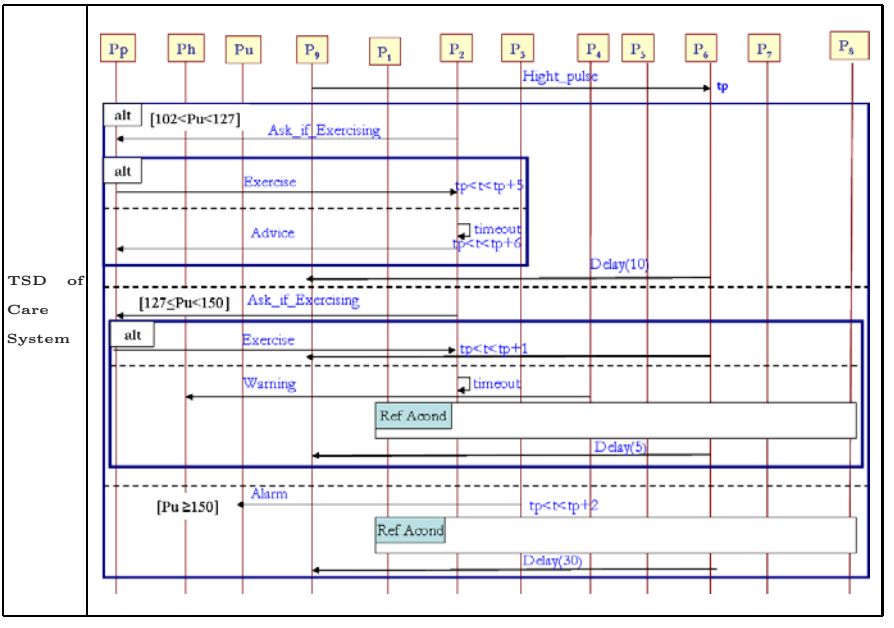


Table 3 resume the mapping of this diagram into timed traces applying a set of rules established in [1].

Table 3. Mapping of Timed Sequence Diagram into Timed traces

Mapping to Timed Traces	$\mathbf{TSeD} = (?Hight\_pulse, tp)^\frown$ $(((Ask\_if\_Exercising, t1)^\frown(?Exercise, tp < t \leq tp + 5)$ $\mid (timeout, tp + 5)^\frown(?advise, t2))^\frown Delay(10))$ $^\frown (((Ask\_if\_Exercising, t1)$ $^\frown(?Exercise, tp < t \leq tp + 1) \mid (timeout, tp + 1)^\frown(?Warning, t2))$ $^\frown TSeD^\frown_{Acond} Delay(5)) ^\frown ((Alarm, tp < t < tp + 2) ^\frown Read\_temp^\frown$ $TSeD^\frown_{Acond} Delay(30))$

Comparing the specification $TSeD$ in Table 3 and the specification of the requirement $S_1(tr)$, $S_2(tr)$ and $S_3(tr)$, we deduce that:

$$TSeD \ \mathbf{sat} \bigwedge_{i \in 1..3} S_i(tr) \tag{1}$$

Table 4 shows the timed state diagram of the *CareSystem* which is designed to describe it's internal behavior and the mapping of this diagram into syntactical

Table 4. *CareSystem* Timed State Diagram

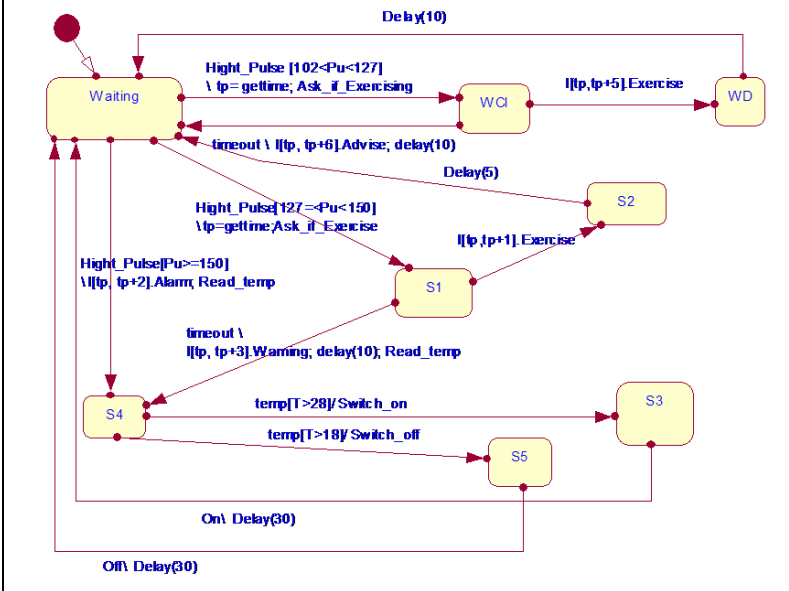
Timed State Diagram of Care System	

Table 5. Mapping of Timed State Diagram into CSP+T

Mapping to CSP+T processes	$\begin{aligned} \mathbf{TSD} &= (*.0 \rightarrow Waiting) \\ Waiting &= ((?Hight\_pulse[102 < Pu \leq 127] \bowtie tp)^\frown (Ask\_if\_Exercising) \rightarrow WCI) \\ &\quad Hight\_pulse[127 < Pu \leq 150] \bowtie tp)^\frown (Ask\_if\_Exercising) \rightarrow S1) \\ &\quad \mid (?Hight\_pulse[Pu > 150] \bowtie tp)^\frown I[tp, tp+2].Alarm \rightarrow Read_t emp \rightarrow S4) \\ WCI &= (I[tp, tp+5].Exercise \rightarrow WD) \mid \\ &\quad (timeout^\frown I[tp, tp+6].Advise^\frown Delay(10) \rightarrow Waiting) \\ WD &= Delay(10)^\frown Waiting \\ S1 &= (I[tp, tp+1].Exercise \rightarrow S2) \mid \\ &\quad (timeout^\frown I[tp, tp+3]^\frown Warning^\frown Read_t emp^\frown Delay(10) \rightarrow S4) \\ S4 &= (temp[T > 28]^\frown Switch_o n \rightarrow S3) \\ &\quad \mid (temp[T < 18]^\frown Switch_o ff \rightarrow S5) \\ S3 &= On \rightarrow Delay(30) \rightarrow Waiting \\ S5 &= Off^\frown Delay(30) \rightarrow Waiting \end{aligned}$

process CSP+T. To be able to verify the refinement relation between timed state diagram into timed sequence diagram we transform TSD (in Table 4) into timed traces applying the semantic function F_{TT} described in [1]. The result of this mapping is showed in Table 5 and 6, respectively.

Table 6. Mapping the TSD specification into timed Traces

Timed behaviour of *CareSystem* in terms of timed traces
$\mathbf{F_{TT}(TSD)} = (*, 0) \frown F_{TT}(Waiting)$
$F_{TT}(Waiting) = (?Hight\_pulse, tp) \frown ((Ask\_if\_Exercising, t1)$
$\frown (F_{TT}(WCI) \mid F_{TT}(S1)) \mid ((Alarm, tp < t < tp + 2) \frown$
$(Read\_temp, t) \frown F_{TT}(S4)$
$F_{TT}(WCI) = (?Exercise, tp < t \leq tp + 5) \mid$
$(timeout, tp + 5) \frown (?advise, t2)) \frown Delay(10)) \frown F_{TT}(Waiting)$
$F_{TT}(S1) = (?Exercise, tp < t \leq tp + 1) \frown Delay(5) \frown F_{TT}(Waiting)$
$F_{TT}(S4) = (temp, t2) \frown ((Switch\_on \frown On \mid ((Switch\_off \frown Off)$
$\frown Delay(30) \frown F_{TT}(Waiting)$

Comparing the specifications TSD and $TSeD$, we conclude that:

$$TSD \models TSeD \tag{2}$$

From the equation 1 and 2, we deduce that:

$$TSD \ \mathbf{sat} \bigwedge_{i \in 1..3} S_i(tr) \tag{3}$$

The equation 3 means that the component *CareSystem* fulfill its requirements.

4 Conclusions

Ambient Intelligence technologies are widely developed to construct safe environments around assisted people and help them maintain independent living. In order to enable elderly people and people with specific needs to live an independent and safe life, AAL systems have to cope with a series of non-functional requirements (i.e., safety and timeliness). To assure the correct functioning of these systems we have presented a verification approach based on the methodology MEDISTAM-RT, whereby both requirements and system behavior are captured in timed traces. The satisfaction and refinement relations can be checked in the timed traces semantic model. As a consequence, we can verify whether a system behavior specification fulfills its requirements.

An emergency assistance system for monitoring people suffering from cardiac alteration has been used to specify and verify non-functional requirements by means of the proposed approach.

Acknowledgments. This research is funded by the Spanish Government's Ministry of Science and Innovation, via the project TIN2008-05995/TSI.

References

1. Benghazi, K.: MEDISTAM–RT: Metodología de diseño y análisis de sistemas de tiempo real. PhD Thesis, University of Granada, Spain (2009)
2. Benghazi, K., Capel, M.I., Holgado, J.A., Mendoza, L.E.: A methodological approach to the formal specification of real-time systems by transformation of UML-RT design models. Science of Computer Programming 65(1), 41–56 (2007)
3. Fraser, M.D., Kumar, K., Vaishnavi, V.K.: Strategies for incorporating formal specifications in software development. Commun. ACM 37(10), 74–86 (1994)
4. Gray, J.W., McLean, J.: Using Temporal Logic to Specify and Verify Cryptographic Protocols. In: Proc. Eighth Comp. Sec. Found. Workshop (CSFW 1995), pp. 108–117 (1995)
5. Nehmer, J., Becker, M., Karshmer, A., Lamm, R.: Living assistance systems: an ambient intelligence approach. In: ICSE 2006: Proceeding of the 28th international conference on Software engineering, pp. 43–50. ACM Press, New York (2006)
6. Mantel, M., Sabelfeld, A.: A Generic Approach to the Security of Multithreaded Programs. In: Proc. 14th Comp. Sec. Found. Workshop (CSFW 2001), Cape Breton, Nova Scotia, Canada, June 2001, pp. 126–142 (2001)
7. Ras, E., Becker, M., Koch, J.: Engineering Tele-Health Solutions in the Ambient Assisted Living Lab. In: 21st International Conference on Advanced Information Networking and Applications (AINA), Niagara Falls, Canada (2007)
8. Selic, B., Rumbaugh, J.: Using UML for Modeling Complex Real-Time Systems, ObjecTime Limited/Rational Software whitepaper (1998)
9. Palanque, P., Basnyat, S., Navarre, D.: Improving Interactive Systems Usability Using Formal Description Techniques: Application to Health Care. In: Holzinger, A. (ed.) USAB 2007. LNCS, vol. 4799, pp. 21–40. Springer, Heidelberg (2007)
10. Herzberg, D., Marsden, N., Leonhardt, C., Kübler, P., Jung, H., Thomanek, S., Becker, A.: Using Formal Specification Techniques for Advanced Counseling Systems in Health Care. In: Holzinger, A. (ed.) USAB 2007. LNCS, vol. 4799, pp. 41–54. Springer, Heidelberg (2007)
11. Schneider, S.: Concurrent and real-time systems the CSP approach. John Wiley & Sons, Ltd., Chichester (2000)
12. Zic, J.: CSP+T: a Formalism for Describing Real-Time Systems. PhD Thesis, University of Sydney, Australia (1991)
13. Cleland-Huang, J., Settimi, R., Zou, X., Solc, P.: Automated classification of non-functional requirements. Requirements Eng. (12), 103–120 (2007)